20 Years of *Real* Real Time Model Validation *

Kim G. Larsen, Florian Lorber, and Brian Nielsen

Department of Computer Science, Aalborg University, Aalborg, Denmark
{kgl,florber,bnielsen}@cs.aau.dk

Abstract. In this paper we review 20 years of significant industrial application of the Uppaal Tool Suite for model-based validation, performance evaluation and synthesis. The paper will highlight a number of selected cases, and discuss successes and pitfalls in achieving industrial impact as well as tool sustainability in an academic setting.

1 Introduction

In 1995 the first release of the real-time verification tool Uppaal [43] was presented – together with a number of other emerging tools such as HyTeCH and Kronos – at the very first TACAS conference [15]. Soon after the tool was used for off-line verification of a number of *real* (i.e. industrially used) protocols, where real-time aspects were of essence. Today, in 2018, the most recent branches of Uppaal are applied for on-line optimization of home automation and traffic control systems. In this short note, we aim to recall some of the success stories of Uppaal over the years in terms of industrial applications, discuss what it takes to achieve lasting industrial take-up as well as reflect on the influence on the development of the tool from industrial feedback.

An overview of the most important case studies which will be discussed whithin this paper can be found in Table 1.

The remainder of the paper will be structured as follows: first, in Section 2, we will give an overview of the Uppaal tool family. Then, in Section 3 we will present our major use cases in the context of verification. Afterwards, in Section 4, we present our case studies in the area of testing and in Section 5 we will present cases in which we used Uppaal for scheduling and controller synthesis. Finally, in Section 6, we will present the most important lessons we learned while working on the presented case studies.

2 The Uppaal Tool Suite

This section will give an overview over the Uppaal tool family, its components and their main purposes.

* Work supported by Innovation Center DiCyPS, DFF project ASAP, and the ERC Advanced Grant Project Lasso.

Usecase	Tool	Goal	Partners	Outcome
PACP	UPPAAL	Verification	Philips	Significant tool improvement
BRP	UPPAAL	Verification	Philips, Twente University	Protocol verified
BOP	UPPAAL	Verification	Bang & Olufsen	Bug found and corrected
BOPC	UPPAAL	Verification	Bang & Olufsen	Frequency limits identified
FR	UPPAAL	Verification	FlexRay Consortium	Improved fault-tolerance guarantees
FW	UPPAAL	Verification	Radboud University	Sound timing restrictions identified
GC	UPPAAL	Verification	MECEL, Uppsala University	Several requirements verified
HPS	UPPAAL	Verification	Herchel & Planck	Schedulability of task-set established
NMN	UPPAAL	Verification	Neocortec	Energy performance of protocol
D	UPPAAL TRON	Testing	Danfoss	Demonstration of feasibility of online testing
NN	UPPAAL YGGDRASIL UPPAAL CORA	Testing	Novo Nordic	Two times industrial takeup
G	UPPAAL YGGDRASIL	Modelling, Testing	Grundfos	Interest provoked - new collaboration
S	UPPAAL TIGA	Controller Synthesis	Skov	Synthesis of zone-based controller
H	UPPAAL TIGA	Controller Synthesis	Hydac, ULB, ENS Cachan	Improved controller
BPNS	UPPAAL STRATEGO	Scheduling	GomSpace	Batterty life improvement of a satellite
HA	UPPAAL STRATEGO	Controller Synthesis	Seluxit	Intelligent floor heating
ICTL	UPPAAL STRATEGO	Controller Synthesis	Municipality of Køge	Efficient traffic controller

Fig. 1. Industrial Use Cases using UPPAAL.

UPPAAL The underlying formalism of UPPAAL is that of timed automata with the tool providing support for model checking of hard real-time properties. Since the introduction of the tool in 1995, significant effort have been put into development and implementation of improved datastructures and algorithms for the analysis of timed automata. Besides the several advances with respect to the verification engine, significant effort has over the years been put on the graphical interface of the tool (e.g. [8]), and on the modelling side the introduction of user-defined, structured datatypes and procedures has undoubtedly made the tool significantly more usable in modeling real control programs and communication protocols [7].

UPPAAL CORA Motivated by the need for addressing (optimal) usage of resources, priced timed automata were introduced in 2001. [9,4] (independently) demonstrated decidability of cost-optimal reachability. Soon after, an efficient priced extension of the symbolic datastructures used in UPPAAL was implemented in the branch UPPAAL CORA. Combined with a symbolic A* algorithm UPPAAL CORA turned into a new generic tool for cost-optimal planning which was competitive to traditional OR methods such as Mixed-Integer Linear Programming [39].

UPPAAL TRON In 2004 the branch UPPAAL TRON was introduced offering the possibility of performing on-line conformance testing of *realistic* real-time systems with respect to timed input-output automata [45,41]. UPPAAL TRON implements a sound and (theoretically) complete randomized testing algorithm, and uses a formally defined notion of correctness to assign verdicts: i.e. relativized timed input/output conformance providing a timed extension of Jan Tretmans ioco [52]. Using online testing, events are generated and simultaneously executed on the system under test.

UPPAAL YGGDRASIL is an off-line test case generator integrated into the main UPPAAL component. It aims at creating a test suite for edge coverage in a three phase process, which includes testing according to user-specified test purposes, random testing, and afterwards reachability analysis towards uncovered transitions. The tool enables the user to associate test code with transitions and locations, which is integrated into the test case whenever a trace traverses them. This enables UPPAAL YGGDRASIL to create test scripts in any desired language, which can be executed directly by the chosen execution engine.

