SION Application

May 29, 1998

1 Part A

1.1 Title

Validation of Programmable Logic Controllers

1.2 Applicants

dr. Angelika Mader	prof.dr. Frits Vaandrager (project leader)	dr. Hanno Wupper
Phone: 024 - 3652271	Phone: 024 - 3652216	Phone: 024 - 3652227
Email: `mader@cs.kun.nl`	Email: `fvaan@cs.kun.nl`	Email: `Hanno.Wupper@cs.kun.nl`

1.3 Institution

All the applicants are affiliated with the following research institute, where the proposed research will also take place:

Computing Science Institute
Katholieke Universiteit Nijmegen
PO Box 9010
6500 GL Nijmegen

1.4 Summary of Proposal

One of the most important developments in the application of information technology is the almost dramatic growth of applications of so-called *embedded systems*. The main characteristic of an embedded system is its interaction in real-time with a physical environment, in contrast to more conventional IT-applications that interact with other digital systems and/or human users. This specific nature of embedded systems, that of their applications, and their widespread use makes their correct functioning an issue of general concern. It is not realistic to expect that a single theory or formal method can be used to cover the whole range of embedded systems. Instead, it is much more promising to identify interesting subclasses of embedded systems that allow for a more specialized, and therefore more effective treatment, and for which certain existing formal methods and tools are likely to provide a successful point of departure. Applications of Programmable Logic Controllers (PLCs) form such a class. They have an enormous range of applications including elevators, traffic lights, traffic guidance systems, household appliances, cars of higher price level etc.. They are increasingly used to control industrial processes and plants, especially at the lowest level of the automation hierarchy, viz. directly with the devices to be supervised, controlled and regulated. Many of their applications are safety critical, such as traffic control systems or chemical plants, or economically critical, where costs of a bug are measured in millions of dollars, e.g. for shutting down a factory.

There exist several different programming languages for PLCs. They have been designed with a main emphasis on applicability, each intended for a specific application domain, and based on the background of the control engineers who use them. However, from the point of view of computer scientists some of these programming languages lag behind several generations of development in programming methodology. The PLC programming languages of the standard IEC 1131-3 still contain semantic ambiguities, and no formal semantics is provided, which makes precise reasoning impossible.
The goal of the project proposed is to develop methods for the effective validation of industrial PLC applications. More specifically, we want to address the formal verification and systematic testing of the real-time behavior of such applications. In general, we want to take a pragmatic approach: we do not intend to develop completely new theories or to develop new tools, but rather plan to use as much as possible existing theories and verification tools. First step is to identify well-behaved (but still sufficiently expressive and relevant) subsets of the PLC programming languages from the standard IEC 1131-3, and provide those with a formal, operational semantics in terms of timed automata. Having such a semantics allows for making use of existing tool sets such as KRONOS, UPPAAL, and HyTech, which represent the actual state-of-the-art in automatic verification of real-time systems. For conformance testing of timed automata we want to tailor existing methods to the setting of PLCs.
We put emphasis on industrial case studies: for the semantic part we expect indication which subsets of PLC programming languages are relevant, and in which context. Also the kind of properties that have to be shown for real applications is relevant. It determines the necessary granularity of the formal model of PLCs and their programs. Altogether, we want to prove, modify and demonstrate our methods with hopefully successful case studies.
There are rich sources for industrial case studies available: Rijkswaterstaat who is a large scale consumer of PLC-controlled equipment, an ESPRIT project on Verification of Hybrid Systems, where the University of Nijmegen is one of the partners, and the German UniForM-project, where we have an ongoing cooperation with the University of Oldenburg, which is one of the partners.

1.5 CR Categories

C.3 Special-Purpose and Application-Based Systems: process control systems, real-time systems;
D.2 Software Engineering: Program Verification, Testing;
F.3 Logics and Meanings of Programs.

1.6 Support Requested

1 postdoc for 3 years.

1.7 Signature Applicants and Date

Date:	Date:	Date:
dr. A.H. Mader	prof.dr. F.W. Vaandrager	dr. H. Wupper

2 Part B: Proposal

2.1 Content

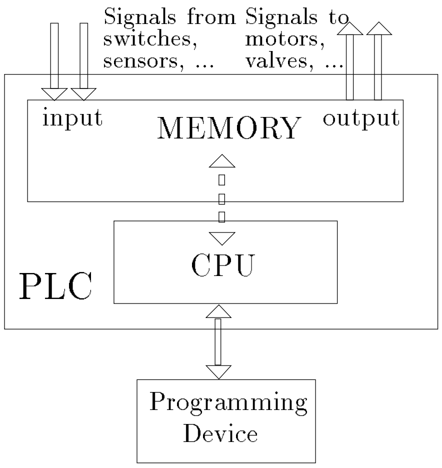
2.1.1 Background

Embedded systems One of the most important developments in the application of information technology is the almost dramatic growth of applications of so-called *embedded systems* [RV98]. Embedded systems are computer systems that form an integral part of a larger system like an automated factory, a controller for a home heating system, a television set, or an airplane. The term "embedded systems" encompasses thus a broad class of systems, ranging from simple micro-controllers to large and complex multi-processors and distributed systems. In general engineering terms, embedded systems are used for the control of industrial or physical processes. Sensors continuously gather information from the environment. The service of the embedded system is to process this information and to signal the actuators in accordance with the mechanisms of the controlled process.
The main characteristic of an embedded system is its interaction in real-time with a physical environment, in contrast to more conventional IT-applications that interact with other digital systems and/or human users. This specific nature of embedded systems, that of their applications, and their widespread use makes their correct functioning an issue of general concern. Not only are such systems embedded in many safety-critical applications, for which the importance of their correct operation is obvious, but many applications of embedded systems are highly replicated (e.g. cars, consumer electronics), which greatly increases the potential cost of malfunctioning, making such systems economically critical. As physical environments are characterized by continuous phenomena, this complicates the interaction with discrete digital controllers. Moreover, physical environments are often less forgiving with respect to errors than, for example, human users. It can therefore be said that the correctness of embedded systems is at the same time a more pressing as well as a more complicated problem as with traditional information technology.
To achieve correct, reliable embedded systems we must have a good understanding of their functioning and adequate models, notations, theories, and tools to support their design and realization. It is not realistic to expect that a single theory or formal method can be used to cover the whole range of embedded systems. Instead, it is much more promising to identify interesting subclasses of embedded systems that allow for a more specialized, and therefore more effective treatment, and for which certain existing formal methods and tools are likely to provide a successful point of departure. Applications of Programmable Logic Controllers (PLCs) form such a class.

