nks

Nordisk kerne-sikkerheds-forskning | Nordisk kärn-säkerhets-forskning | Pohjoismainen ydin-turvallisuus-tutkimus | Nordic nuclear safety research

RAK2

NKS/RAK-2(97)TR-B3

CAMS ACHIEVEMENTS IN 1996

P. Fantoni
Y. Iguchi
G. Meyer
A. Sørenssen
C. Van Dycke

Institutt for energiteknikk (IFE)
OECD Halden Reactor Project
Halden, Norway

January 1997

CAMS Achievements in 1996

by

Paolo Fantoni, Yukihiro Iguchi, Geir Meyer, Aimar Sørenssen, Claude Van Dyck

Institutt for energiteknikk (IFE)
OECD Halden Reactor Project
Halden, Norway

January 1997

ABSTRACT

CAMS (Computerized Accident Management Support) is a system that will provide assistance to the staff in the control room, in the technical support centre, and in a national safety centre. These three groups of users do not need the same type of support. Support is offered in identification of the plant state, in assessment of the future development of the accident, and in planning of accident mitigation strategies.

In May 1995, the predictive part of the system was tested at a safety exercise arranged by the Swedish Nuclear Inspectorate, and found to be a useful tool, with potential for further develoment.

Now, new methods are added in signal validation, state identification, tracking simulation, predictive simulation, risk monitoring, and man-machine interface design. A prototype was demonstrated at Loen in May 1996 and at a seminar at Barsebäck in September 1996. This prototype has been further developed during the autumn of 1996. The purpose of this prototype is to test those methods in a simulated environment to verify that the developed functions, using different techniques, can work together producing the desired result in an efficient way. The plan is to test these techniques at power plants.

During the CAMS design, a considerable effort has been given to maintain the generality of the CAMS concept; although the referenced process has been so far a BWR nuclear plant, the use of this structure and design can be applied to other processes, including non-nuclear processes.

CAMS is a system being developed as a joint research activity at the Halden Project in close cooperation with member organizations with additional financing from the Swedish Nuclear Inspectorate (SKI) and the Nordic NKS/RAK-2 project.

TABLE OF CONTENTS

1. INTRODUCTION

CAMS (Computerized Accident Management Support) is a system that will provide support in normal states as well as in accident states. Support is offered in identification of the plant state, in assessment of the future development of the accident, and in planning of accident mitigation strategies. (It does not give support in *execution* of the chosen mitigation strategy.)

We imagine different types of users: operators and shift leaders in the control room, the staff in the technical support centre (TSC), and people in the national safety authorities. These different types of users need different types of support.

CAMS picks up information from the plant and transforms it into a more digestible form before presenting it to the users. This transformation process can be controlled by the user.

CAMS consists of a data acquisition module (DA), a signal-validation module (SV), a tracking simulator (TS), a predictive simulator (PS), a state-identification module (SI), a probabilistic safety assessment module (PSA), and a man-machine interface module (MMI). The work of these modules is coordinated by a module called the system manager (SM). In addition, there are the strategy generator (SG) and the critical function monitor (CFM), these two are not integrated into the present version of CAMS.

The purpose of the prototype is to study how advanced information techniques can be utilized efficiently in accident management. Various methods are tested. The possibilities and also the difficulties of the chosen design are evaluated.

The design of the first CAMS prototype has been described in an earlier report, HWR-390, Reference [1]. Since then several pieces of work have been done:

- CAMS has been tested at a safety exercise at the Swedish Nuclear Inspectorate in May 1995. This work has been reported in a joint paper between staff from the Swedish Nuclear Inspectorate and staff from the HRP, Reference [2]. The main conclusion was that CAMS was useful already in the incomplete form that it had at that time, and that it had room for improvement and extension.
- A lessons-learned report has been written, Reference [3], about on-line simulation and estimation, to review ideas that can be used in CAMS. Some of these ideas have been used in the TS.
- A study has been made of fuzzy-logic methods for plant-state identification, Reference [4]. These methods look promising, but time has not permitted us to implement such methods into the new prototype.