UPPAAL TIGA In 2005 - encouraged by suggestions from Tom Henzinger – the branch UPPAAL TIGA was released, allowing for control strategies to be synthesized from timed games, i.e. two-player games played on timed automata [16,6]. The branch implements an efficient symbolic on-the-fly algorithm for synthesizing winning strategies for reachability, safety as well as Büchi objectives and taking possible partial observability into account [17]. The branch marks a disruptive direction with respect to development of control programs for embedded systems: rather than manually developing the control program with subsequent

model checking (and correction), UPPAAL TIGA provides a fully automatic method for deriving a correct-by-construction control program.

ECDAR In 2010 the branch ECDAR was introduced supporting a scalable methodology for compositional development and stepwise refinenemet of real-time systems [30,29]. The underlying specification theory is that of timed I/O automata being essentially timed games (with inputs being controllable, and outputs being uncontrollable) equipped with suitable methods for refinement checking (in terms of an alternating simulation between two timed game specifications), consistency checking, logical as well as structural composition. For a full account of ECDAR we refer the reader to the tutorial [28].

UPPAAL SMC One of the most recent branches of the UPPAAL tool suite – UPPAAL SMC introduced in 2011 – allows for performance evaluation on the expressive formalisms of stochastic hybrid automata and games [26,27], and has by now been widely applied to analysis of a variety of case studies ranging from biological examples [25], schedulability for mixed-critical systems [22,14], evaluation of controllers for energy-aware buildings [19], social-technical attacks in security [31], as well as performance evaluation of a variety of wireless communication protocols [53,53]. For a full account of UPPAAL SMC we refer the reader to the recent tutorial [24].

UPPAAL STRATEGO from 2014 [21,20] is the most recent branch of the UPPAAL tool suite that allows to generate, optimize, compare and explore consequences and performance of strategies synthesized for stochastic priced timed games (SPTG) in a user-friendly manner. In particular, UPPAAL STRATEGO comes with an extended query language, where strategies are first class objects that may be constructed, compared, optimized and used when performing (statistical) model checking of a game under the constraints of a given synthesized strategy.

3 Verification

The early development of UPPAAL was highly driven by colleagues in the Netherlands using the tool for automatic verification of industrial protocols. During a time-span of only a few years this resulted in a huge performance improvement reducing both time- and space-consumption with over 99%.

Philips Audio Control Protocol (PACP) Before the release of UPPAAL Bosscher, Polak and Vaandrager had in 1994 modelled and verified a protocol developed by Philips for the physical layer of an interface bus that connects the various devices of some stereo equipment (tuner, CD player,...). Essentially – after a suitable translation – the model of the protocol is a timed automata. Whereas the first proof in [13] was manual, the first automated verification of the protocol was done using the tool HyTech. Later, automated – and much faster – verifications were obtained using UPPAAL and Kronos. However, all these proofs were based

on a simplification on the protocol, introduced by Bosscher et. al. in 1994, that only one sender is transmitting on the bus so that no bus collisions can occur. In many applications the bus will have more than one sender, and the full version of the protocol by Philips therefore handles bus collisions. Already in the autumn of 1995 an automatic analysis of a version of the Philips Audio Control Protocol with two senders and bus collision handling was achieved using UPPAAL 0.96. To make the analysis feasible a notion of *committed location* was introduced (to remove unnecessary interleavings) and the analysis was carried out on a super computer, a SGI ONYX machine [11]. The total verification time was 8.82 hrs using more 527.4 MB. It is interesting to note that using UPPAAL 3.2 the same verification was reduced to only 0.5 sec using 2.5 MB of memory. In any case, the success in 1996 was a true milestone in the development of UPPAAL as this version of the protocol was orders of magnitude larger than the previously considered version with only one sender, e.g. the discrete state-spaces was 10^3 times larger and the number of clocks and channels in the model was also increased considerably.

Bounded Retransmission Protocol(BRP) In parallel with the collaboration with the group of Vaandrager, a group from Twente University (D'Argenio, Katoen, Reus and Tretmans) was also applying – and seriously testing – the first versions of UPPAAL. In particular, they successfully modelled and verified the Bounded Retransmission Protocol, a variant of the alternating bit protocol introduced by Philips. In [18] it is investigated to what extent real-time aspects are important to guarantee the protocol's correctness using UPPAAL and the Spin model checker.

B&O Protocol (BOP) In 1996, we were ourselves approached by Bang & Olufsen with a request of "analysing their proprietary IR Link protocol". The protocol, about 2800 lines of assembler code, was used in products from the audio/video company Bang&Olufsen throughout more than a decade, and its purpose was to control the transmission of messages between audio/video components over a single bus. Such communications may collide, and one essential purpose of the protocol was to detect such collisions. The functioning was highly dependent on real-time considerations. Though the protocol was known to be faulty (in the sense that messages were lost occasionally), the protocol was too complicated in order for the company to locate the bug using normal testing. However - after 4-5 iterations refining the model of the protocol - an error trace was automatically generated using UPPAAL and confirmed in the actual implementation of the protocol. Moreover, the error was corrected and the correction was automatically proven correct, again using UPPAAL [36].