PLCs Historically, PLCs replaced hard-wired control. The possibility of having one kind of hardware that can be adapted to a specific control task by simply changing the software was much more flexible than to build new hardware for each problem. Now, PLCs are of great technical and economic interest as they provide small, cheap, flexible and extensively used controllers. They can be found more and more in every day life, including applications such as household appliances, medical systems, traffic control systems, motor vehicles, etc. They are increasingly used to control industrial processes and plants, especially at the lowest level of the automation hierarchy, viz. directly with the devices to be supervised, controlled and regulated. Many of their applications are safety critical, such as traffic control systems or chemical plants, or economically critical, where costs of a bug are measured in millions of dollars, e.g for shutting down a factory.
The tasks a PLC has to solve typically include real-time aspects. They are **real-time systems** as in the folk definition [LSW97]: **Real-time systems are those systems in which correctness depends not only on the logical result of the computation but also on the time at which the results are produced.**
Real-time aspects play a role in at least two different levels: one is the direct control of external devices that have to be switched on or off at certain points in time. This kind of requirement is also called

timeliness. The other one concerns, at a much finer time scale, communication between either PLCs, or different programs on one PLC, that run in parallel solving a complex control task in common. Time requirements of this sort concern **simultaneity**.

The hardware of a PLC consists of a CPU that is microprocessor based, a memory, input- and output-points where signals can be received, e.g. from switches and sensors, and sent to actuators, e.g. to motors or valves.



Usually, a PLC is equipped with an operating system that allows to load and run programs and perform self-checks.

Programs for PLCs are developed and compiled on external devices. The main difference to conventional systems is the operation mode: PLC programs are executed in a permanent loop. In each iteration of the loop, commonly called a **scan cycle**, input is read, the program computes a new internal state and output, and the output is updated. In the program timers can be set and read, and the computation may depend on the state of the timers. There exists an upper time bound for each scan cycle, which typically is in the order of milli-seconds and depends on the size of the program. It is this cyclic behaviour that makes PLCs suitable for control tasks and interaction with a continuous environment.

There exist several different programming languages for PLCs. They have been designed with a main emphasis on applicability, each intended for a specific application domain, and based on the background of the control engineers who use them. One example is Ladder Diagrams, which are still being used heavily, and closely resemble the diagrams of traditional relay logic. From the point of view of computer scientists some of these programming languages lag behind several generations of development in programming methodology.

In order to achieve more conformity of the different PLC programming languages the standard IEC 1131-3 [sta93] was developed. However, it still contains semantic ambiguities, and it provides no formal semantics, which makes precise reasoning impossible.

It is expectable that in several application areas PLCs will be replaced by PCs in the future. However, the cyclic operation principle and the use of timers will be maintained. Therefore, considerations about PLCs in fact apply to a larger class of systems, which we call **polling real-time systems**.

2.1.2 Scientific Problem

The goal of the project is to develop methods and tools for the effective validation of industrial PLC applications. More specifically, we want to address the formal verification and systematic testing of the real-time behavior of such applications, and work on the following questions:

1. Identify well-behaved (but still sufficiently expressive and widely used) subsets of the PLC programming languages from the standard [sta93], and provide those with a formal, operational semantics in terms of timed automata [AD94].

2. Use tool sets such as KRONOS [DY95, DOTY96], UPPAAL[BGK+96, BLL+96], and HyTech [AHH93, HH95], which represent the actual state-of-the-art in automatic verification of real-time systems, to analyze the behavior of industrial PLC applications.

3. Tailor existing methods for conformance testing such as [SVD97] to the setting of PLCs, and use this to validate industrial PLC applications.

Testing is often the only practically feasible technique for complex systems, and the only validation technique that addresses the physical realization of a system, and not some abstraction of it. As such it is a necessary ingredient of any validation procedure that intends to assess systems at product level. Verification is of interest because the aforementioned criticality of many PLC applications. For these applications, testing techniques are typically too weak to achieve the desired level of reliability and more rigorous methods must be applied. A prerequisite for applying such validation methods is the development of suitable formal models for PLCs and PLC programs. The development and study of such models is the logical first step of our programme. As the underlying semantic model for our project we intend to use the model of timed automata of Alur and Dill [AD94], which has witnessed a very rapid and successful development over the past years, both theoretically [ACD93, AH93, AH94, AIKY95] and in terms of tools and practical applications [DY95, DOTY96, BGK+96, BLL+96].

2.1.3 Approach

Case Studies as Primary Motivation PLC programming languages, notations and methods have been developed mainly by control engineers, quite independent of research in computer science. We think it is very important for our project to obtain a clear picture of how exactly control engineers develop PLC applications in practice, and what problems they face when they want to validate these applications. Our aim will then be to help the control engineers with solving these problems. At this moment we have, amongst others, the following three rich sources for industrial case studies:

1. **Rijkswaterstaat**, part of the Dutch Ministery for Watermanagement and Public Works is responsible for watermanagement (quality and quantity), motorway infrastructure, motorway traffic management and traffic safety. For more information we refer to URL `http://www.minvenw.nl/`. Rijkswaterstaat has a large installed base of measurement and control equipment. For monitoring water purposes this includes: monitoring of waterlevels, rates of flow, chemical and biological composition, temperature and many other parameters. For monitoring traffic this includes: traffic intensities, speeds, weather conditions, vehicle heights (tunnel protection!), vehicle weights, etc. For water control purposes, we can mention lock and sluice control, flood-control dams, surge barriers, etc. For traffic control, we can mention object management (movable bridges, tunnels, barriers), traffic signals, road lighting control, etc. This type of equipment usually contains many PLCs. The control or monitoring task is usually too complicated to be dealt with by means of simple electronics, but too simple for full fledged computer equipment, it is usually open air equipment (at least not office conditions) and there is a need for remote access and control, requirements that can be fulfilled with today's PLCs. So Rijkswaterstaat is clearly a large scale consumer of PLC-controlled equipment and is therefore interested in methodologies that suit the purpose of building more powerful and more reliable PLC-based equipment.

