Project 22703: SYRF

**Deliverable 1**

# Case Studies

---

## 1 Schneider Electric Case Study

### 1.1 Overview of the Case Study

The case study proposed by Schneider Electric concerns a Source Range Instrumentation System (SRIS) which is part of a nuclear protection system. Using measurements from neutron detectors, the SRIS controls the emergency shutdown of the reactor and other protection actions to avoid accidental damage. This system operates only when the neutron flow is at a low level (i.e. during the start-up or the shutdown of the reactor).

The SRIS is organised into 2 redundant groups. Each group consists in 3 redundant measurement channels which perform protection requests and one voting system which decides to trigger or not the protection actions. Each channel is periodically tested (test mode). As regards the SYRF project, we focus on one channel.

Schematically, a measurement channel is composed of :

- a neutron detector which measures the neutron flow;
- an electronic rack which processes the signal coming from the detector;
- a pulse inputs board which delivers pulses proportional to the detected neutron flow;
- a software which computes the protection requests, sent to the voting system.

The power supplied by a nuclear reactor is divided into 3 ranges: the source range, the intermediate range, the power range. For each range, specific detectors and processing are required. The low level detector is a proportional counter whose sensitivity allows to count the neutrons multiplication factor of the fuel during the reactor shutdown or start-up. To produce a signal representative of the neutron reaction, the detector is supplied with a polarising high voltage and a discrimination threshold to disregard the reaction due to other particles than neutrons.

The electronic rack provides the detector with the high voltage and the discrimination threshold. To prevent detector life shortening, the high voltage shall be cut-off when the SRIS is not used to carry out the protection actions. To check the detector service life, the operator allows the high voltage value and the discrimination threshold to vary and observe the signal delivered by the detector. This procedure is called the adjusting mode. It provides the operator with data representing the reliability of the neutron measurements.

The pulses input board is an interface between the detector and the software. It delivers a number of pulses representative of the neutron reaction (counter of pulses), and the time needed to count these pulses (counter of time). If the number of pulses slowly increases, the board waits for a maximum duration to deliver the 2 counters. If the number of pulses rapidly increases, the board waits for a minimum duration.

Using data from the pulse inputs board, the software calculates a counting rate which corresponds to the number of pulses over certain time, and a period which corresponds to the change in the neutron population over certain time. Four protection requests are performed by comparing these data to given thresholds:

1. the reactor shutdown if the counting rate is more than a given threshold;
2. the reactor shutdown if the period is less than a given threshold provided that the counting rate is high enough (see request nr.4);
3. the constant rod drop at a given speed if the period is less than a given threshold provided that the counting rate is high enough (see request nr.4);
4. the fourth protection request allows the 2 previous ones to become effective if the counting rate is more than a given threshold.

The software performs tests to detect whether a fault has occurred. The channel goes in fault mode in 9 cases: problems with the pulse inputs board, channel both in test mode and in adjusting mode, high voltage cut-off during adjusting mode are some examples which lead to fault mode. When a fault mode is detected, the CPU is stopped.

To sum up, the functions of the software are:

- calculation of
  - a counting rate corresponding to the number of pulses over certain time;
  - a period corresponding to the change in the neutron population over certain time;
- computing protection requests when one of these data (counting rate or period) go above or below given thresholds;
- detect a fault mode;
- display on a local panel, in the control room and on an auxiliary system;
- producing an audio signal.

The software has 29 inputs. Some of them are listed below; they are classified depending on their origin. Examples of inputs coming from:

- the operator : channel in test mode, channel in adjusting mode, thresholds for protection requests
- the digital signal processing : high voltage value, discrimination threshold
- the pulse inputs board : number of pulses

- the source rack : door closed/opened, ventilation

The software has 50 outputs. Some of them are listed below; they are classified depending on their destination. Examples of outputs sending to:

- the voting logic: protection requests
- the local display panel: counting rate, period
- the audio board: vibration frequency
- the LEDS: channel in test mode, channel in adjusting mode, fault code
- the network: protection requests, counting rate and period, indicators

The software has been developed using the Scade tool. It is made of 36 Lustre nodes and 3 functions in C language.