B&O Powerdown control (BOPC) [35] Our first collaboration with Bang & Olufsen was very much characterized as a reverse engineering exercise of an existing protocol: the only documentation of the protocol was the 2800 lines of assembler code together with 3 flow-charts and a (very) knowledgeble B&O engineer. In our second collaboration with the company, modelling and verification in UPPAAL was carried out in parallel with the actual implementation

of a new real-time system for power-down control in audio/video components. During modeling 3 design errors were identified and corrected, and the following verification confirmed the validity of the design but also revealed the necessity for an upper limit of the interrupt frequency. The resulting design was later (seamlessley) implemented and incorporated as part of a new product line.

Whereas the above collaborative projects with B&O were very successful, neither UPPAAL nor model-driven development were taken-up in the company. An obvious reason could the immaturity (and lack of GUI) of the tool back then. However, in retrospect, an other equally likely reason is the fact that we were spending (all) our effort in collaborating with technicians in the company and not on marketing our tool and "disruptive" methodology to decision-makers in the company.

Flexray (FR) As part of the German DFG project AVACS [1] the FlexRay protocol was modeled and verified using UPPAAL. Flexray is a standard, developed by a cooperation of leading companies in the automotive industry, as a robust communication protocol for distributed components in modern vehicles. Developed by the FlexRay Consortium, a cooperation of leading companies including BMW, Bosch, Daimler, Freescale, General Motors, NXP Semiconductors, and Volkswagen, FlexRay was first employed in 2006 in the pneumatic damping system of BMW's X5, and fully utilized in 2008 in the BMW 7 Series. The FlexRay specification was completed in 2009 and is widely expected to become the future standard for the automotive industry. In [34] a timed automata model of its physical layer protocol is presented, and UPPAAL is used to automatically prove fault tolerance under several error models and hardware assumptions. In particular, it is shown that the communication system meets, and in fact exceeds, the fault-tolerance guarantees claimed in the FlexRay specification.

Firewire (FW) The IEEE 1394-1995 serial bus standard defines an architecture that allows several components to communicate at very high speed. Originally, the architecture was designed by Apple (FireWire), with more than 70 companies having been involved in the standardisation effort. In [50] a timed automata model of the leader election protocol is presented and its correctness is established using UPPAAL. In particular, it is shown that under certain timing restrictions the protocol behaves correctly. The timing parameters in the IEEE 1394 standard documentation obey the restrictions found in this proof.

MECEL Gear Controller (GC) In [44] an application of UPPAAL to the modelling and verification of a prototype gear controller was developed in a joint project between industry and academia. In particular, the project was carried out in collaboration between Mecel AB and Uppsala University. Within the project, the (timely) correctness of the controller was formalized (and verified) in 47 logical formulas according to the informal requirements delivered by industry.

[1] http://www.avacs.org

Herchel & Planck Schedulatilibity (HPS) In the danish project DaNES, we collaborated with the company Terma on using timed automata model checking as a more exact method for establishing schedulability of a number of periodic tasks executing on a single CPU under a given scheduling policy. In particular a fixed priority preemptive scheduler was used in a combination with two resource sharing protocols, and in addition voluntary task suspension was considered. In [46] schedulability was established under the assumption of exact computation times of the tasks. In [23] non-deterministic computations times were considered; depending on the size of the computation time interval, schedulability was either verified (using UPPAAL) or refuted (using the concrete search engine of UPPAAL SMC).

4 Testing

Our research on model-based test generation for timed (event recording) automata started with the thesis work around 1996-2000 in [47]. The approach aimed at covering timed equivalence classes defined through the clock guards of the timed automata. It assumed strictly deterministic systems, and its scalability was limited by the analysis techniques of the time. It thus had limited industrial applicability [48,49].

Later (2002-2004), inspired by [52,32], we developed the online testing tool UPPAAL TRON[3]. This approach could effectively handle non-determinism in both the specification (due to abstraction) and system under test (due to uncertainties in scheduling, execution times, timing, etc.), scaled to large models, and provided response times low enough for many practical cases [42,5,51]. Online testing generates effective randomized long tests, but coverage must be evaluated post-mortem and cannot be guaranteed a priori. Moreover, it is difficult to repeat the precise same test and inspect the set of test cases (might be required by certification bodies).

Our first work on offline test-case generation (with Uppsala University) appeared [37] in 2003. Here we showed how to interpret witness traces generated by the UPPAAL model-checker as test cases for the sub-class deterministic output urgent timed automata. Specifically, we showed how to generate the test cases with the minimum duration that satisfied a given test purpose formulated as a reachability property by exploiting UPPAAL's fastest witness trace generation feature. We furthermore formulated coverage as a reachability question, giving the ability to generate (time optimal) tests that guarantee meeting common coverage criteria. This work led to the UPPAAL COVER tool (no longer developed) and UPPAAL YGGDRASIL.

The Danfoss Case (D) We applied and evaluated UPPAAL TRON on an embedded controller supplied by the company Danfoss' Refrigeration Controls Division around year 2003-2004 [42]. The target device was a stable product of a refrigerator controller for industrial and large supermarket installations. As computer scientists we did not have domain expertise, and it soon became clear

that the supplied documentation (high-level requirements and user manuals) was insufficient for us to build accurate models. Hence, we ended up formulating a hypothesis model, running the test, and refining the model when the test failed. The final model consisted of 18 concurrent components (timed automata), 14 clock variables, and 14 discrete integer variables, and was thus quite large for the time. When confronting the refined model with Danfoss engineers, they too were surprised about certain aspects of its behavior, and needed to have that confirmed by other developers. Although we found no confirmed defects, the case showed that our techniques were practically applicable, and effective in finding discrepancies between specified and observed behavior. Encouraged by these results, both parties continued the collaboration on automated testing. At the end, our testing approach was not included in their new test setup that emphasized a new test harness for automated execution of manually defined scripts. Retrospectively, the gap between our method and their established development processes and tools was too big.