2. **VHS project** The University of Nijmegen is one of the partners in the Esprit project 26270 Verification of Hybrid Systems (VHS), which has started in May 1998. Other partners in the project include Verimag, Dortmund, BRICS, Kiel, the Weizmann Institute, CWI, Ghent, and the industrial partners Sidmar, Nylstar and Krupp. The main activity of the project consists of performing case studies on the control of industrial processes, primarily in the chemical industry. The project will build formal models of plants using, amongst others, timed and hybrid automata. It intends to analyze the behavior of these plants using both simulation tools and tools for (semi-)automatic verification that have been built for the timed and hybrid automata models (the tools UPPAAL and KRONOS have been developed by partners BRICS and Verimag, respectively). Specific case studies that will be addressed in the VHS project include

   - an experimental batch plant developed at the Dept. of Chemical Engineering of the University of Dortmund,

- a polymerization reactor, also developed at the University of Dortmund,
- the controller design for an SMB process at Merck,
- steel production at Sidmar, and
- yarn production at Nylstar.

In all of these case studies PLC controllers play a vital role. The VHS project and the proposed project are complementary in that VHS concentrates mostly on hybrid aspects and the analysis of abstract models, whereas this project emphasizes semantics and analysis of PLC programs.

3. **UniForM project** We have an ongoing cooperation with the University of Oldenburg, which is one of the partners within the German UniForM-project [KBPO+96]. One of the goals of this project is to validate PLC software that has been developed by one of the partners, the Firma Elpro LET GmbH (Prof.Dr. D. Balzer and A. Baer), for a tram guidance system in Berlin. As a first result of this collaboration, we have translated the PLC-Automata programming language that has been developed within UniForm to the model of timed automata, see [DFMV98]. As a next step we plan to tackle the verification challenges that arise in the tram guidance system.

Semantics and Mathematical Models Thus far very little work has been done on the formal semantics of PLC programming languages. The semantics of the languages in the standard [sta93] is only defined informally through examples. From a negative perspective, the standard is basically just a huge bag of features that have been grouped together following demands from various interest groups. Many of these features are extremely complex and the standard is ambiguous about their exact meaning. It is generally considered "dangerous" to combine certain features within a single program. In practice, control engineers that have to produce PLC software for safety-critical applications always use well-identified, restricted subsets of the languages from the standard. Clearly, it makes no sense to aim for a completely formal semantics of the full PLC standard. Rather, we want to identify subsets of the standard that are widely used and see for which features in these subsets a simple, formal semantics can be given (as a rule of thumb, we mistrust features with a complex formal semantics).
Important initial research in this direction has been carried out at the University of Edinburgh [AT97, AT98, Tou96, Tou97]. However, this work does not address real-time issues. In recent joint work with the University of Oldenburg, [DFMV98], we do address real-time issues for a very simple language of PLC-Automata (not part of the standard). Currently we are trying to generalize these results to the richer languages from the standard [sta93].
We expect that it will not be sufficient to have just a single semantic model for any given PLC language. The reason is that, depending on the application, we will need to describe the relationships between the input and output signals induced by PLC programs at different levels of granularity.

Verification We take a pragmatic approach to verification: we do not intend to develop completely new theories or to develop new tools, but rather plan to use as much as possible existing theories and verification tools. We will only construct interfaces and front ends that implement the semantic translations from PLC programs to (for instance) timed automata in order to be able to use these verification tools for PLC applications.
As mentioned above, timed automata constitute one of the most successful models for the description of real-time systems. In a timed automaton the discrete part of a system is modeled by a "classical" finite automaton. Timing behavior can be specified by the introduction a set of (real-valued) clocks variables. Conditions on clock values may restrict the occurrence of state transitions in time, and clocks may be reset as the result of performing a transition. The success of the timed automaton model relies on the fact that although at the semantic level the state space of systems is infinite, analysis can be based on a finitary model (the so-called **region graph**), provided that the clock conditions are restricted to a simple linear form. This requirement is met for a very large class of systems,

thus allowing effective analysis on the basis of this model. Based on this finite representation result of [AD94], a rich literature of verification approaches for timed automata has come into existence [Alu91, AFH91, ACH+92, ACD93, AH93, ACH94, AH94, AIKY95, SV96]. Basically, verification tries to demonstrate that a formal model satisfies its **specification**. Such a specification may either be another (usually simpler) formal model, in which case verification consists of proving that both systems have a suitably related observable behaviour. Alternatively (and for timed automata this is the more popular approach thus far), a specification can be a set of properties, expressed as formulae in a suitable logic, and verification then consists of the proof that the formulae of the specification hold for the model.
Although verification is conducted on the basis of finite mathematical structures, they are so large even for relatively small systems (the **state explosion problem**), that they can only be evaluated with the aid of software tools. The current state-of-the-art verification tools for timed systems are KRONOS [DY95, DOTY96], UPPAAL[BGK+96, BLL+96], and HyTech [AHH93, HH95]. Progress in this area is very impressive, and fundamental research combined with ever faster machines and more and cheaper memory enables the (semi-)automated analysis of ever greater systems. Nevertheless, the use of timed automata tools for the analysis of industrial PLC programs will be a great challenge and require much of our ingenuity: the problem is that straightforward translation of PLC programs will lead to timed automata that are too complex for the tools to handle. We plan to tackle this problem by a combination of (amongst others) the following ideas:

1. The timed automata that arise from PLC programs will have a specific structure, which we may be able to exploit during verification. For instance, in order to specify timing constraints PLC programming languages typically allow one to set a timer, which then counts down and generates an event at time 0 ("egg-timer"). This type of timers is strictly less expressive than the clocks that are used in timed automata, and we hope that some of the verification procedures can be made more efficient for the induced subclass of timed automata.

