SION Application Summer 1996

April 29, 1996

Part A

1 Title

Stepwise Refinement of Hybrid Systems

2 Applicant

prof.dr F.W. Vaandrager
Computing Science Institute
Faculty of Mathematics and Computer Science
University of Nijmegen
Toernooiveld 1
6525 ED Nijmegen
The Netherlands
phone: +31 24 - 36 52216
e-mail: Frits.Vaandrager@cs.kun.nl

3 Organization where research will be performed

Computing Science Institute
Faculty of Mathematics and Computer Science
University of Nijmegen
Toernooiveld 1
6525 ED Nijmegen
The Netherlands

4 Summary of proposal

Hybrid systems are systems that intermix discrete and continuous components, typically computers that interact with the physical world. Due to the rapid development of processor and circuit technology, such systems are becoming more and more common in all application domains, ranging from avionics and process control to robotics and consumer electronics. The specification, design and analysis of hybrid systems require a synthesis of ideas, concepts, mathematical theories and tools that are currently spread over distinct disciplines, most notably Computer Science and Control Theory. The interest in the formal treatment of digital systems that interact with an analog environment is certainly not new, but received a new impetus by the extension of formal models from Computer Science with real-time around 1991 and by the explosion of embedded applications of computer technology within our society. Today, the study of hybrid systems has grown into an area of research with rapidly increasing popularity.

The goal of the proposed project is to (1) develop a methodology for stepwise refinement of hybrid systems which allows one to prove that one hybrid system is an abstraction of another, and (2) to find hybrid system models that are computationally tractable, i.e. can be analyzed by algorithmic means. In order to reach this goal, the project will tackle a series of case studies, taken from or directly inspired by practical problems from the area of embedded systems in which a nontrivial interaction between discrete and continuous components takes place.

5 Classification

CR Categories: D.2.1., D.2.4, F.1.1, F.3.1.

6 Requested support

(Standard) support is asked to employ one OIO for a period of four years.

7 Signature and date

April 29, 1996

prof.dr. F.W. Vaandrager

Part B

8 Content

8.1 What are Hybrid Systems?

With the decrease in the size and price of computing elements, more and more computers are used within real-world technical applications such as in avionics, process control, robotics, telecommunications and consumer products. The software within such applications is called *embedded* since it determines to a large extent the functionality of the product; the software and the rest of the product have no meaning without each other. Typically, the initial purpose of introducing computer technology in products is to implement existing functionality cheaper. However, once the microprocessors are there, one quickly realizes that many new features can be realized via software as well, and this then often leads to a dramatic growth of the size and complexity of the software. For many products in the area of consumer electronics the amount of code is doubling every two years. As a result, for instance, a high-end television today contains about 600 Kbytes of software and a GSM hand held telephone 512 Kbytes of software. Another instructive statictics is that the number of microprocessors that we encounter daily in our lives has increased from 3 in 1985 to 50 in 1995 [Bou95]. These developments makes the construction of large real-time embedded computer systems one of the most challenging tasks facing the computer science community, if not the engineering community as a whole [BÅ93].

At the ICL/CEC workshop "Software 2000: a view of the future" [RRW94], it was estimated that the number of software engineers who design and write embedded software will increase from 2 million in 1994 to 10 million in 2004. Software engineers involved in the development of embedded software typically operate within large, multidisciplinary teams together with physicists, chemists, electrical and mechanical engineers, etc. Also in the Netherlands there is a substantial activity in the area of embedded software. There are many companies, ranging from very big to small, that either produce embedded software or use it in their products (Philips, Stork, Océ, Nedap, etc.). A recent study [IDC95] reveiled that in 1994 Dutch companies which, amongst others, develop embedded software, spent 28% of their R&D budget on embedded software.

Characteristic of embedded computers systems is the fact that they are *reactive*: they accept stimuli from the outside world and react to those stimuli. This means that one can *only* design and reason about the correctness of these systems if one takes the behavior of the outside world into account. As an example, consider a computer controlling a chemical plant (see Figure 1). Regularly, the control program reads sensor data, such as temperature and pressure, of the plant. Based on these data, the computer may decide to turn on a heating system, switch off a pump, etc. When a dangerous situation arises, for instance the pressure in a tank gets too high, the computer has to initiate appropriate action, like opening a valve. In order to formally prove correctness properties like "No explosion will ever occur", the underlying mathematical model needs to contain information about the way in which the chemical reactions take place, the pressure in the tank evolves, etc.