The Novo Nordic Case (NN) The first version of Uppaal Yggdrasil was developed in 2007-2009 specifically to support a collaboration with Novo Nordic for model-based GUI testing for medical devices. This version used Uppaal CORA as back-end, and operated in a 3 step process inspired by the company's needs: 1) Generating a separate test sequence for each user defined (supposedly critical) test purpose, 2) using Uppaal's search heuristics for optimizing model (edge) coverage considering constraints on the maximum lengths of the test cases, and 3) generating targeted test cases for each of the remaining uncovered transitions. The actual test case code was generated from model-annotations that the test engineers added to the model issuing appropriate GUI commands and assertions. Initially, the models were made using UML state-charts (and then translated into the Uppaal syntax) due to the engineers familiarity with this notation. It is important to remark that the engineers had no prior experience with formal modelling, and models were made for illustrative purposes using Microsoft Visio. Even then, making models that now had a tangible and formal meaning required a substantial training period. First the models were jointly developed assisted by the tool developer, and later only by company engineers with ordinary support.

This approach reduced the time used on test construction from upwards of 30 days to 3 days spent modelling and then a few minutes on actual test generation. At the same time, coverage was easier to establish than in the manual approach, and script maintenance greatly reduced. Later again, the company started using the Uppaal-editor directly, circumventing a heavy (and costly) UML tool. The approach was thus successfully embedded within the company. Unfortunately, that development team was dissolved as part of a company restructuring a year later, and the competence was no longer used.

MBAT Since the original Uppaal Yggdrasil was tailormade for this collaboration, and since it used the Uppaal CORA engine that is also no longer being developed, it ended up in a non-usable state. Recently, as part of the EU

Artemis MBAT (Combined Model-based Testing and Analysis) project, we re-architected the tool, and integrated it into — and shipped with — the main branch of UPPAAL, such that it now 1) uses the normal search engine, and 2) uses the graphical editor to create the needed annotations, and 3) provides a GUI widget for creating the test case configurations.

UPPAAL YGGDRASIL was applied to a case-study [38], and evaluated positively by a few consortium member companies. However, the collaboration did not result in commercial exploitation, partly because the project came to an end, and partly because we did not have an established company that could sell the licenses, and required maintenance, training, and consultancy.

MBAT also facilitated further developments for tool interoperability that is seen as crucial for large companies owning hundreds of various software development tools. That included prototyping of Open Services for Lifecycle Collaboration (OSLC)[2] adaptors for UPPAAL, and prototyping of Functional Mock-up Interfaces (FMI)[3] co-simulation interfaces. So it is regretful that this source of funding for Artemis/ECSEL industrial collaboration at a European scale ceased, as the Danish government halted national co-funding.

Grundfos (G) Grundfos is a major Danish company and world renowned for its pump products. In a recent meeting in the context of the DiCyPS project[4], we discussed different possible topics for further evaluation, including model-based testing. Based on our positive experiences with Danfoss (whose refrigerator controllers at an abstract level are similar to Grundfos pump controllers) we presented all the benefits/strengths of online model-based tested. However, it was when we presented offline testing that their interest was really triggered. They in particular liked our idea of modelling each of their requirements, using this (combined) model to automatically generate test scripts, and executing these on their existing test harness. Hence, there is a strong fit with their existing testing process and equipment. Also they believed that the (formalized) requirement models could be a valuable documentation complementing the existing design documentation. Hence, we decided to focus the collaboration on this approach, and postpone online testing.

In the first phase, we (university/tool provider/academics) perform the modelling and test case generation in order to prepare the tool and evaluate the method, for this particular case. We have identified an interesting, non-trivial subsystem of a newly developed pump controller exhibiting core functionality. If this stage is successful we plan to train selected Grundfos engineers and evaluate their experiences. Since the collaboration is ongoing, we cannot report on the outcome here.

[2] `https://open-services.net`

[3] `http://fmi-standard.org`

[4] National Innovation Fund supported project on Data-Intensive Cyber-Physical Systems.

5 Planning, Scheduling and Synthesis

Within its newer branches, the Uppaal tool suite allows for the usage of prices and stochastic elements, in order to enable various features, such as cost-optimal reachability, optimal scheduling or synthesis of strategies. The first practical step in this direction was made in 2002, with the initial release of Uppaal CORA. Uppaal CORA was developed as part of the VHS and AMETIST projects, and uses *linear priced timed auomata* (LPTA) for reachability problems, searching for paths with the lowest accumulated costs. The idea behind Uppaal Stratego came up in the CASSTING project. It was released in 2014, and facilitates the generation, optimization, comparison as well as consequence and performance exploration of strategies for *stochastic priced timed games* (SPTGs) in a user-friendly manner. The tools were since applied in several case studies, such as optimal planning of missions for battery-powered nano-satellites [12], efficient heating in home automation [40] or traffic light scheduling [33]. Below we will give an overview of the three mentioned case studies.