2. In practical cases control engineers will primarily be interested in analysis of certain restricted properties of PLC programs, for instance simple response properties, that no important signals are "missed", simple reachability analysis, etc. For these specific, application dependent properties we may again be able to come up with more efficient verification procedures.

3. When programming PLCs, control engineers often use **Sequential Function Charts (SFCs)** to describe the overall control structure and timing behavior of the program, and a logic or dataflow oriented language (**Relay Ladder Logic (RRL)** or **Function Blocks (FB)**) to describe the input/output relationships within a given the control locations [sta93]: depending on the occurrence of input signals and timeouts, the program switches between the modes of the SFC, whereas the behavior for each individual mode is governed by RLL diagrams or FBs.

   If within a PLC program a clear distinction is made between the control/timing level and the logic/dataflow level, then we may want to use timed automata tools at first only for analysis of the control/timing level.

4. For analysis of RLL diagrams and FBs, propositional logic theorem provers such as [Bry86, Gro94, Stå89, Stå91] appear to be appropriate tools, see [SS90, GKV95b, GKV95b, FH98]. Using a clever combination of timed automata tools and propositional logic theorem provers, we may be able to fully analyze PLC programs containing both SFCs and RLLs/FBs.

Within the context of the Esprit project VHS we collaborate closely with the BRICS group of prof. Kim Larsen in Aalborg (one of the developers of UPPAAL) and the Verimag group of prof. Joseph Sifakis in Grenoble (the developers of KRONOS). This means that in the case that our results allow for more efficient special case algorithms, we have a direct connection to leading tool experts, who are interested in precisely this kind of application.

Testing Black-box testing is the approach to testing which relies on the specification of the system which is being tested to derive test cases. It permits to define the notion of a *conformance* relation linking the system to the specification, and the notion of a *verdict* associated to the application of a test case. The last two decades have witnessed a lot of research activity in the area of (black-box) conformance testing. Especially for the class of finite state models, many (both complete and incomplete) test generation algorithms have been devised, which have been used successfully for the validation of hardware circuits and communication protocols [Koh78, Hol91, ADLU91, TKB92, Tre92, Tre96].
Recently, testing theory for timed systems based on related models has become a subject of study [CL97, MMM95, SVD97]. The basic ingredients for testing PLCs, therefore, are available. This makes time right for further development of these test methods for PLC-related validation, both from the scientific and application point of view. Within our project we want to invest serious effort to see whether currently available methods and tools for test generation, such as the Phact toolset [MRS+97] which is being developed at Philips, can be applied in the setting of PLCs.

Other approaches It is only recently that verification of PLCs got a research subject in formal methods of computer science.
One basic approach is to verify the functional behaviour of PLC programs. One of the first results can be found in [HJKS93], where so-called function blocks, one of the main structural elements of PLC programs, are verified with several different methods: with Dijkstra's weakest condition approach, with Hoare style proofs, and higher order logic theorem proving. In [KV97] the authors do verification based on higher order logic theorem proving and make use of the Isabelle/HOL system. The approach in [AT98] is to reconstruct the language of function blocks providing it with a formal semantics. Having a formal semantics gives the basis for formal reasoning on programs.
The argument for verifying only the functional behaviour of a program is the cyclic operation mode of PLCs. It is possible to give an upper time bound for each cycle of the PLC. Counting cycles and adding up upper time bounds gives a time bound that is sufficient for several properties to show. For many applications, e.g. process control, timing is vital. For dealing with those applications time should be an integral part of the model allowing direct reasoning about time.
Closer to our interest is the approach in the UniForM project of Oldenburg. There, a formal model called "PLC-automata", was developed and equipped with a logical semantics [Die97a]. The logic used is the Duration Calculus, which allows to express explicitly real time behaviour. There exists a synthesis algorithm mapping a specification in the Duration Calculus to a PLC-automaton [Die97b], and further a transformation from PLC-automata to PLC programs that can be directly implemented on a PLC [Die97a]. Additionally, there was a simulator for PLC-automata built [Tap97] which also is useful to increase reliability of PLCs. However, Duration Calculus, which underlies the activities in the UniForM project, is not suitable for automatic verification and testing issues, which are in the focus of our interest.
As a final piece of related work we mention [AFS98], where constraint-based program analysis techniques are used to detect races in relay ladder logic programs. The method proposed in this paper is restricted to finding one kind of bugs, but is clearly effective in doing this.

2.1.4 Expected Results

First result should be an **operational semantics** for PLCs and their programs that addresses timing aspects. Such an operational semantics is a necessary prerequisite for further results: It allows to detect ambiguities and possible inconsistencies of the languages in the standard IEC 1131-3. Based on this well-behaved subsets of the languages in the standard can be identified. Moreover, an operational semantics provides the bridge between PLCs on one hand, and verification tools and testing theory on the other hand.

A further result should be a number of serious **case studies**, where verification of relevant properties is successfully carried through.
Altogether, both a proper operational semantics and the case studies should indicate some guidelines for PLC programs, which contain restrictions of language elements that may be used and require a specified level of programming discipline. For programs according to the guidelines we want to offer the possibilities of **applying verification tools** (UPPAAL, KRONOS and HyTech) and/or **testing theory**.
Making use of existing verification tools includes the implementation of front ends or interfaces that fit to our applications, if necessary. What we do **not** intend is to build tools ourselves.