Traditionally, computer scientists take a discrete view of the world: they assume that both a computer system and its environment can be modeled as a discrete event system. This

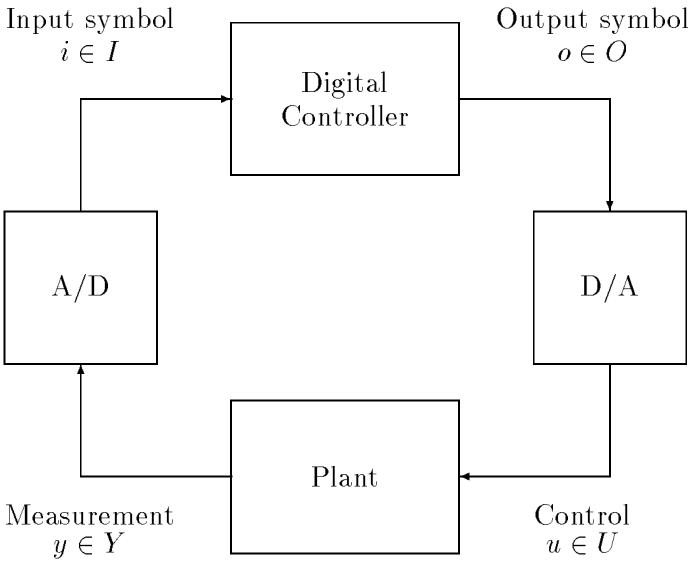

Figure 1: Hybrid System.

assumption is justified and has proved to be very successful within application areas such as operating systems and computer networks. However, we see more and more applications, in particular in the area of embedded computer systems, in which modeling the environment as a discrete process greatly distorts reality and may lead to unreliable conclusions [MMP92]. For these applications, the continuous models for the real-world developed by physicists and control engineers are typically more compact, more tractable and more accurate than the discrete approximations computer scientists tend to come up with. Since designing and reasoning about software for reactive systems is only possible if one has a model for both the software and the environment in which it operates, this provides a real practical motivation for the development of a comprehensive theory of *hybrid systems*, i.e., systems consisting of a non-trivial mixture of discrete and continuous components.

A real need for such a theory has also been identified by the control community [BÅ93, HYB93]. Being called to construct controllers for inherently unstable systems such as flight by wire of an unstable aircraft, it is obvious that such systems cannot be controlled by a classical control system employing a single control rule. The only solution, which is the one being implemented today for such systems is a hybrid system, in which a digital controller keeps switching between different continuous control laws or control modes very rapidly.

The real-world is usually modeled by control engineers as a continuous dynamical system, described in the language of functional analysis. Computer scientists investigate the dynamics of discrete systems, described in the language of mathematical logic. Since the construction of large embedded computer systems will always involve representatives of both cultures, a comprehensive theory of hybrid systems is required to avoid a Tower of Babel. Such a theory must comprise a synthesis of ideas, concepts, mathematical theories and tools that are currently spread over several disciplines, most notably Computer Science and Control Theory, but also Electrical Engineering, Mechanical Engineering, Physics and others.

Technically speaking, hybrid systems involve both continuous-valued and discrete variables. Their evolution is given by laws of motion that generally depend on all variables. These laws contain mixtures of logic, discrete-valued or *digital* dynamics, and continuous-variable or *analog* dynamics, generally given by differential equations or inclusions. The continuous and discrete dynamics interact at times when the continuous state hits certain prescribed sets in the continuous state space. At these points discrete state transitions may take place that can change both the continuous and the discrete state, as well as the analog dynamics of the system.

Below we mention a some real-world examples of hybrid systems (see also [Bra95]):

- Systems with relays, switches, and hysteresis [Wit66]. A prototypical example is the temperature in a thermostatically controlled room. The analog dynamics of temperature depends on the current temperature and whether the furnace is switched On or Off. The system switches from On to Off if temperature reaches a boundary value T_u, and goes back from Off to on On when another boundary value T_l is reached.

- Computer disk drives [GV89]. The behavior of the disk drive is governed by the differential (or difference) equations modeling the dynamic behavior of the disk, spindle, disk arm, and motors. The drive receives symbolic inputs of disk sectors and locations, and transmits symbolic outputs corresponding to the bytes read.

- Transmissions, stepper motors, and other motion controllers [Bro93]: the analog dynamics of a manual transmission, which involves continuous variables such as ground speed and engine RPM, depends on the value of discrete variables such as throttle position and gear shift position.

- Constrained robotic systems. A nice example here is provided by the hopping robots of Raibert of MIT [Rai86]. The dynamics of these devices are governed by gravity, as well as the forces generated by passive and active (pneumatic) springs. The dynamics change abruptly at certain event times, and fall into distinct phases: Flight, Compression, Thrust and Decompression. In fact, Raibert has built controllers for these machines that embed a finite state machine transitions according to these detected phases.

- Intelligent vehicle/highway systems [Var93]. A really complicated example of a hybrid system arises in the control structures for so-called intelligent vehicle and highway systems (IVHS). The basic goal of one such system is to increase highway throughput by means of a technique known as *platooning*. A platoon is a group of between, say, one and twenty vehicles traveling closely together (2m) in a highway lane at high speeds. It needs no arguing that this is a safety critical system! Protocols for basic maneuvers such as Merge, Split and ChangeLane have been proposed in terms of finite state machines, whereas more conventional controllers govern the engines and brakes of individual vehicles. The California PATH project [DV95], which aims at actually building such a system for some of the highways in California, has given a major impetus to the research on hybrid systems in the USA.