Battery-Powered Nano-Satellites (BPNS) This case study focused on the battery consumption of a GOMX-3 satellite built by the company GomSpace. It contains several antennas, solar panels and a battery. Depending on the scheduling of the different tasks of the satellite, the deterioration of the battery may vary significantly, depending on, for instance, the depth the battery is discharged to before reloading it. Uppaal Stratego was used to analyze different battery usage profiles, to optimize the lifetime of the satellite. This was done via a wear score function, which ranked the profiles according to their impact on the battery life. Additionally, the satellite was modelled as an SPTG in an abstract way. It could choose between the four different experiment types with different strains on the battery. Using the reinforcement learning approach implemented in Uppaal Stratego we could near-optimize the scheduling of the experiments with respect to both the battery life and the number of experiments performed.

Home Automation (HA) In [40] we collaborated with the Danish company Seluxit within the European project CASSTING. Our focus was on using timed games to synthesize a controller for a floor heating system of a single family house. Each room of the house has its own hot-water pipe circuit, which is controlled based on the room temperature. The original system used a simple "Bang-Bang"-like strategy, which turned the heating on if the temperature fell below a certain threshold, and turned it back off if it exceeded another threshold. Our goal was to use weather forecast information to synthesize an improved control strategy. Due to the state-space explosion caused by the number of control modes, we could not apply Uppaal Stratego directly. To cope with this, we proposed a novel online synthesis methodology, which is periodically called and learns an optimal controller for a limited timeframe. We further improved this approach by applying compositional synthesis, making it scalable enough for the study. The controller could access the weather forecast for the next 45 minutes,

and used that information to shut down or start the valves much earlier than other controllers, resulting in substantial energy savings and increased comfort.

Intelligent Control of Trafic Light (ICTL) Within the Innovation Center DiCyPS we used Uppaal Stratego for the synthesis of an efficient traffic control strategy. The controller gains information about the traffic via radar detectors and aims at optimizing the total traffic flow in a given traffic light junction. The strategy optimizes the total delay, the queue length and the number of times the vehicles have to stop. Again the synthesis is done online, this time in 5 second intervals, during which the next operation of the traffic light is calculated. We investigated an existing intersection in the municipality of Køge, Denmark, and simulated it with the open source tool SUMO and the commercial tool VISSIM. The strategy computed by Uppaal Stratego could be integrated into these tools, to analyze the behaviour based on randomly generated traffic scenarios. We evaluated the strategies in comparison to a static controller and a so called Loop controller, under three types of traffic szenarios with low, medium and maximal traffic. For low traffic, all controllers performed very similar, with the Loop controller showing the best results and for medium traffic, all performed equally. However, for high traffic, Uppaal Stratego outperformed both other controllers significantly, essentially halving the expected waiting time [33].

6 Lessons Learned

Based on 20 years of practical experience in using Uppaal on industrial case studies – as illustrated by the list of case studies given in the previous sections – we believe that a number of lessons may be learned.

It is important to have a dedicated team consisting of committed developers and inquisitive researchers in order to develop efficient and usable tools. In addition, the tools developed must have an interface and functionality which fits the use-case company's tool-chain, development method, and knowledge.

Formal methods tools must fit development methodology applied by industry.

Having the tool developer applying it in close interaction with the industrial user – e.g. through collaborative projects – gives a strong incentive for achieving alignment with and impact on industrial methodology. The tool developer can then strive to align the tool and the industrial verification workflow, both by adapting the tool and by influencing the used methods.

Industrial impact requires an *evolution of both their methods and our tools*, potentially in several iterations of collaboration.

The exact formal notations need not be a show-stopper, as long as the notation used is engineer friendly, and supported by a well-designed user-interface. Using a familiar notation is helpful in reducing the entry barrier and learning curve.

Use of engineer friendly, yet formal, notation increases chances of impact.

Sustaining use may be difficult in a dynamic industrial environment, and requires several collaborations and/or repeated introduction. Follow-up projects can benefit this greatly.

Sustained industrial use needs repeated committed collaboration.

Tool development needs to be continuously sustained beyond the first case-study and paper-publication. This requires committed developers, continuous maintenance including bug fixing, making enhancements of usability, functions, performance, and performing testing, release management, license serving, ... This is obviously time consuming and requires financial support. More importantly, because formal tools often require specialized expertise knowledge, few of these tasks can be subcontracted to a generic software engineer. Hence, also academic recognition and rewards are needed for such developments that do not readily result in publications.

Tool development needs to be continuously sustained. This requires increased academic recognition to tool developers.

On the other hand, we ourselves only made few serious attempts at commercializing our tools beyond selling licenses. This is likely because we are researchers at heart.

Industrial impact could be increased by offering tools and consultancy on commercial terms through spin-out companies.

Finally:

A successful case study is not the same as industrial impact.

References

1. *Proceedings of the 18th IEEE Real-Time Systems Symposium (RTSS '97), December 3-5, 1997, San Francisco, CA, USA*. IEEE Computer Society, 1997.
2. *Third International Conference on the Quantitative Evaluation of Systems (QEST 2006), 11-14 September 2006, Riverside, California, USA*. IEEE Computer Society, 2006.
3. Marius Mikucionis and Kim G. Larsen and Brian Nielsen. T-uppaal: Online model-based testing of real-time systems. In P. Grunbacher, editor, *19th IEEE International Conference on Automated Software Engineering (ASE 2004) Proceedings*, pages 396–397, United States, 2004. IEEE Computer Society Press. ISSN ; 1068-3062.