Since the design reported in Reference [1], the prototype has been expanded in several ways:

- A special data acquisition module has been added, making it easier to couple CAMS to any data source, be it a plant or a simulator.
- The SV used an approach with a single neural network, and this module had not been integrated into the prototype. Now it has been expanded into an approach using combination of fuzzy logic and neural networks, and this expanded SV has been integrated into the prototype.
- The TS has been added to the system. The tracking is made by adjustment of parameters rather than of variables, using a least-squares criterion.
- In addition to the version describing the plant Forsmark-2, a version for another plant, the Barsebäck-1, has also been made. It is, however, the Forsmark-2 version which has been integrated into the present prototype.
- The SI, which existed only in a rather rudimentary form in the previous prototype, has been expanded and integrated into the prototype.
- A PSA module has been written and integrated into the prototype.

- New pictures have been added to exploit the new modules; SV, SI and PSA. The new and improved trend system of Picasso-3 version 2.0 has been taken into use and all the supporting programs use the new application program interface.
- As the prototype now contains a large number of modules, a SM has been added to coordinate the other modules.
- Emphasis has been placed on making a general design and structure facilitating easy maintenance and adaption to different reactor types. Although specific plant knowledge is implemented inside each module, it should not be necessary to rearrange the whole design when changing to another plant.

The present prototype has been made for a boiling-water reactor, but the possibility of making a version for pressurized-water reactors will be investigated.

This report will concentrate on the prototype as it is today, its various modules, and how they work together.