2.1.5 Scientific Relevance

Volume 1165 Lecture Notes in Computer Science [JRA96] is fully devoted to a "competition" between formal methods: all the authors were asked to apply their favourite formalism to specify and program a simple controller for a steam boiler. In the introduction, the editors compare and evaluate the different solutions and arrive at the conclusion that all methods either fail short on the formal analysis side or on the implementation side: papers that present a convincing mathematical development of the controller fail to make the last step towards physical implementation, and paper which describe a nice implementation fail to adequately validate the proposed solution through the application of formal methods. Clearly, a major challenge for the formal methods community is to bridge the gap between the application of formal methods for validation and the corresponding implementations. In general this is a very difficult problem, but we believe that for the simple hardware architecture of PLCs and for relatively simple applications such as the steam boiler controller, it will soon be possible to bridge the gap. If successful, our project will greatly contribute to this.
At a more technical/specific level, we think that this project is scientifically interesting because it addresses

1. real-time semantics of programming languages for reactive systems,
2. automatic verification of real-time systems,
3. testing of real-time systems.

Recently, quite a number of papers have appeared which indicate that, both because of the great societal relevance and because of the scientific importance of the area, correctness of PLC systems is quickly becoming a hot research topic [AFS98, AT97, AT98, Die97a, Die97b, DT98, DFMV98, HJKS93, KV97, Tou96, Tou97].

2.2 Societal Usefulness

The enormous range of applications of PLCs is obvious (see sections 2.1.1 and 2.1.3). Proper functioning of PLCs is an issue, for both human safety and economical reasons. For examples of the first just think of the use of PLCs for controlling high speed trains, railway tracks, chemical batch plants, micro waves, traffic lights and elevators; for an example of the second we refer to [AFS98]: the analysis in their paper revealed a (known) bug in a relay ladder program, that originally costed approximately $750,000 in factory down time.

2.3 Collaboration with Subdisciplines in Computer Science

PLCs and PLC programming languages have been developed by control engineers. So far they have been studied mainly in engineering departments. Essential for getting a good insight in state-of-the-art PLC technology are therefore close contacts with relevant groups in engineering departments. Within the VHS project we collaborate with the chemical engineering group of prof. Engell of the University of Dortmund, which has extensive experience with PLCs. Additional, we have regular contacts with the mechanical engineering group of prof. Rooda of the Technical University of Eindhoven.

2.4 National and International Initiatives or Activities

Apart from the already mentioned projects VHS and UniForm, our proposal is related to the following activities:

- DAAD scholarship: A. Mader is presently getting a one-year grant from the Deutsche Akademische Austauschdienst, lasting from 5/98 till 4/99. The topic of the project is validation of PLCs and work done within this year should be seen as preparation for the proposed project.
- SION project 612-14-004 "Stepwise Refinement of Hybrid Systems", April 1997 - April 2001. Project of Frits Vaandrager, supporting one PhD student (Ansgar Fehnker).
- SION project 612-33-006 "Testing and Verification of Timed Systems", June 1995 – May 1999. Project of Frits Vaandrager together with prof. Ed Brinksma, University of Twente. This project supported one postdoc at the KUN (Jan Springintveld), and still supports one PhD student in Twente (Pedro D'Argenio).

  The current proposal directly builds on the results of this SION project: it aims to apply and further develop the validation techniques for timed systems in the specific setting of PLCs.
- SION project Classifying Proof Techniques for Propositional Logic at CWI.

  Close contacts exist with the group of dr. Jan Friso Groote at CWI. We want to link the the theorem proving tools that are developed in this group ("Heerhugo") to the verification tools for timed automata that are central in our project.
- Our group is part of the Dutch graduate school IPA, the Institute for Programming Research and Algorithmics. Our project proposal perfectly fits into the area of Embedded Systems, which has been identified as one of the research priorities of IPA.

2.5 Plans for Implementation

2.5.1 Activity Planning

Preceding to the proposed project we work on semantics of PLC programs and timed automaton models. Starting from the results there we plan to proceed in a cyclic way, where we pass through the cycle at least once each year. First, we want to do a **case study**, followed by a **validation phase** and, finally, the development or adjustment of **formal models**.
During the first two years the project will run in parallel with the VHS project. According to the close cooperation with the VHS project and the flow of results in both directions we concentrate in the first part of the project on semantics and verification. When the VHS project has terminated we start to address the testing issue.
In general, it is very difficult to give a detailed planning of our activities. This is due to the fact that we want to do industrial case studies. The precise specifications of these case studies will only be available when starting. This makes an assessment of their precise complexity impossible to predict.
We arrive at the following global planning for our project:
Year 1: case study; semantics and verification
Year 2: case study; verification and testing
Year 3: case study; verification and testing

2.5.2 Personnel Planning

Angelika Mader will work full time on the project for 3 years, supported by SION.
Additionally, Frits Vaandrager, Hanno Wupper, Ansgar Fehnker (SION project on Hybrid Systems) and N.N. (PhD student on the VHS project) will actively participate in the project.

2.5.3 Use of other Facilities

None.

2.5.4 Financial Planning

Standard NWO planning for a three-year post-doc position.

2.5.5 Output

We intend to publish our results in technical reports, conference proceedings and journals. We also would like to organize workshops on that topic bringing together researchers and people from industry who have interest in increasing the reliability of PLC applications.

2.5.6 Collaboration

As described in section 2.1.3 we will cooperate with Rijkswaterstaat, the VHS project and UniForm, who all will provide us with case studies and we will benefit from their knowledge about PLCs. In addition, in the VHS project much expertise on verification tools is available.

2.6 Available Expertise

Angelika Mader

Academic qualifications:
MSc in Computer Science with Minor Mathematics, Technische Universität München, 1990; PhD in Computer Science, Technische Universität München, 1997;

Employment:
After her PhD she spent a year as post-doc at the KU Nijmegen in the group of prof. Vaandrager, supported by the HCM network EXPRESS. At the moment, she is getting a one-year grant from the Deutscher Akademischer Austauschdienst (DAAD) for validation of PLCs at the KU Nijmegen.
She was participating in several projects: *Verifikation paralleler Prozesse* with Fa. Siemens, ZFE; *Werkzeuge und Methoden für die Nutzung paralleler Rechnerarchitekturen*, Sonderforschungsbereich 342, Deutsche Forschungsgemeinschaft; *Theory and Application of Causal Fix-Point Logics* with the University of Edinburgh.