4. R. Alur, S. La Torre, and G. J. Pappas. Optimal paths in weighted timed automata. In Benedetto and Sangiovanni-Vincentelli [10], pages 49–62.
5. H. R. Asaadi, R. Khosravi, M. R. Mousavi, and N. Noroozi. Towards model-based testing of electronic funds transfer systems. In F. Arbab and M. Sirjani, editors, *Fundamentals of Software Engineering - 4th IPM International Conference, FSEN 2011, Tehran, Iran, April 20-22, 2011, Revised Selected Papers*, volume 7141 of *Lecture Notes in Computer Science*, pages 253–267. Springer, 2011.
6. G. Behrmann, A. Cougnard, A. David, E. Fleury, K. G. Larsen, and D. Lime. Uppaal-tiga: Time for playing games! In W. Damm and H. Hermanns, editors, *Computer Aided Verification, 19th International Conference, CAV 2007, Berlin, Germany, July 3-7, 2007, Proceedings*, volume 4590 of *Lecture Notes in Computer Science*, pages 121–125. Springer, 2007.
7. G. Behrmann, A. David, K. G. Larsen, J. Håkansson, P. Pettersson, W. Yi, and M. Hendriks. UPPAAL 4.0. In *Third International Conference on the Quantitative Evaluation of Systems (QEST 2006), 11-14 September 2006, Riverside, California, USA* [2], pages 125–126.
8. G. Behrmann, A. David, K. G. Larsen, P. Pettersson, and W. Yi. Developing UPPAAL over 15 years. *Softw., Pract. Exper.*, 41(2):133–142, 2011.
9. G. Behrmann, A. Fehnker, T. Hune, K. G. Larsen, P. Pettersson, J. Romijn, and F. W. Vaandrager. Minimum-cost reachability for priced timed automata. In Benedetto and Sangiovanni-Vincentelli [10], pages 147–161.
10. M. D. D. Benedetto and A. L. Sangiovanni-Vincentelli, editors. *Hybrid Systems: Computation and Control, 4th International Workshop, HSCC 2001, Rome, Italy, March 28-30, 2001, Proceedings*, volume 2034 of *Lecture Notes in Computer Science*. Springer, 2001.
11. J. Bengtsson, W. O. D. Griffioen, K. J. Kristoffersen, K. G. Larsen, F. Larsson, P. Pettersson, and W. Yi. Verification of an audio protocol with bus collision using UPPAAL. In R. Alur and T. A. Henzinger, editors, *Computer Aided Verification, 8th International Conference, CAV '96, New Brunswick, NJ, USA, July 31 - August 3, 1996, Proceedings*, volume 1102 of *Lecture Notes in Computer Science*, pages 244–256. Springer, 1996.
12. M. Bisgaard, D. Gerhardt, H. Hermanns, J. Krcál, G. Nies, and M. Stenger. Battery-aware scheduling in low orbit: The gomx-3 case. In J. S. Fitzgerald, C. L. Heitmeyer, S. Gnesi, and A. Philippou, editors, *FM 2016: Formal Methods - 21st International Symposium, Limassol, Cyprus, November 9-11, 2016, Proceedings*, volume 9995 of *Lecture Notes in Computer Science*, pages 559–576, 2016.
13. D. Bosscher, I. Polak, and F. W. Vaandrager. Verification of an audio control protocol. In H. Langmaack, W. P. de Roever, and J. Vytopil, editors, *Formal Techniques in Real-Time and Fault-Tolerant Systems, Third International Symposium Organized Jointly with the Working Group Provably Correct Systems - ProCoS, Lübeck, Germany, September 19-23, Proceedings*, volume 863 of *Lecture Notes in Computer Science*, pages 170–192. Springer, 1994.
14. A. Boudjadar, A. David, J. H. Kim, K. G. Larsen, M. Mikucionis, U. Nyman, and A. Skou. Degree of schedulability of mixed-criticality real-time systems with probabilistic sporadic tasks. In *2014 Theoretical Aspects of Software Engineering Conference, TASE 2014, Changsha, China, September 1-3, 2014*, pages 126–130. IEEE Computer Society, 2014.
15. E. Brinksma, R. Cleaveland, K. G. Larsen, T. Margaria, and B. Steffen, editors. *Tools and Algorithms for Construction and Analysis of Systems, First International Workshop, TACAS '95, Aarhus, Denmark, May 19-20, 1995, Proceedings*, volume 1019 of *Lecture Notes in Computer Science*. Springer, 1995.