- Systems involving hardware/software co-design. An example here is the family of Manchester communication protocols studied in [BPV94, Moo94]. Protocols at the physical level of the OSI hierarchy aim at the transmission of bit streams between processors in a network. These protocols can be described in terms of finite state machines. However, the communication medium in between the sender and the receiver (a copper wire, for instance) is best described in terms of continuous models. In order to analyze the protocols and the allowed tolerance on timing, one has to take into account the finite state machines, the continuous model of the wire, and the interaction between these two.

Many more examples can be found in [BM95]: chemical batch plants, electromechanical processes, pulse detectors, industrial manufacturing processes, robotics, steam boilers, water tanks, aircraft landing gears, railroad crossings, etc.

Even though hybrid applications have been built in practice since the beginning of the fifties, it is only since a few years that the mathematical study of hybrid systems is starting to receive the attention that it deserves. The earliest reference we are aware of is the pioneering work of Witsenhausen from MIT, who formulated a class of hybrid-state, continuous-time dynamic systems and examined an optimal control problem [Wit66]. For a good overview of this and other early work with a control theory flavor, we refer to Branicki's PhD Thesis [Bra95].

Until the end of the seventies, most computer science research on verification was devoted to the analysis of programs for "autistic" (i.e. nonreactive) sequential batch computers. The eighties have witnessed a revolution in the formal methodology for the specification, verification, and development of reactive programs (and more general reactive systems) [dBRR89]. Once the concept of a reactive system was well understood, it turned out not to be so difficult to bring quantitative, real-time aspects into the picture. Even though the development of reliable real-time systems will remain an important research topic for many years to come, several important theoretical results concerning real-time systems were obtained by 1991 [dBHRR92].

Around that time, the first two important workshops on hybrid systems were being organized in Ithaca, NY, USA and Lyngby, Denmark [GNRR93], and triggered a rapidly increasing interest in this new area, both in computer science and in control theory [Cou93, Dil94, Vyt92, LRV94, PS95, Ben94, BÅ93].

Several semantic models for hybrid systems have been proposed. We refer to the Springer LNCS volume [GNRR93], where many of these models are described in papers such as [ACHH93, Hoo93, Lam93, MP93, NK93], to the recent Special Issue of Theoretical Computer Science [PS95], and to the recent PhD work of Branicki [Bra95] and Deshpande [Des94]. Even though there are many differences between the approaches, there is a surprising consensus that the behavior of a hybrid system is best represented as an alternating sequence of instantaneous "discrete" actions and "continuous" phases during which time advances. Within the continuous phases the system's variables vary continuously according to a control law, fixed for the interval. Via the discrete actions the system can move to a new state, where a new control law becomes valid.

This view of hybrid systems is highly compatible with the earlier computer science semantic models for real-time systems of [AL94, AD94, HMP92, LV92, LV93, LV94]: roughly speaking, the only difference is that where in real-time models the only available information about the continuous phases is the amount of time that passes, the hybrid models allow one to use differential equations, etc. to specify how precisely the system variables evolve in a continuous phase. Due to this similarity, many automaton based semantic models for timed systems generalize smoothly to hybrid systems [MMP92, ACHH93, NSY93, ACH+95, LSVW96].

In several cases it is possible to use verification techniques for timed systems directly for hybrid systems [BPV94, DY95, Lam93, OSY94]. Also in the area of specification logics many ideas generalize from timed to hybrid systems and logics such as TCTL [ACD93, HNSY94], ICTL [AHH93], TLA [Lam94] and the Calculus of Durations [CHR91] are being used in both settings.

However, there is a problem. Because the expressivity of the hybrid system models is enormous, the verification problem for these models is intrinsically difficult, even under severe restrictions. Typically, verification problems that are decidable in polynomial time for untimed systems (represented as finite transition systems), become exponentially hard for timed systems, and undecidable even for simple classes of hybrid systems [ACH+95]. A very promising line of research though is the work on (semi) decision procedures and tools for (subclasses of) the so-called *linear hybrid automata* [ACH+95, ACHH93, AHH93, ACD93, HNSY94, MV94, OSY94, PV94, AIKY95]. Two prototype tools have been developed, the KRONOS tool [NSY92, NSY93] in Grenoble and the HYTECH tool [AHH93, HHWT95, HH95b] at Cornell. Using these tools, their implementors have been able to verify automatically dozens of toy examples proposed in the literature on hybrid systems, as well as some practical examples from industry, such as the Philips audio control protocol of [BPV94]. The key idea behind these tools is to represent infinite sets of states as convex polyhedra in multi-dimensional space and to use standard data structures and geometric algorithms for polyhedral manipulation to do reachability analysis and model checking [Che68, Hal93].

Clearly, the work on semantic models, logics and verification methods for hybrid systems is just starting. In the case of untimed discrete event systems, a rich body of closely related theories has been created during the last twenty years involving temporal and modal logics [MP92, Sti92], assertional verification methods [LT87, LV95, AL93, Lam94], process algebras [Mil89, BW90], and tools for computer aided verification [Hol91, KL93, SR91, CPS93]. Most of this work still needs to be lifted to (or at least related to) the setting of hybrid systems.