A detailed description of the case study is given in Annex 1.a. Annex 1.b gives the properties that should be proved, and assumptions about the system environment. Both of these documents are confidential, their access is limited to the partners and the reviewers of the project.

### 1.2 Verification with XLESAR Task(3.1)

Annex 1.c describes the use of the model-checker XLESAR to verify properties about the SRIS software. 19 properties have been automatically proved, generally in less than 1 second per property. Some ambiguities in the specification were detected. Several "verification engines" were tried and compared. The tool has been improved during, and because of this experimentation.

### 1.3 Automatic Testing (Task 3.5)

Deliverable 3.5.2 (2nd year) described the use of the tool Lurette for automatic testing of the SRIS software. The case study was particularly interesting for that, since there are strong assumptions about the system environment, wich must be fulfilled by the selected test sequences. This knowledge about the system environment was described by an assertion observer, which was also strengthenned in various ways to drive the testcase generation towards interesting scenarii. Some desired properties were also specified by an oracle observer. Then the tool was used to generate and run test sequences, while checking the satisfaction of the specified properties.

### 1.4 Reingeniering with Mode-Automata (Task 2)

The SRIS software has been partially rewritten, using "mode-automata", a formalism developped within task 2 ("Combination of Formalisms"), and combining the data-flow style used in Lustre-Scade, with *running modes* described by automata. This experiment is described by Annex 1.d. Moreover, the obtained program was compared with the original one, using both XLESAR (comparison of finite-state components) and Lurette (running test case, and comparing the results).

# 2 SAAB Case Studies

## 2.1 Background

Today's systems are getting more and more complex leading to a demand for more efficient tools and methods in the development process. Both within academia and tool-suppliers there is a constantly ongoing research and development for improving these tools and methods. In an ambition to learn more about the "inside" of these tools - to be better prepared for later including them in our development process - Saab joined the SYRF-project. Also, Saab's interest has been specially devoted to the ability of doing automatic verification, which we have used for the main purpose of doing safety and reliability analysis.

Saab's role in the project was to give research partners access to examples whereby it should be possible to test and demonstrate new ideas. We present a short description of the two examples and a "practical" evaluation of results presented in project.

The two examples from Saab are reactive systems for regulating temperature, airflow and pressure in closed areas. They are different in size, one small and one large system, but they represent the same type of functionality. The small system is a made up "demo" example representing a climatic chamber and the large one is the Environmental Control System (ECS) in the aircraft JAS 39 Gripen. The reason for suggesting two examples is the interesting question - or rather a requirement from the industry - of the ability to deal with large systems, specially when performing automatic verification. The overall functionality of the two systems is described below.

## 2.2 The climatic chamber

The system consists of a heater and a fan placed in a room, which is to be tempered, having one inlet, one outlet and sensors for temperature and airflow. Also, there is a control panel including meters to show the actual temperature and airflow, warning-light and sound and a knob for selecting room temperature. So, the aim is to ventilate the chamber via the inlet and outlet to keep the selected temperature and ventilation within given limits. Since the system functionality is described in detail, for example in [1], [2] and [3], it is not further described here.

## 2.3 The ECS

This is a system in JAS 39 Gripen supplying different areas with tempered air or pressure. The main functions of the system are:

- Cabin air conditioning including ventilated suit
- Cabin pressurization, overpressure and vacuum relief
- Cooling of avionics and ventilation of avionics equipment bays
- Windshield defrosting/defogging
- Pressurization of miscellaneous equipment, for example the fuel tanks.

The main parts of the system consist of a controller, many valves and air pipes, cooling equipment and numerous sensors. System requirements are mainly concerned with temperature and pressure to be regulated within specified limits. Also, many safety related requirements concern the handling of exceptional events, for example what actions should be made when hot air is ventilated into the avionics equipment bay or when pressure is getting low in the fuel tanks.