Research interests: Computer-aided verification, modal and temporal logics, concurrency, real-time systems, embedded systems, testing, transfer of formal methods to real-world applications.

Frits Vaandrager

Academic qualifications:
MsC in Mathematics and Computer Science, University of Leiden, 1985; PhD in Mathematics and Computer Science, University of Amsterdam, 1990.

Employment:
After his PhD studies he spent one year as a post-doc at the Massachusetts Institute of Technology (MIT) in the Distributed Systems Group of Prof. Nancy Lynch, and six months as "Maître de

Recherche Associé" in the Meije group of Dr. Gérard Berry at the Ecole des Mines in Sophia-Antipolis. From 1992 until 1995 he led the group "Concurrency and real-time systems" at the CWI in Amsterdam, and held in addition a part-time position as associate professor at the University of Amsterdam. In December 1995 he became full professor at the University of Nijmegen, where he currently leads the research area Informatics for Technical Applications within the Computing Science Institute.
Frits Vaandrager has been (is) involved in nine European projects, three SION projects, and one project funded by Philips. He frequently serves on program committees of international conferences on semantics, concurrency theory and computer aided-verification, for instance CAV'97, TACAS'98, PROCOMET'98 and ICALP'98. He is member of the IFIP Working Group 2.2 on the Formal Description of Programming Concepts.

Research interests: Theory and application of formal methods for software development for technical applications; specification and verification of concurrent, real-time and hybrid systems, computer-aided verification using model checkers and proof tools, temporal and modal logics, operational semantics.

Hanno Wupper

Academic qualifications:
MsC in Mathematics and Computer Science, University of Bochum, D, 1974; PhD in Mathematics and Computer Science, University of Oldenburg, D, 1986.

Employment:
After his graduation he was responsible for system development and maintenance at the computer centre of the University of Bochum, where he developed one of the first completely machine- and device independent graphical programming systems, which has been used at several German university computer centres until the breakthrough of GKS. He worked in DIN and ISO on the standardization of programming languages. After his PhD he has been working as assistant and associate professor at the University of Nijmegen since 1986, where he was involved in an Esprit BRA and in two SION and STW-projects and since 1992 in a number of innovative projects at the School for Informatics of the University of Nijmegen.

Research interests: Formal methods for development of technical applications using and proof tools; specification and verification methodology for embedded systems, unification of concepts from different disciplines.

3 Publications

3.1 Publications of the Applicants

[BPV94] D.J.B. Bosscher, I. Polak, and F.W. Vaandrager. Verification of an audio control protocol. In 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 (FTRTFT'94)*, Lübeck, Germany, September 1994, volume 863 of *Lecture Notes in Computer Science*, pages 170–192. Springer-Verlag, 1994.

[BEM96] J. Bradfield, J. Esparza, and A. Mader. An effective tableau system for the linear time mu-calculus. In *Proceedings of ICALP '96*, volume 1099 of *LNCS*, pages 98–109. Springer, 1996.

[DFMV98] H. Dierks, A. Fehnker, A. Mader, and F.W. Vaandrager. Operational and logical semantics for polling real-time systems. Technical Report CSI-R9813, Computing Science Institute, University of Nijmegen, April 1998. To appear in *Proceedings FTRTFT'98*.

[DGRV97] M.C.A. Devillers, W.O.D. Griffioen, J.M.T Romijn, and F.W. Vaandrager. Verification of a leader election protocol — formal methods applied to IEEE 1394. Technical Report CSI-R9728, Computing Science Institute, University of Nijmegen, December 1997. Submitted.

[DNV95] R. De Nicola and F.W. Vaandrager. Three logics for branching bisimulation. *Journal of the ACM*, 42(2):458–487, March 1995.

[GV90] J.F. Groote and F.W. Vaandrager. An efficient algorithm for branching bisimulation and stuttering equivalence. In M. Paterson, editor, *Proceedings* 17^{th} *ICALP,* Warwick, volume 443 of *Lecture Notes in Computer Science*, pages 626–638. Springer-Verlag, July 1990.

[GV98] W.O.D. Griffioen and F.W. Vaandrager. Normed simulations. Technical Report CSI-R97xx, Computing Science Institute, University of Nijmegen, 1998. To appear in *Proceedings CAV'98.*

[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.

[HV96] D.K. Hammer and F.W. Vaandrager. Ingebouwde systemen nog verre van feilloos. *Automatisering Gids*, 17, April 1996.

[LSVW96] N.A. Lynch, R. Segala, F.W. Vaandrager, and H.B. Weinberg. Hybrid I/O automata. In Alur et al. [AHS96], pages 496–510.

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

[LV96a] N.A. Lynch and F.W. Vaandrager. Action transducers and timed automata. *Formal Aspects of Computing*, 8(5):499–538, 1996.

[LV96b] N.A. Lynch and F.W. Vaandrager. Forward and backward simulations, II: Timing-based systems. *Information and Computation*, 128(1):1–25, July 1996.

[Mad95] A. Mader. Modal μ-calculus, model checking and Gauß elimination. In *Proceedings of TACAS'95*, volume 1019 of *Lecture Notes in Computer Science*, pages 72–88. Springer, 1995.

[Mad97] A. Mader. *Verification of Modal Properties Using Boolean Equation Systems.* Bertz Verlag, Berlin, October 1997. edition versal 8, Dissertation (TU München).

[RV96] J.M.T. Romijn and F.W. Vaandrager. A note on fairness in I/O automata. *Information Processing Letters*, 59(5):245–250, 1996.

[RV98] G. Rozenberg and F.W. Vaandrager, editors. *Embedded Systems*, volume xxx of *Lecture Notes in Computer Science*. Springer-Verlag, 1998.