8.2 Scientific Problem

As pointed out above, one can typically only reason about the correctness of embedded computer systems if one takes the behavior of the outside world into account. Consequently, hybrid system models will often describe both the discrete dynamics of a digital controller and the continuous dynamics of their environment. Correctness of the controller becomes a property that is defined relative to a model of the environment. This implies that, especially in the case of safety critical systems, it becomes vital to have a good model of the environment available at the "right" level of abstraction.

At this point however, a difficult dilemma arises. If one aims at an *accurate* model of the environment, one will typically end up with something highly complex, involving many states, nonlineair differential equations, etc., that is completely intractable. If, on the contrary, the aim is to obtain a *tractable* model with few states, only linear differential equations or inclusions, etc., then there is a big danger that the resulting structure is no longer an accurate model of the physical environment. In the case of embedded computer systems, the control engineers will usually be able to come up with accurate models of the physical environment. These models, developed over many years by engineering disciplines, are typically tractable when used to analyze the continuous systems for which they were designed. The discrete dynamics of the digital controller, however, adds a level of complexity that is still underestimated in many cases and that can make the overall model extremely complex.

One way to deal with this complexity is to build more than one model of a single system: first a model that is generally believed to be accurate (but possibly intractable), and then another model that is (more) tractable and a provable abstraction of the first model. The transformation from accurate to tractable may need several intermediate steps, and thus lead to a hierarchy of models of the same system.

In computer science, theories of *stepwise refinement* [LV95, LV93] have been developed to prove that one (discrete) system is an abstraction of another. In this body of work, the abstract model if typically a *specification* of the desired external behavior of a system, whereas the concrete model describes a proposed *implementation*. Within theories of stepwise refinement, *simulation relations* [LV95, LV93] provide a key technique to prove correspondence between two levels of abstraction. Depending on the abstraction level that is defined first, the theory of stepwise refinement is used both for *verification* and *design*. A related important piece of work in computer science is the theory of *abstract interpretations*. In this theory the only properties required of the abstract models is that (1) they are (computationally) tractable, and (2) if they satisfy a specific desired property the original system does so as well. Also in control theory there is a substantial body of work that allows one to relate system descriptions at different levels of abstraction, and (for instance) to approximate systems of nonlinear differential equations by linear systems of differential inclusions. In the area of hybrid systems, the only work in this direction that we are aware of is the work of Lynch et al on simulation relations for hybrid automata [LSVW96, Lyn96, WLD96], and the work of Henzinger and Ho [HH95a] in which nonlinear hybrid systems are approximated by linear ones.

The goal of the proposed project is to (1) develop a methodology for stepwise refinement of hybrid systems which allows one to prove that one hybrid system is an abstraction of another, and (2) to find hybrid system models that are computationally tractable, i.e. can be analyzed by algorithmic means. In order to reach this goal, the project will tackle a series of case studies, taken from or directly inspired by practical problems from the area of embedded systems in which a nontrivial interaction between discrete and continuous components takes place.

8.3 Significance for Development of Science

Hybrid systems are a new and very important area of research. Within the formal methods field the work on hybrid systems seems to be developing into one of the most productive areas in terms of potential applicability and foundational value. Internationally, the study of hybrid systems has grown into an area of research with rapidly increasing popularity in which many highly prestegeous research groups have become active (e.g., in Grenoble, Oxford, Lund, Aalborg, Berkeley, Cornell, Stanford and at MIT).

Currently, very little is known about synthesis of digital controllers for hybrid systems, and most of the efforts concentrate on run-time simulation (see for instance [And94]) and on verification. A solid body of theory about stepwise refinement of hybrid systems, even though developed primarily for verification purposes, may as a spinn-off also suggest some design principles for hybrid controllers.

The applicant has close contacts with many of these teams (notably Grenoble, Aalborg, Berkeley and MIT), and his work is quite visible within the hybrid systems community. This is witnessed for instance by the fact that he was invited to give presentations at two leading international meetings on hybrid systems in Grenoble, France, and New Brunswick, USA. His PIONIER proposal on hybrid systems was put forward by SION last year and got rejected only in the very last round.

9 Societal Usefulness

The construction of embedded software systems poses enormous problems to both science and technology (see [HV96] for an overview). One major problem is to give abstract specifications of the (desired) behavior of components in an embedded applications that are understandable for the different membes of a multi-disciplinary development team. If one leafes through a document with a state-of-the-art specification of an embedded application, one will encounter a mixture of architecture diagrams, programming text, flow charts, transition tables, diagrams by electrical engineers with IC's, transistors, resistors, etc., and diagrams made by mechanical engineers in which the mechanical parts of the system are described. The relations between the various parts of the specification are only described by informal text, there is no such thing as a common semantic framework for all the design notations that are used. This is undesirable, since in the design of embedded systems hardware and software are more and more viewed as interchangeable: often mechanical and electronic implementations of new and improved functions are replaced by software solutions (anyone can think of dozens of examples in the area of consumer electronics). The applicant believes that hybrid systems theory will be absolutely vital as a semantic basis for a new generation of formalisms in which *all* relevant elements in the design of embedded applications can be described formally in an integrated way. In order to be accepted by software and control engineers in industry, these new languages should contain (close variants) of design notations that are currently used as sublanguages. The role of hybrid system theory will mainly be one of glue by which different design notations can be formally related (cf. the "pivot" formalisms advocated by [BÅ93]).