Note, that the climatic chamber, the "demo" example, includes the same type of functionality and requirements as the ECS. Thereby the acquired techniques for modeling and analyzing are the same, except that it is much smaller.

## 2.4 Analyses and Results

### 2.4.1 Climatic chamber

Many of the partners within the project have used the climatic chamber for demonstrating different aspects. For example in [1] and [2] a mixing of different modeling techniques is outlined and in [3] a study of modeling both the controller and the physical environment is done.

This example was suitable, both in size and complexity, for evaluating different theoretical aspects. The small size made it possible to get a complete overview of its behavior and yet the functional aspects, which should be modeled and verified, were representative also for genuine systems. Due to the extensive documentation of analyses and results of the climatic chamber it is not further discussed here except for some conclusions found in some references:

- The case study was successful for evaluating the use of mode-structures within Lustre programs [2].

- Although relatively small, the experiment we have presented can be considered in several aspects as representative of multi-formalism designs using semantically well-founded languages, such as synchronous languages [1].

- The work serves as an illustration of many steps "in between" going from an informal document and a synchronous program to formal verification of "hybrid" properties [3].

### 2.4.2 Environmental Control System

The controller of the ECS has been analyzed by one partner, Prover Technology. This was made possible since the ECS controller was modeled in the ASA+ format and also an automatic translation capability was at hand for transforming to the NP-Tools format. The reason why we had the ASA+ model and the translator, which we had already before the start of the SYRF-project, was Saab's interest in evaluating new techniques utilizing a model-based development process.

The translation from the ASA+ format to the NP-Tools format resulted in a macro having about 700 in pins and 500 out pins. In this translation only the top level macro is shown with all the logic and arithmetic hidden in the macro. Two of the analyses performed are:

1. The regulation of the ejector valve is controlled by three factors: throttle position, outside pressure and speed. For certain combinations of values of these factors it should be impossible to order the ejector valve to close. This requirement was proven to be correct.

2. Whenever the air temperature in the electronic bay exceeds 50 deg. C the ram air intake shall under all circumstances be activated. This requirement was also satisfied.

Yet another analysis resulted in a possible division by zero, as the ASA+ model was interpreted by NP-Tools, which led to investigations on how to handle this problem in NP-Tools.

### 2.5 Summary and Evaluation

Even though the climatic chamber was a small demo system it reflects much of the functionality included in real systems except for the size. Our interpretation is that the example has been useful for demonstrating both technical problems and new ideas.

Unfortunately the ECS has not been investigated so much due to the problem of transforming the ASA+ format to some format utilized by the project partners, except for Prover. This is of course a drawback since one of the main interests in industry is to investigate the ability of this "formal method approach" to analyze large systems since most industrial systems are large. Anyhow, a few analyses have been performed demonstrating the ability to successfully do automatic verification for this large system.

So, one lesson learned is that it is impossible in this type of project to utilize real systems for testing unless there is a "computer model" at hand from the industrial partner. This is mainly because the time required to do the modeling, starting from a specification in natural language, is enormous compared to the total project budget.

The use of formal methods (FM) in practical applications have been evaluated a number of times during the last ten years, see for example [4] and [5]. From the experiences of participating in the SYRF project, doing some research for utilizing formal methods for reliability analysis and having used NP-Tools at Saab for occasional investigations, in our opinion, the future of "formal method" depends on three areas:

**Tools:** This area includes both the "engine", i.e. the algorithms being able to handle large industrial systems, the formats and the translations between formats and the interface between the user and the tool. Some of the reasons why industry have not yet started to use FM can be found in this area. One difficulty is that of translating between different formats which results in problems both when we want to use two or more design tools or when we want to be able to perform analysis, for example automatic verification, not included in the design tool. Another one is the doubt about being able to handle practically large systems.

**Methodology:** Both from our own experience and from references [4] and [5] it is clear that just giving the engineer access to a tool does not mean he will be able to perform a better and more effective work. Much work needs to be done before a methodology can be presented on how to use the different tools efficiently.