16. F. Cassez, A. David, E. Fleury, K. G. Larsen, and D. Lime. Efficient on-the-fly algorithms for the analysis of timed games. In M. Abadi and L. de Alfaro, editors, *CONCUR 2005 - Concurrency Theory, 16th International Conference, CONCUR 2005, San Francisco, CA, USA, August 23-26, 2005, Proceedings*, volume 3653 of *Lecture Notes in Computer Science*, pages 66–80. Springer, 2005.
17. F. Cassez, A. David, K. G. Larsen, D. Lime, and J. Raskin. Timed control with observation based and stuttering invariant strategies. In K. S. Namjoshi, T. Yoneda, T. Higashino, and Y. Okamura, editors, *Automated Technology for Verification and Analysis, 5th International Symposium, ATVA 2007, Tokyo, Japan, October 22-25, 2007, Proceedings*, volume 4762 of *Lecture Notes in Computer Science*, pages 192–206. Springer, 2007.
18. P. R. D'Argenio, J. Katoen, T. C. Ruys, and J. Tretmans. The bounded retransmission protocol must be on time! In E. Brinksma, editor, *Tools and Algorithms for Construction and Analysis of Systems, Third International Workshop, TACAS '97, Enschede, The Netherlands, April 2-4, 1997, Proceedings*, volume 1217 of *Lecture Notes in Computer Science*, pages 416–431. Springer, 1997.
19. A. David, D. Du, K. G. Larsen, M. Mikucionis, and A. Skou. An evaluation framework for energy aware buildings using statistical model checking. *SCIENCE CHINA Information Sciences*, 55(12):2694–2707, 2012.
20. A. David, P. G. Jensen, K. G. Larsen, A. Legay, D. Lime, M. G. Sørensen, and J. H. Taankvist. On time with minimal expected cost! In F. Cassez and J. Raskin, editors, *Automated Technology for Verification and Analysis - 12th International Symposium, ATVA 2014, Sydney, NSW, Australia, November 3-7, 2014, Proceedings*, volume 8837 of *Lecture Notes in Computer Science*, pages 129–145. Springer, 2014.
21. A. David, P. G. Jensen, K. G. Larsen, M. Mikucionis, and J. H. Taankvist. Uppaal stratego. In C. Baier and C. Tinelli, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 21st International Conference, TACAS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings*, volume 9035 of *Lecture Notes in Computer Science*, pages 206–211. Springer, 2015.
22. A. David, K. G. Larsen, A. Legay, and M. Mikucionis. Schedulability of herschel-planck revisited using statistical model checking. In T. Margaria and B. Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation. Applications and Case Studies - 5th International Symposium, ISoLA 2012, Heraklion, Crete, Greece, October 15-18, 2012, Proceedings, Part II*, volume 7610 of *Lecture Notes in Computer Science*, pages 293–307. Springer, 2012.
23. A. David, K. G. Larsen, A. Legay, and M. Mikucionis. Schedulability of herschel revisited using statistical model checking. *STTT*, 17(2):187–199, 2015.
24. A. David, K. G. Larsen, A. Legay, M. Mikucionis, and D. B. Poulsen. Uppaal SMC tutorial. *STTT*, 17(4):397–415, 2015.
25. A. David, K. G. Larsen, A. Legay, M. Mikucionis, D. B. Poulsen, and S. Sedwards. Statistical model checking for biological systems. *STTT*, 17(3):351–367, 2015.
26. A. David, K. G. Larsen, A. Legay, M. Mikucionis, D. B. Poulsen, J. van Vliet, and Z. Wang. Statistical model checking for networks of priced timed automata. In U. Fahrenberg and S. Tripakis, editors, *Formal Modeling and Analysis of Timed Systems - 9th International Conference, FORMATS 2011, Aalborg, Denmark, September 21-23, 2011. Proceedings*, volume 6919 of *Lecture Notes in Computer Science*, pages 80–96. Springer, 2011.

27. A. David, K. G. Larsen, A. Legay, M. Mikucionis, and Z. Wang. Time for statistical model checking of real-time systems. In G. Gopalakrishnan and S. Qadeer, editors, *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, volume 6806 of *Lecture Notes in Computer Science*, pages 349–355. Springer, 2011.
28. A. David, K. G. Larsen, A. Legay, U. Nyman, L. Traonouez, and A. Wasowski. Real-time specifications. *STTT*, 17(1):17–45, 2015.
29. A. David, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. ECDAR: an environment for compositional design and analysis of real time systems. In A. Bouajjani and W. Chin, editors, *Automated Technology for Verification and Analysis - 8th International Symposium, ATVA 2010, Singapore, September 21-24, 2010. Proceedings*, volume 6252 of *Lecture Notes in Computer Science*, pages 365–370. Springer, 2010.
30. A. David, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. Timed I/O automata: a complete specification theory for real-time systems. In K. H. Johansson and W. Yi, editors, *Proceedings of the 13th ACM International Conference on Hybrid Systems: Computation and Control, HSCC 2010, Stockholm, Sweden, April 12-15, 2010*, pages 91–100. ACM, 2010.
31. N. David, A. David, R. R. Hansen, K. G. Larsen, A. Legay, M. C. Olesen, and C. W. Probst. Modelling social-technical attacks with timed automata. In E. Bertino and I. You, editors, *Proceedings of the 7th ACM CCS International Workshop on Managing Insider Security Threats, MIST 2015, Denver, Colorado, USA, October 16, 2015*, pages 21–28. ACM, 2015.
32. R. G. de Vries and J. Tretmans. On-the-fly conformance testing using SPIN. *STTT*, 2(4):382–393, 2000.
33. A. B. Eriksen, C. Huang, J. Kildebogaard, H. Lahrmann, K. G. Larsen, M. Muniz, and J. H. Taankvist. Uppaal stratego for intelligent traffic lights. In *ITS European Congress*, 2017.
34. M. Gerke, R. Ehlers, B. Finkbeiner, and H.-J. Peter. Model checking the flexray physical layer protocol. In S. Kowalewski and M. Roveri, editors, *Formal Methods for Industrial Critical Systems*, pages 132–147, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
35. K. Havelund, K. G. Larsen, and A. Skou. Formal verification of a power controller using the real-time model checker UPPAAL. In J. Katoen, editor, *Formal Methods for Real-Time and Probabilistic Systems, 5th International AMAST Workshop, ARTS'99, Bamberg, Germany, May 26-28, 1999. Proceedings*, volume 1601 of *Lecture Notes in Computer Science*, pages 277–298. Springer, 1999.
36. K. Havelund, A. Skou, K. G. Larsen, and K. Lund. Formal modeling and analysis of an audio/video protocol: an industrial case study using UPPAAL. In *Proceedings of the 18th IEEE Real-Time Systems Symposium (RTSS '97), December 3-5, 1997, San Francisco, CA, USA* [1], pages 2–13.
37. A. Hessel, K. Larsen, B. Nielsen, P. Pettersson, and A. Skou. Time-optimal test cases for real-time systems. In *Proceedings of the 1st International Workshop on Formal Modeling and Analysis of Timed Systems (FORMATS 2003)*, volume 2791, pages 234–245, 2003. ISSN ; -.
38. J. H. Kim, K. G. Larsen, B. Nielsen, M. Mikučionis, and P. Olsen. Formal analysis and testing of real-time automotive systems using uppaal tools. In M. Núñez and M. Güdemann, editors, *Formal Methods for Industrial Critical Systems*, pages 47–61, Cham, 2015. Springer International Publishing.