Even though we currently wittness an explosion in the use of hybrid formalisms for the formal description and simulation of embedded applications (see, for instance, [BM95]), verification techniques for hybrid systems are still of limited applicability: automatic verification techniques perform well in the case of small tricky circuits like the audio control protocol of [BPV94], but are not designed to face the immense timing problems that arise in larger applications, say the control system for a sonar application outlined in [Boa93]. Nevertheless, one can envisage two important uses of this work in a large-scale application. First, the knowledge about tractable hybrid system models will be quite valuable in the mathematical formulation of embedded applications. Second, there will be isolated situations where the tools themselves can be applied.

Formal verification methods have not progressed to a state where a designer can analyze a complex protocol at the same time that he/she is designing the protocol: the two activities together are simply too time consuming. This is mainly because knowledge of formal methods and the inner workings of tool sets are crucial for creating tractable verification models. However, for the successful application of formal verification methods it is essential that the person who builds the verification models has a detailed knowledge of the system that is being designed. Therefore, we think that the right approach of the use of these advanced verification techniques in industry is that a designing team "hires" one or more verification specialists to assist them with their job. In recent years several small companies have been founded that specialize in formal and symbolic verification techniques, such as Cadence Berkeley Labs, CLInc. in Austin, Texas, Formal Systems Ltd in Oxford, ORA Corporation, Synopsys and Logikkonsult in Stockholm. We are convinced that the market for this type of companies will grow quickly.

Transfer of results of the projects will take place though (1) direct contacts with the companies/institutes from which the case studies are taken, (2) presentation of results at scientific meetings. The applicant is currently active as Program Chair of the IPA/BRICS/TUCS School on Embedded Systems, to be held in Veldhoven, The Netherlands, in November of this year. At this meeting attention will be paid both to very practical issues that arise in the construction of embedded systems, but also to hybrid system models.

10 Interdisciplinary Collaboration

By nature, research in the area of hybrid systems is a multi-disciplinary activity. On the one hand, the field is situated in between computer science and control theory. On the other hand, permanent interaction is required between the theoretical computer scientists/control persons who develop hybrid system models and the more practically oriented software-, electrical- and mechanical engineers who have expressed the need for such models in the design of embedded software. Bridges have to be built both between computer science and control theory, and between theory and practice of software design for technical applications.

For the project to be successful it is vital that good contacts exist with other research groups in hybrid systems and, more generally, in formal methods, system and control theory, embedded software and electrical/mechanical engineering. In addition, good contacts with industry are required to demonstrate and assess the effectiveness of the tools and techniques that are developed. Vaandrager has already many contacts with the internationally leading research groups in hybrid systems and with industry.

In order to manage the complexity of realistic systems, computer support is absolutely vital: if the verification or design steps are not carried out fully automatically, then at least the computer has to check each of the proof steps in a manual proof and fill in the holes in such a proof. At this moment, Vaandrager is involved in the SION project "Checking Verifications of Concurrent Systems with Type Theory Tools" (together with dr J.F. Groote, CWI), and we definitely expect to use the outcomes of this work. However, for the efficient formalization of correctness proofs for hybrid systems, we need tools that are both good in *logic* and in *algebra*, and such tools are not yet available. Fortunately, the main aim of the SION project "WINST: Themes for collaboration in mathematics and computer science" (prof. H.P. Barendregt, prof. J.W. Klop, prof. M. Hazewinkel and prof. A.M. Cohen) is exactly to build such tools, and thus we expect to benefit from cooperation with this project.

In Nijmegen, prof. J. Vytopil has much expertise in the area of embedded real-time systems from which the project can benefit. Good contacts exist with the System and Control Theory group of prof. J.H. van Schuppen at CWI. These contacts will be intensified. Clearly, also close contacts with groups in Electrical and Mechanical Engineering are desirable. In particular, collaboration will be sought with the Production Automation group of prof.J.E. Rooda of the Faculty of Mechanical Engineering at Eindhoven University of Technology, and (because of the expertise on switched bond graphs) the group of dr P.C. Breedveld of the Faculty of Electrical Engineering at the University of Twente. In fact, one of the aims of this project will be to bring together from different disciplines all the interested research groups in the Netherlands through the organization of monthly seminars, visits between sites, etc. At this moment the applicant is already involved in the organization of a biweekly seminar on timed and hybrid automata in Nijmegen.

11 National and International Initiatives or Activities

The proposal fits very well with the research topics that are put forward within the Dutch graduate school IPA (Institute for Programming Research and Algorithmics), notably with the Research priority Embedded Software. It is our aim to link this SION project closely to what is going on within IPA.

Together with prof. J.H. van Schuppen (CWI) and dr Oded Maler (Verimag, Grenoble), the applicant is currently preparing a European initiative in the area of Hybrid Systems.

