form

info@form-ict.nl
www.form-ict.nl
Ir. Frank Stappers & **Ir. Sjoerd Cranen**
+31(0)6 2151 0775
Maaslandlaan 144
6004GH Weert
KvK 55171214

Form offers analytic methods that can be integrated into development trajectories to optimise system and business processes. These tailor made solutions consist of a combination of formal and model based techniques, that assist engineers in designing systems correctly.

The analytic methods are applicable for high-tech enterprises that manufacture embedded systems or develop mission critical applications. Form offers product quality improvement, lower lead time of development and reduces costs for testing by verifying designs first.

Our services

We offer an impact analysis that explains where in your organisation the use of formal methods is beneficial. We assist you by integrating formal methods into your current work flow, so your business can enjoy the benefits with minimal adaptation. If applicable to your situation, we provide domain specific language verification, making the languages currently in use by your company even more productive.

If you are having trouble with a system under test or a deployed system, or if you simply want to have more guarantees about the behaviour of such a system, then our system verification service can provide you with the information you need. Are you planning to embark on a large programming effort? Design verification can save a lot of time by reducing testing effort and minimising rework.

You...

- Want to be one step ahead of the market by guaranteeing correct behaviour of your designs.
- Employ model driven system engineering.
- Integrate formal test and verification methodology into your existing system development.
- Want to reduce the cost of software development and testing.
- Use your favourite set of tools and extend them with verification functionality.

We...

- Analyse complex systems and prove correctness of software designs.
- Offer user friendly verification, tailored to seamlessly integrate into existing development processes.
- Apply formal verification to existing domain-specific models.
- Aspire solutions that guarantee freedom, no vendor lock-in, and no exclusive maintenance contracts.
- Design distributed systems using a model-based approach.

> ‘More companies should offer formal methods in a commercial setting.’
>
> prof.dr.ir. J.C.M. Baeten
> in Bits & Chips (April 2011)

Manifest

We believe that model based verification techniques are ready to be applied on an industrial scale.

Because the verification of a system inherently involves describing its essence, we do not believe that all systems can be verified using a single tool. Any such tool would have to be able to capture the meaning of any domain, thus confusing the engineer with too many parameters, or depriving him of the possibility to correctly describe the system at an appropriate level of abstraction.

Verification significantly improves the quality of software and increases the speed of development.

In our view, verification procedures for complex systems need to be tailored to the requirements of an engineer. These procedures should be integrated into model-driven engineering development trajectories, using domain specific languages as a primary means to achieve this.

We believe that industry is ready to embrace formal methods as a fundamental part of their development process. Hence, we assist industry to decide on the scale and form in which verification methods are applied.

Our Solution

Form leverages techniques that have been developed in academia for the last few decades to bring industrial software quality to a higher level. Combining our backgrounds in both academia and industry, we search for the most beneficial solution, using the most advanced techniques available. As we hold tight connections with academia, we can bring promising new techniques into practice. By integrating them into your existing work flow, we make sure that our solutions are accessible to the people using them. Formal methods have changed the way we look at software; we invite you to enjoy the view from our vantage point.

Formal Methods Projects

Academia carried out many experiments with formal verification. In almost 100% of the case studies, bugs and erroneous behaviour have been found. We highlight a few examples where this technology was successfully applied.

Automated Parking Garage

A design for a garage that automatically stows cars in free spaces and retrieves them on the owner's request included a software controller. As other garages reported a number of severe incidents, many of them related to the behaviour of the software controller, the designers decided that the controller should be verified prior to building the garage.

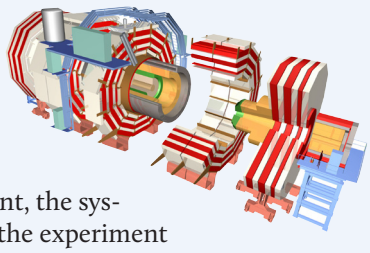
A number of safety requirements were checked on mCRL2 models that modelled several aspects of the controller. The analysis was assisted by a custom created visualization tool. The analysis revealed a number of serious errors, including bisection of cars when exiting or moving them around.

The Modular HEF System

The modular HEF System is a distributed lift system for lifting heavy vehicles, like trucks. Every lever has a microcontroller, that controls and identifies connected levers. The information exchange between the levers is communicated over a CAN bus. Occasionally levers became non-responsive during startup.

By modelling the system in the μCRL language, the behaviour of the system was verified. This uncovered four errors in the original design, for which appropriate solutions were proposed and implemented.

CERN's Large Hadron Collider



CERNS's Large Hadron Collider is used to study a wide range of particles and phenomena produced in the high-energy collisions. The hierarchical composition of the system is described through a domain-specific language, called State Manager Language (SML).

In the Compact Muon Solenoid (CMS) experiment, the system suffered from outages. SML specifications for the experiment have been translated to mCRL2 specifications. Their verification revealed that up-to 5% of all finite state machines in the control software suffered from livelocks, due to which parts of the system became (temporarily) non-responsive, and nearly 11% of the finite state machines suffered from reachability issues.

Frank Stappers, born 24-04-1982 (Weert), obtained his Master's degree Computer Science at the Eindhoven University of Technology (TU/e) in April 2007. Subsequently, he began to work as a Ph.D. student at the Design and Analysis of Systems Group at the computer science department of the TU/e. The first three years he was involved in the project TWINS: Optimizing Software Hardware Co-design Flow for Software Intensive Systems. In 2010 he moved to the Formal Methods Group to carry out the project LythoSysSL at ASML. In 2011 he returned to the university to write his thesis, which he defends in 2012.

Sjoerd Cranen, born 17-12-1984 (Wijchen), graduated cum laude from the Eindhoven University of Technology in 2008 after finishing his Master's thesis in Sheffield, UK. After that, he worked as a software developer for Imtech ICT Technical Systems in Amersfoort, where he was mainly working on the development of orchestration software for the Dutch and German wind tunnels in Marknesse. In November 2009, he returned to the TU/e to do a Ph.D. study, in the HTAS Verified project. This project is a cooperation between several faculties of the TU/e, TNO and various partners in the automotive industry, in which Sjoerd participates as a member of the Model Driven Software Engineering group of the TU/e.