[SV96] J.G. Springintveld and F.W. Vaandrager. Minimizable timed automata. In B. Jonsson and J. Parrow, editors, *Proceedings of the Fourth International Symposium on Formal Techniques in Real Time and Fault Tolerant Systems (FTRTFT'96),* Uppsala, Sweden, volume 1135 of *Lecture Notes in Computer Science*, pages 130–147. Springer-Verlag, 1996.

[SVD97] J.G. Springintveld, F.W. Vaandrager, and P.R. D'Argenio. Testing timed automata. Technical Report CSI-R9712, Computing Science Institute, University of Nijmegen, August 1997. Submitted to *Theoretical Computer Science.*

[Vaa90] F.W. Vaandrager. *Algebraic Techniques for Concurrency and Their Application.* PhD thesis, University of Amsterdam, February 1990.

[Vaa95] F.W. Vaandrager. Verification of a distributed summation algorithm. In I. Lee and S.A. Smolka, editors, *Proceedings CONCUR 95,* Philadelphia, PA, USA, volume 962 of *Lecture Notes in Computer Science*, pages 190–203. Springer-Verlag, 1995. A full version is available as CWI Report CS-R9505, Amsterdam, January 1995.

[WV89] H. Wupper and J. Vytopil. A Systems Specification Language treating Time and Reliability as "First Class Citizens". In: J. Zalewski (ed.), *Proc. IFIP/IFAC Conf. on Hardware and Software for Real-Time Process Control*, Warsaw. Elsevier, Amsterdam, 1989.

[WM97a] H. Wupper and H. Meijer. Towards a Taxanomy for Computer Science. In: *Informatics as a discipline and in other disciplines: what is in common?* IFIP WG 3.2 Working Conference, Enschede, August 1997.

[WM97b] H. Wupper and H. Meijer. A Taxanomy for ComputingScience. Technical Report CSI-R9713, Computing Science Institute, University of Nijmegen, August 1997 (also available via http://www.cs.kun.nl/ wupper/taxonomy/taxonomy-frame.html).

[Wup97a] H. Wupper. Design as the Discovery of a Mathematical Theorem: What designers should know about the art of mathematics. Technical Report CSI-R9729 Computing Science Institute, University of Nijmegen, December 1997.

[Wup97b] H. Wupper. Quantified Modal Algebra: Quantified modal operators and their use in specification. Technical Report CSI-R97309 Computing Science Institute, University of Nijmegen, December 1997.

3.2 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+92] R. Alur, C. Courcoubetis, N. Halbwachs, D. Dill, and H. Wong-Toi. Minimization of timed transition systems. In W.R. Cleaveland, editor, *Proceedings CONCUR 92,* Stony Brook, NY, USA, volume 630 of *Lecture Notes in Computer Science*, pages 340–354. Springer-Verlag, 1992.

[ACH94] R. Alur, C. Courcoubetis, and T.A. Henzinger. The observational power of clocks. In B. Jonsson and J. Parrow, editors, *Proceedings CONCUR 94,* Uppsala, Sweden, volume 836 of *Lecture Notes in Computer Science*, pages 162–177. Springer-Verlag, 1994.

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

[ADLU91] A.V. Aho, A.T. Dahbura, D. Lee, and M.Ü. Uyar. An optimization technique for protocol conformance test generation based on UIO sequences and Rural Chinese Postman Tours. *IEEE Transactions on Communications*, 39(11):1604–1615, 1991.

[AFH91] R. Alur, T. Feder, and T.A. Henzinger. The benefits of relaxing punctuality. In *Proceedings of the* 10^{th} *Annual ACM Symposium on Principles of Distributed Computing*, pages 139–152, 1991.

[AFS98] A. Aiken, M. Fändrich, and Z. Su. Detecting races in relay ladder logic programs. In B. Steffen, editor, *Proceedings of TACAS'98*, volume 1384 of *Lecture Notes in Computer Science*, pages 184–200. Springer, 1998.

[AH93] R. Alur and T.A. Henzinger. Real-time logics: complexity and expressiveness. *Information and Computation*, 104(1):35–77, 1993.

[AH94] R. Alur and T.A. Henzinger. A really temporal logic. *Journal of the ACM*, 41(1):181–204, 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.A. Henzinger, and E.D. Sontag, editors. *Hybrid Systems III*, volume 1066 of *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.

[Alu91] R. Alur. *Techniques for Automatic Verification of Real-time Systems*. PhD thesis, Dept. of Computer Science, Stanford University, 1991.

[AT97] S. Anderson and K. Tourlas. Diagrams and programming languages for programmable controllers. In *Proceedings of the Formal Methods Europe Symposium*, LNCS, pages 1–19. Springer-Verlag, 1997.

[AT98] S. Anderson and K. Tourlas. Design for proof: An approach to the design of domain specific languages. May 1998.

[BGK+96] J. Bengtsson, W.O.D. Griffioen, K.J. Kristoffersen, K.G. Larsen, F. Larsson, P. Pettersson, and Wang Yi. Verification of an audio protocol with bus collision using UPPAAL. In R. Alur and T.A. Henzinger, editors, *Proceedings of the 8th International Conference on Computer Aided Verification,* New Brunswick, NJ, USA, volume 1102 of *Lecture Notes in Computer Science*, pages 244–256. Springer-Verlag, July/August 1996.

[BLL+96] J. Bengtsson, K.G. Larsen, F. Larsson, P. Pettersson, and Wang Yi. UPPAAL: a tool suite for the automatic verification of real-time systems. In Alur et al. [AHS96], pages 232–243.

[Bry86] R.E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, August 1986.

[CL97] D. Clarke and I. Lee. Automatic generation of tests for timing constraints from requirements. In *Proceedings of the Third International Workshop on Object-Oriented Real-Time Dependable Systems,* Newport Beach, California, February 1997.

[Die97a] H. Dierks. PLC-automata: A new class of implementable real-time automata. In M. Bertran and T. Rus, editors, *Transformation-Based Reactive Systems Development (ARTS'97)*, volume 1231 of *Lecture Notes in Computer Science*, pages 111–125. Springer-Verlag, 1997.