12 Plans for Implementation

12.1 Activity planning

Year 1: getting acquainted with the relevant literature ;
taking courses offered by the graduate schools IPA and DISC;
study of several concrete examples of hybrid systems;
Year 2: work on stepwise refinement methods for hybrid systems;
additional courses;
Year 3: application of stepwise refinement methods on concrete examples;
work on tool support;
Year 4: write PhD Thesis.

12.2 Use of other facilities + financial planning

Nothing nonstandard.

12.3 Output

Technical reports, conference papers, journal articles, and Ph.D. Thesis.

13 Available Expertise

Although the applicant has ample experience with stepwise refinement methods as developed in computer science, additional expertise in system and control theory is absolutely essential to make further progress on the topic of the proposal. When hiring a researcher, we will therefore look for someone a background in system and control theory. Courses of the Graduate Program Systems and Control that are offered by the Dutch Institute of Systems and Control (DISC) may help to obtain additional expertise. Further we the project will closely collaborate with the group of prof. Van Schuppen at CWI.

Part C

14 Relevant publications of the applicant

[BPV94] D.J.B. Bosscher, I. Polak, and F.W. Vaandrager. Verification of an audio control protocol. In Langmaack et al. [LRV94], pages 170–192. Full version available as Report CS-R9445, CWI, Amsterdam, July 1994.

[HSV94] L. Helmink, M.P.A. Sellink, and F.W. Vaandrager. Proof-checking a data link protocol. In H. Barendregt and T. Nipkow, editors, *Proceedings International Workshop TYPES'93,* Nijmegen, The Netherlands, May 1993, volume 806 of *Lecture Notes in Computer Science*, pages 127–165. Springer-Verlag, 1994. Full version available as Report CS-R9420, CWI, Amsterdam, March 1994.

[HV96] D. Hammer and F.W. Vaandrager. Ingebouwde systemen. *Automatiserings Gids*, May 3, 1996.

[JSV93] A.S.A. Jeffrey, S.A. Schneider, and F.W. Vaandrager. A comparison of additivity axioms in timed transition systems. Report CS-R9366, CWI, Amsterdam, November 1993.

[LSVW96] N.A. Lynch, R. Segala, F.W. Vaandrager, and H.B. Weinberg. Hybrid I/O automata. In Alur et al. [AHS96]. To appear. Also available as Report CS-R9578, CWI, Amsterdam, December 1995.

[LV92] N.A. Lynch and F.W. Vaandrager. Forward and backward simulations for timing-based systems. In de Bakker et al. [dBHRR92], pages 397–446.

[LV93] N.A. Lynch and F.W. Vaandrager. Forward and backward simulations – part II: Timing-based systems. Report CS-R9314, CWI, Amsterdam, March 1993. Also, MIT/LCS/TM-487.b, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA. To appear in *Information and Computation.*

[LV94] N.A. Lynch and F.W. Vaandrager. Action transducers and timed automata. Report CS-R9460, CWI, Amsterdam, November 1994. Also, Technical Memo MIT/LCS/TM-480.b, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, October 1994. To appear in *Formal Aspects of Computing.*

[LV95] N.A. Lynch and F.W. Vaandrager. Forward and backward simulations. part I: Untimed systems. *Information and Computation*, 121(2):214–233, September 1995.

[RV95] J.M.T. Romijn and F.W. Vaandrager. A note on fairness in I/O automata. Report CS-R9579, CWI, Amsterdam, December 1995.

[Vaa96] F.W. Vaandrager. Hybrid systems. In *Images of SMC Research 1996.* CWI, 1996.

15 Other relevant publications

[ACD93] R. Alur, C. Courcoubetis, and D.L. Dill. Model checking in dense real time. *Information and Computation*, 104:2–34, 1993.

[ACH+95] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J.Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[ACHH93] R. Alur, C. Courcoubetis, T.A. Henzinger, and P.-H. Ho. Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems. In Grossman et al. [GNRR93], pages 209–229.

[AD94] R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[AHH93] R. Alur, T.A. Henzinger, and P.-H. Ho. Automatic symbolic verification of embedded systems. In *Proceedings of the 14th Annual Real-time Systems Symposium*, pages 2–11. IEEE Computer Society Press, 1993.

[AHS96] R. Alur, T. Henzinger, and E. Sontag, editors. *Hybrid Systems III*, Lecture Notes in Computer Science. Springer-Verlag, 1996.

[AIKY95] R. Alur, A. Itai, R. P. Kurshan, and M. Yannakakis. Timing verification by successive approximation. *Information and Computation*, 118(1):142–157, April 1995.

[AL93] M. Abadi and L. Lamport. Composing specifications. *ACM Transactions on Programming Languages and Systems*, 1(15):73–132, 1993.

[AL94] M. Abadi and L. Lamport. An old-fashioned recipe for real time. *ACM Transactions on Programming Languages and Systems*, 16(5):1543–1571, September 1994.

[And94] M. Andersson. *Object-Oriented Modeling and Simulation of Hybrid Systems*. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Sweden, December 1994.

[BÅ93] A. Benveniste and K.J. Åström. Meeting the challenge of computer science in the industrial applications of control: An introductory discussion to the special issue. *IEEE Transactions on Automatic Control*, 38(7):1004–1010, July 1993.