**People:** A third area is that a proper education is needed of the engineers who are going to use these tools and methodologies. One important circumstance to bear in mind is the pitfall of using a too complicated mathematical notation within formal methods. So, in the further development of tools it is desirable to "hide" as much as possible of the complicated mathematics within the tool and trying to have an engineering friendly interface, which would make it easier to educate larger groups of engineers.

The concluding Saab remark of the SYRF project is that it certainly has made contributions, specially to the "Tool" part, since both algorithms, formats and tool-interfaces have been further developed within the project.

# 3 EDF Case Studies

## 3.1 Incremental Design of a Power Transformer Station Controller using a Controller Synthesis Methodology

The Signal language is developed for precise specification of real-time reactive systems. In such systems, requirements are usually checked a posteriori using property verification and/or simulation techniques. Control theory of Discrete Event Systems (DES) allows to use constructive methods, that ensure, a priori, required properties of the system behavior. The validation phase is then reduced to properties that are not guaranteed by the programming process [8].

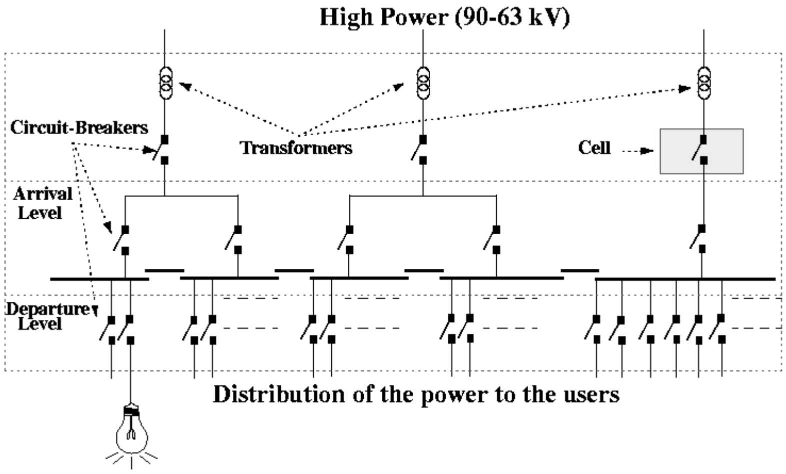


Figure 1: The power transformer station topology.

In Annex 1.e [9], we consider the application of these techniques to the specification of the automatic control system of a power transformer station. It concerns the response to electric faults on the lines traversing it. It involves complex interactions between communicating automata, interruption and preemption behaviors, timers and timeouts, reactivity to external events, among others. The functionality of the controller is to handle the power interruption, the redirection of supply sources, and the re-establishment of the power following an interruption. The objective is twofold: the safety of material and uninterrupted best service. The safety of material can be achieved by (automatic) triggering circuit-breakers when an electric fault occurs on lines, whereas the best quality service can be achieved by minimizing the number of costumers concerned by a power cut, and re-establishment of the current as quickly as possible for the customers hit by the fault (i.e, minimizing the failure in the distribution of power in terms of duration and size of the interrupted sub-network).

## 3.2 The Steam-Boiler Control Problem

Among the various formalisms for the design of reactive systems, the Signal-Coq formal approach, i.e. the combined use of the synchronous dataflow language Signal and the proof assistant Coq, seems to be especially suited and practical. Indeed, the deterministic concurrency implied by the synchronous model on which Signal is founded strongly simplifies the specification and the verification of such systems. Moreover, Coq is not limited to some kind of properties and so, its use enables to disregard what can be checked during the specification stage. In Annex 1.f [6, 7] we underline the various features of this Signal-Coq formal approach with a large scale case study, namely the Steam Boiler problem.