[Die97b] H. Dierks. Synthesising controllers from real-time specifications. In *Proceedings of Tenth International Symposium on System Synthesis*, pages 126–133. IEEE CS Press, 1997.

[DOTY96] C. Daws, A. Olivero, S. Tripakis, and S. Yovine. The tool KRONOS. In Alur et al. [AHS96], pages 208–219.

[DT98] H. Dierks and J. Tapken. Tool-supported hierarchical design of distributed real-time systems. In *Proceedings of EuroMicro 98*, 1998.

[DY95] C. Daws and S. Yovine. Two examples of verification of multirate timed automata with KRONOS. In *Proceedings of the 16th IEEE Real-Time Systems Symposium (RTSS'95)*, Pisa, Italy, pages 66–75. IEEE Computer Society Press, December 1995.

[FH98] W.J. Fokkink and P.R. Hollingshead. Verification of interlockings: from control tables to ladder logic diagrams. In J.F. Groote, S.P. Luttik, and J.J. van Wamel, editors, *Proceedings 3rd Workshop on Formal Methods for Industrial Critical Systems (FMICS'98)*, pages 171–185. Stichting Mathematisch Centrum, 1998.

[GKV95a] J.F. Groote, J.W.C. Koorn, and S.F.M. van Vlijmen. Formele analyse van het veiligheidssysteem op het station van Hoorn-Kersenboogerd. *Informatie*, 36(6):397–404, 1995.

[GKV95b] J.F. Groote, J.W.C. Koorn, and S.F.M. van Vlijmen. The safety guaranteeing system at station Hoorn-Kersenboogerd (extended abstract). In *Proceedings 10th Annual Conference on Computer Assurance (COMPASS'95)*, Gaithersburg, Maryland, pages 57–68, 1995.

[Gro94] J.F. Groote. Hiding propositional constants in BDDs. Logic Group Preprint Series 120, Utrecht University, 1994. Available by ftp from `ftp.phil.ruu.nl` as `logic/PREPRINTS/preprint120.ps.Z`.

[HH95] 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.

[HJKS93] W.A. Halang, S.-K. Jung, B.J. Krämer, and J.J. Scheepstra. *A Safety Licensable Computing Architecture*. World Scientific, 1993.

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

[JRA96] H. Langmaack J.-R. Abrial, E. Boerger, editor. *Formal Methods for Industrial Applications (Specifying and Programming the Steam Boiler Control)*, volume 1165 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.

[JT97] K.-H. John and M. Tiegelkamp. *SPS-Programmierung mit IEC 1131-3*. Springer, 1997.

[KBPO+96] B. Krieg-Brückner, J. Peleska, E.-R. Olderog, D. Balzer, and A. Baer. UniForM — Universal Formal Methods Workbench. In U. Grote and G. Wolf, editors, *Statusseminar des BMBF Softwaretechnologie*, pages 357–378. BMBF, Berlin, March 1996.

[Koh78] Z. Kohavi. *Switching and Finite Automata Theory*. McGraw-Hill, Inc., Second edition, 1978.

[KV97] B.J. Krämer and N. Völker. A highly dependable computer architecture for safety-critical control applications. *Real-Time Systems Journal*, 13(3):237–251, 1997.

[LSW97] K.G. Larsen, B. Steffen, and C. Weise. Continuous modeling of real-time and hybrid systems: from concepts to tools. *International Journal on Software Tools for Technology Transfer*, 1(1):64–85, 1997.

[MMM95] D. Mandrioli, S. Morasca, and A. Morzenti. Generating test cases for real-time systems from logic specifications. *ACM Transactions on Computer Systems*, 13(4):365–398, 1995.

[MRS+97] J.R. Moonen, J.M.T. Romijn, O. Sies, J.G. Springintveld, L.M.G. Feijs, and R.L.C. Koymans. A two-level approach to automated conformance testing of vhdl designs. *Testing of Communicating Systems*, pages 432–447, 1997.

[Roh96] P. Rohner. *Automation with Programmable Logic Controllers.* Macmillan, 1996.

[SS90] G. Stålmarck and M. Säflund. Modelling and verifying systems and software in propositional logic. In *Proceedings 9th Conference on Computer Safety, Reliability and Security (SAFECOMP'90)*, pages 31–36. Pergamon Press, 1990.

[Stå89] G. Stålmarck. A note on the computational complexity of the pure classical implication calculus. *Information Processing Letters*, 31:277–278, 1989.

[Stå91] G. Stålmarck. Normalization theorems for full first order classical natural deduction. *Journal of Symbolic Logic*, 56:129–149, 1991.

[sta93] *IEC International Standard 1131-3, Programmable Controllers, Part 3, Programming Languages*, 1993.

[Tap97] J. Tapken. Interactive and compilative simulation of plc-automata. In Winfried Hahn and Axel Lehmann, editors, *Proceedings of Simulation in Industry ESS'97*, pages 552–556. Society for Computer Simulation, 1997.

[TKB92] J. Tretmans, P. Kars, and E. Brinksma. Protocol conformance testing : A formal perspective on ISO IS-9646. In J. Kroon, R. J. Heijink, and E. Brinksma, editors, *Fourth International Workshop on Protocol Test Systems*, number C-3 in IFIP Transactions, pages 131–142. North-Holland, 1992. Extended abstract of Memorandum INF-91-32, University of Twente, The Netherlands, 1991.

[Tou96] K. Tourlas. Semantic analysis and design of languages for programmable logic controllers. Master's thesis, Department of Computer Science, The University of Edinburgh, 1996.

[Tou97] K. Tourlas. An assessement of the IEC 1131-3 standard on languages for programmable controllers. In Peter Daniel, editor, *SafeComp'97*, pages 210–219, 1997.

[Tre92] J. Tretmans. *A Formal Approach to Conformance Testing.* PhD thesis, University of Twente, December 1992.

[Tre96] J. Tretmans. Test generation with inputs, outputs, and repetitive quiescence. *Software–Concepts and Tools*, 17:103–120, 1996.