[Ben94] A. Bensoussan. New trends in applied mathematics. In K.R. Apt, A. Schrijver, and N.M. Temme, editors, *From Universal Morphisms to Megabytes – a Baayen Space Odyssey*, pages 153–178. CWI, Amsterdam, 1994.

[BM95] A. Bouajjani and O. Maler, editors. *Proceedings Second European Workshop on Real-Time and Hybrid Systems*, Grenoble, France, June 1995.

[Boa93] M. Boasson. Control systems software. *IEEE Transactions on Automatic Control*, 38(7):1094–1106, July 1993.

[Bou95] R.H. Bourgonjon. Software pervasion in the electronic industry, June 1995. Slides of talk presented at meeting of Dutch graduate school IPA.

[Bra95] M.S. Branicky. *Studies in Hybrid Systems: Modeling, Analysis, and Control.* PhD thesis, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA, June 1995.

[Bro93] R.W. Brocket. Hybrid models for motion control systems. In H.L. Trentelman and J.C. Willems, editors, *Essays on Control: Perspectives in the Theory and its Applications*, volume 14 of *Progress in Systems and Control Theory*, pages 29–53. Birkhäuser, Boston, 1993.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra.* Cambridge Tracts in Theoretical Computer Science 18. Cambridge University Press, 1990.

[Che68] N.V. Chernikova. Algorithm for discovering the set of all solutions of a linear programming problem. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 8(6):282–293, 1968.

[CHR91] Z. Chaochen, C.A.R. Hoare, and A.P. Ravn. A calculus of durations. *Information Processing Letters*, 40:269–276, December 1991.

[Cou93] C. Courcoubetis, editor. *Proceedings of the 5th International Conference on Computer Aided Verification,* Elounda, Greece, volume 697 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.

[CPS93] R. Cleaveland, J. Parrow, and B. Steffen. The concurrency workbench: A semantics based tool for the verification of concurrent systems. *ACM Transactions on Programming Languages and Systems*, 1(15):36–72, 1993.

[dBHRR92] J.W. de Bakker, C. Huizing, W.P. de Roever, and G. Rozenberg, editors. *Proceedings REX Workshop on Real-Time: Theory in Practice,* Mook, The Netherlands, June 1991, volume 600 of *Lecture Notes in Computer Science*. Springer-Verlag, 1992.

[dBRR89] J.W. de Bakker, W.P. de Roever, and G. Rozenberg, editors. *REX School and Workshop on Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency,* Noordwijkerhout, The Netherlands, May/June 1988, volume 354 of *Lecture Notes in Computer Science*. Springer-Verlag, 1989.

[Des94] A. Deshpande. *Control of Hybrid Systems.* PhD thesis, University of California at Berkeley, 1994.

[Dil94] D.L. Dill, editor. *Proceedings of the 6th International Conference on Computer Aided Verification,* Stanford, USA, volume 818 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.

[DV95] A. Deshpande and P.P. Varaiya. Design and evaluation tools for automated highway systems. In A. Bouajjani and O. Maler, editors, *Proceedings Second European Workshop on Real-Time and Hybrid Systems*, Grenoble, France, June 1995.

[DY95] C. Daws and S. Yovine. Two examples of verification of multirate timed automata with KRONOS. Technical Report Spectre-95-06, VERIMAG, Grenoble, France, April 1995. Also appeared in *Proceedings 2nd European Workshop on Real-Time and Hybrid Systems*, June 1995, Grenoble, France.

[GNRR93] R.L. Grossman, A. Nerode, A.P. Ravn, and H. Rischel, editors. *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*. Springer-Verlag, 1993.

[GV89] A. Göllü and P.P. Varaiya. Hybrid dynamical systems. In *Proceedings of the IEEE Conference on Decision and Control,* Tampa, pages 2708–2712, December 1989.

[Hal93] N. Halbwachs. Delay analysis in synchronous programs. In Courcoubetis [Cou93], pages 333–346.

[HH95a] T.A. Henzinger and P.-H. Ho. Algorithmic analysis of nonlinear hybrid systems. In *Proceedings of the 7th International Conference on Computer Aided Verification,* Liège, Belgium, Lecture Notes in Computer Science. Springer-Verlag, July 1995.

[HH95b] T.A. Henzinger and P.-H. Ho. HyTech: The Cornell HYbrid TECHnology Tool. In U.H. Engberg, K.G. Larsen, and A. Skou, editors, *Proceedings of the Workshop on Tools and Algorithms for the Construction and Analysis of Systems,* Aarhus, Denmark, volume NS-95-2 of *BRICS Notes Series*, pages 29–43. Department of Computer Science, University of Aarhus, May 1995.

[HHWT95] T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. HyTech: The next generation, 1995. Manuscript.

[HMP92] T.A. Henzinger, Z. Manna, and A. Pnueli. Timed transition systems. In de Bakker et al. [dBHRR92], pages 226–251.

[HNSY94] T.A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. *Information and Computation*, 111:193–244, 1994.