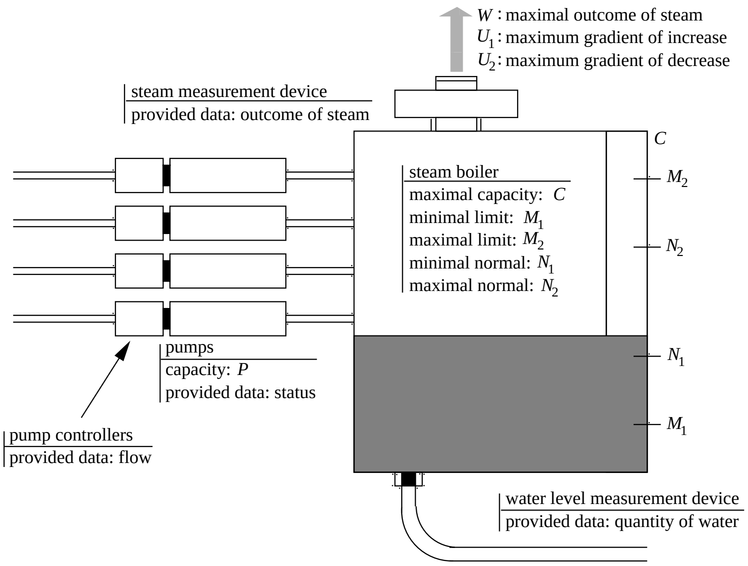


Figure 2: Physical environment

**Physical environment** The physical environment is composed of several units (Fig. 2). Each one is characterized by physical constants and some of them provide data.

**Problem statement** The program has to control the level of water in the steam boiler. This quantity has to be neither too low nor too high. Otherwise, the system might be affected. The program also has to manage physical units failures. For that purpose, at every moment, its takes into account the global state of the physical environment which is denoted by an operation mode. According to this state, the program decides at each cycle if the system must stop or not, and if not, it activates or deactivates pumps in order to keep the level as much as possible in the middle of the steam boiler.

The specification also gives additional information concerning the physical behaviour of the steam boiler. Namely, new values, called adjusted and calculated values, are proposed. They enable to keep the control of the system, managing a vision of its dynamic, when a measurement device is defective.

We chose this case study because it is well adapted to our aim, i.e. to show the interest of the Signal-Coq formal approach. Indeed, the program has to handle several physical parameters and it may use non linear numerical values. Thus, safety properties cannot be proved directly with a classical model checker.

## References

[1] L. Besnard, P. Bournai, T. Gautier, N. Halbwachs, S. Nadjm-Tehrani, A. Ressouche. Design of a multi-formalism application and distribution in a data-flow context: an example. 12th International Symposium on Languages for Intensional Programming, ISLIP' 99, NCSR Demokritos, Athens, Greece, June 1999.

[2] F. Maraninchi, Y. Rémond. The Saab climatic chamber, an experience with mode-automata. Technical Report, Verimag, 1999.

[3] S. Nadjm-Tehrani. Integration of Analog and Discrete Synchronous Design. Second international workshop on Hybrid Systems: Computation and Control, LNCS 1569. March 99.

[4] M. Hinchey and J. Bowen. Applications of Formal Methods. Prentice Hall, 1995.

[5] D. Craigen. An introduction to formal methods, Part III: Industrial usage. ORA Ottawa Canada, 1992.

[6] M. Kerboeuf, D. Nowak, and J.-P. Talpin. The steam-boiler problem in signal-coq. Technical Report 3773, Irisa / Inria-Rennes, October 1999.

[7] M. Kerboeuf, D. Nowak, and J.-P. Talpin. The steam-boiler problem in signal-coq. In *Submitted to TACAS'99*, 1999.

[8] H. Marchand and M. Le Borgne. The supervisory control problem of discrete event systems using polynomial methods. Technical Report 1271, Irisa, October 1999.

[9] H. Marchand and M. Samaan. On the incremental design of a power transformer station controller using controller synthesis methodology. In *World Congress on Formal Methods (FM'99), Volume 1709 of LNCS*, pages 1605–1624, Toulouse, France, October 1999.