39. K. G. Larsen, G. Behrmann, E. Brinksma, A. Fehnker, T. Hune, P. Pettersson, and J. Romijn. As cheap as possible: Efficient cost-optimal reachability for priced timed automata. In G. Berry, H. Comon, and A. Finkel, editors, *Computer Aided Verification, 13th International Conference, CAV 2001, Paris, France, July 18-22, 2001, Proceedings*, volume 2102 of *Lecture Notes in Computer Science*, pages 493–505. Springer, 2001.
40. K. G. Larsen, M. Mikucionis, M. Muñiz, J. Srba, and J. H. Taankvist. Online and compositional learning of controllers with application to floor heating. In M. Chechik and J. Raskin, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9636 of *Lecture Notes in Computer Science*, pages 244–259. Springer, 2016.
41. K. G. Larsen, M. Mikucionis, and B. Nielsen. Online testing of real-time systems using uppaal. In J. Grabowski and B. Nielsen, editors, *Formal Approaches to Software Testing, 4th International Workshop, FATES 2004, Linz, Austria, September 21, 2004, Revised Selected Papers*, volume 3395 of *Lecture Notes in Computer Science*, pages 79–94. Springer, 2004.
42. K. G. Larsen, M. Mikucionis, B. Nielsen, and A. Skou. Testing real-time embedded software using UPPAAL-TRON: an industrial case study. In W. H. Wolf, editor, *EMSOFT 2005, September 18-22, 2005, Jersey City, NJ, USA, 5th ACM International Conference On Embedded Software, Proceedings*, pages 299–306. ACM, 2005.
43. K. G. Larsen, P. Pettersson, and W. Yi. UPPAAL in a nutshell. *STTT*, 1(1-2):134–152, 1997.
44. M. Lindahl, P. Pettersson, and W. Yi. Formal design and analysis of a gear controller. *STTT*, 3(3):353–368, 2001.
45. M. Mikucionis, K. G. Larsen, and B. Nielsen. T-UPPAAL: online model-based testing of real-time systems. In *19th IEEE International Conference on Automated Software Engineering (ASE 2004), 20-25 September 2004, Linz, Austria*, pages 396–397. IEEE Computer Society, 2004.
46. M. Mikucionis, K. G. Larsen, J. I. Rasmussen, B. Nielsen, A. Skou, S. U. Palm, J. S. Pedersen, and P. Hougaard. Schedulability analysis using uppaal: Herschel-planck case study. In T. Margaria and B. Steffen, editors, *Leveraging Applications of Formal Methods, Verification, and Validation - 4th International Symposium on Leveraging Applications, ISoLA 2010, Heraklion, Crete, Greece, October 18-21, 2010, Proceedings, Part II*, volume 6416 of *Lecture Notes in Computer Science*, pages 175–190. Springer, 2010.
47. B. Nielsen. *Specification and Test of Real-Time Systems*. PhD thesis, Aalborg University, 2000.
48. B. Nielsen and A. Skou. Automated test generation from timed automata. In T. Margaria and W. Yi, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 7th International Conference, TACAS 2001 Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2001 Genova, Italy, April 2-6, 2001, Proceedings*, volume 2031 of *Lecture Notes in Computer Science*, pages 343–357. Springer, 2001.
49. B. Nielsen and A. Skou. Test generation for time critical systems: Tool and case study. In *13th Euromicro Conference on Real-Time Systems, Delft, The Netherlands, June 2001*, pages 155–162, 2001.
50. J. Romijn. A timed verification of the IEEE 1394 leader election protocol. *Formal Methods in System Design*, 19(2):165–194, 2001.

51. C. Rütz. Timed model-based conformance testing – a case study using tron: Testing key states of automated trust anchor updating (rfc 5011) in autotrust. b.sc. thesis, 2010.
52. J. Tretmans. A formal approach to conformance testing. C-19:257–276, 1993.
53. R. J. van Glabbeek, P. Höfner, M. Portmann, and W. L. Tan. Modelling and verifying the AODV routing protocol. *Distributed Computing*, 29(4):279–315, 2016.