[Hol91] G.J. Holzmann. *Design and Validation of Computer Protocols.* Prentice-Hall International, 1991.

[Hoo93] J. Hooman. A compositional approach to the design of hybrid systems. In Grossman et al. [GNRR93], pages 121–148.

[HYB93] HYBRID. Tools for the analysis of hybrid systems. Technical annex, Esprit-NSF Proposal EC-US-043, 1993.

[IDC95] IDC. Software in industrie en diensten. In Dutch. Published on the occasion of the conference "Software in Nederland: Motor voor nieuwe produkten en Diensten?", Ministerie van Economische Zaken, Den Haag. IDC Benelux, May 1995.

[KL93] R.P. Kurshan and L. Lamport. Verification of a multiplier: 64 bits and beyond. In Courcoubetis [Cou93], pages 166–179.

[Lam93] L. Lamport. Hybrid systems in TLA$^+$. In Grossman et al. [GNRR93], pages 77–102.

[Lam94] L. Lamport. The temporal logic of actions. *ACM Transactions on Programming Languages and Systems*, 16(3):872–923, March 1994.

[LRV94] H. Langmaack, W.-P. de Roever, and J. Vytopil, editors. *Proceedings of the Third International School and Symposium on Formal Techniques in Real Time and Fault Tolerant Systems,* Lübeck, Germany, September 1994, volume 863 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.

[LT87] N.A. Lynch and M.R. Tuttle. Hierarchical correctness proofs for distributed algorithms. In *Proceedings of the* 6^{th} *Annual ACM Symposium on Principles of Distributed Computing*, pages 137–151, August 1987. A full version is available as MIT Technical Report MIT/LCS/TR-387.

[Lyn96] N.A. Lynch. Modelling and verification of automated transit systems, using timed automata, invariants and simulations, 1996. In Alur et al. [AHS96]. To appear.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall International, Englewood Cliffs, 1989.

[MMP92] O. Maler, Z. Manna, and A. Pnueli. From timed to hybrid systems. In de Bakker et al. [dBHRR92], pages 447–484.

[Moo94] J.S. Moore. A formal model of asynchronous communication and its use in mechanically verifying a biphase mark protocol. *Formal Aspects of Computing*, 6(1):60–91, 1994.

[MP92] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, 1992.

[MP93] Z. Manna and A. Pnueli. Verifying hybrid systems. In Grossman et al. [GNRR93], pages 4–35.

[MV94] J. McManis and P. Varaiya. Suspension automata: A decidable class of hybrid automata. In Dill [Dil94], pages 105–117.

[NK93] A. Nerode and W. Kohn. Models for hybrid systems: Automata, topologies, controllability, observability. In Grossman et al. [GNRR93], pages 317–356.

[NSY92] X. Nicollin, J. Sifakis, and S. Yovine. Compiling real-time specifications into extended automata. *IEEE Transactions on Software Engineering*, 18(9):794–804, September 1992.

[NSY93] X. Nicollin, J. Sifakis, and S. Yovine. From ATP to timed graphs and hybrid systems. *Acta Informatica*, 30(2):181–202, 1993.

[OSY94] A. Olivero, J. Sifakis, and S. Yovine. Using abstractions for the verification of linear hybrid systems. In Dill [Dil94], pages 81–94.

[PS95] A. Pnueli and J. Sifakis, editors. *Special Issue on Hybrid Systems of Theoretical Computer Science,* 138(1). Elsevier Science Publishers, February 1995.

[PV94] A. Puri and P. Varaiya. Decidability of hybrid systems with rectangular differential inclusions. In Dill [Dil94], pages 95–104.

[Rai86] M.H. Raibert. *Legged Robots that Balance*. MIT Press, Cambridge, MA, 1986.

[RRW94] B. Randell, G. Ringland, and B. Wulf, editors. *Software 2000: a View of the Future*. ICL and the Commission of the European Communities, Brussels, 1994.

[SR91] R. de Simone and V. Roy. Auto/Autograph. In E.M. Clarke and R.P. Kurshan, editors, *Proceedings of the 2nd International Conference on Computer-Aided Verification,* New Brunswick, NJ, USA, volume 531 of *Lecture Notes in Computer Science*, pages 65–75. Springer-Verlag, 1991.

[Sti92] C. Stirling. Modal and temporal logics. In S. Abramsky, D. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science, Vol 2*, pages 477–563. Oxford University Press, 1992.

[Var93] P.P. Varaiya. Smart cars on smart roads: Problems of control. *IEEE Transactions on Automatic Control*, 38(2):195–207, February 1993.

[Vyt92] J. Vytopil, editor. *Formal Techniques in Real Time and Fault Tolerant Systems*, volume 571 of *Lecture Notes in Computer Science*. Springer-Verlag, 1992.

[Wit66] H.S. Witsenhausen. A class of hybrid-state continuous-time dynamic systems. *IEEE Transactions on Automatic Control*, 11(2):161–167, 1966.

[WLD96] H.B. Weinberg, N.A. Lynch, and N. Delisle. Verification of automated vehicle protection systems, 1996. In Alur et al. [AHS96]. To appear.