2. THE STRUCTURE OF CAMS

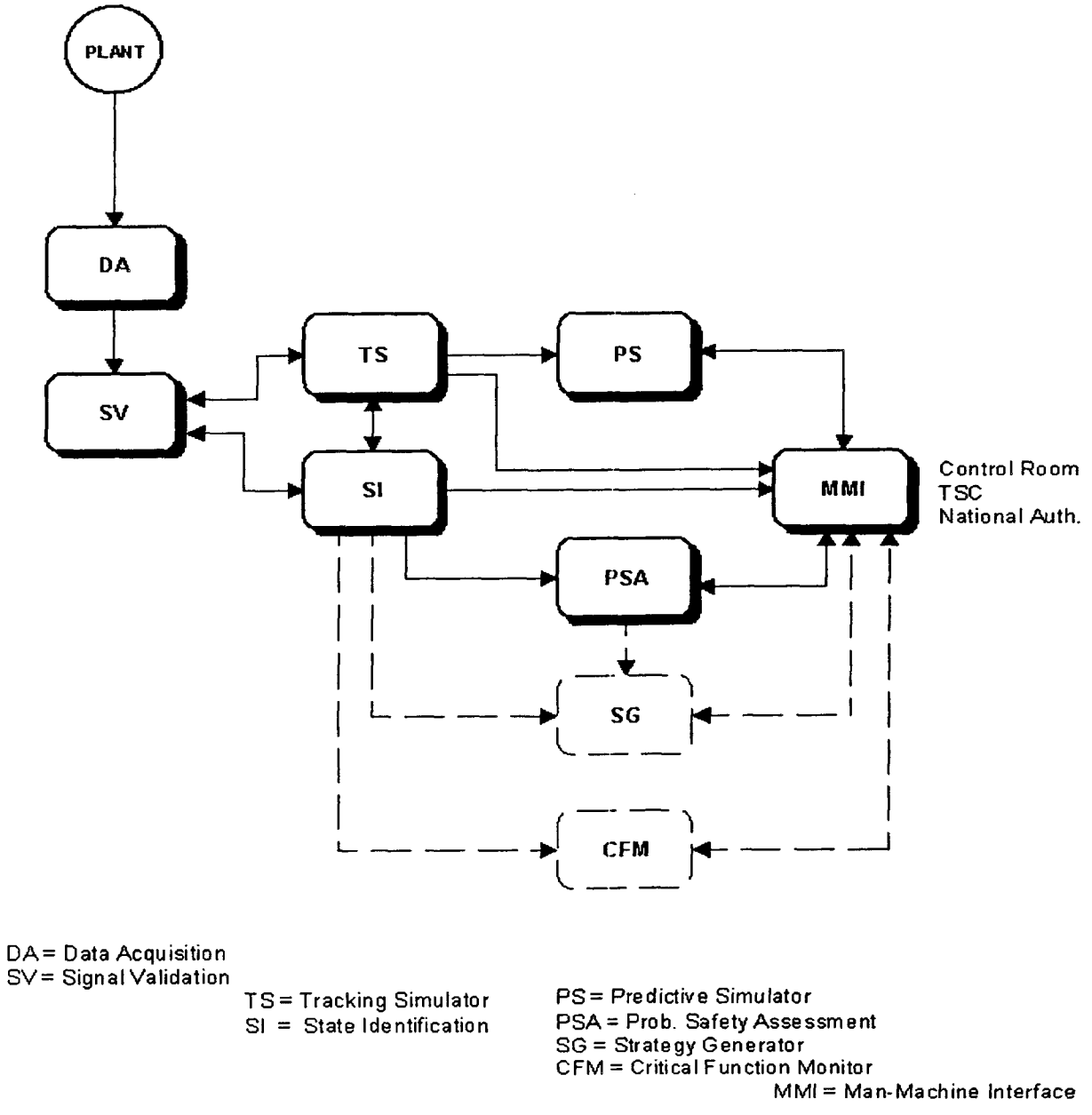

Figure 1. CAMS main components and the functional links between them

Figure 1 shows the main modules of CAMS and the data flow between them.

First we note that CAMS is an information system, data flows from the plant to the user. You can influence what goes on in the modules close to the Man-Machine Interface, but you cannot operate the plant through CAMS.

The plant data are picked up by the Data Acquisition module (DA). From there they flow to the Signal Validation module (SV). Validated data flow to the Tracking Simulator (TS) and to the

State Identification module (SI). The TS augments the measurements with three sorts of extra data:

- data which the user should like to know, but which are not measured,
- data that are used for initialization of the predictive simulator, but which are not measured, and
- data that *are* measured, but which also can be calculated from independent measurements.

In the latter case the TS acts as the calculation assistant of the SV. The cooperation between the SV and the TS is indicated by the double arrow between them.

From the TS, validated and augmented data are available to any module that may request it. In addition to the SV already mentioned, the main customers for such data are the Man-Machine Interface (MMI) and Predictive Simulator (PS).

When requested by the user, the PS will pick up the current state from the TS. The PS may be asked to predict what will happen if no intervention is carried out, or if a certain sequence of interventions is carried out.

The SI produce qualitative information about the plant state: there is or is not a leakage, a component is or is not available, etc. This information is communicated to the user by the MMI. It is also the starting point for the analyses done by the Probabilistic Safety Assessment module (PSA), the Strategy Generator module (SG), and the Critical Function Monitor (CFM).

As indicated, the SG and the CFM have not been integrated into the present version of the prototype.

Data from all these modules flow to the Man-Machine Interface (MMI) to be examined by the user. The user can control the data transformation going on in the PS, PSA, etc., this is indicated by the arrows going backwards from the MMI to these modules.

Later in the report the different modules will be described in more detail.

3. PLANT

3.1 Purpose

During the development and testing of CAMS, it was necessary to feed CAMS with data corresponding to several sorts of accident states. A fake plant was made for this purpose. This is not part of the CAMS system, and serves test purposes only.

3.2 Description

The fake plant contains a model of the plant, identical to the one in the predictive simulator, but not with the same data. Inside this model all sorts of data are available, but the Data Acquisition module is only permitted to read those quantities that are measured in the real plant. It could be nice to know say the temperature in the middle of a fuel element, but as this is not measured in the real plant, this is not available from the fake plant either. CAMS will have to estimate such data from those quantities that are measured.

In real life we do not have a complete knowledge of the plant, and we cannot make a perfect model of it. Therefore, in the future a fake plant will be made that deviates somewhat from the models used in CAMS. This will make the tests more realistic.

Already in the present version of CAMS, noise is added to the measurements from the fake plant. This function is taken care of by the Data Acquisition module and is described under DA core, Section 5.2.1.

4. THE SYSTEM MANAGER (SM)

4.1 Purpose

The SM (which is not shown in Figure 1) is the common functional interface to all the CAMS modules. The main tasks performed here are:

- *Functional linking*: All the modules communicate with each other only through the SM.
- *Synchronization*: Activity in different modules must be synchronized to produce meaningful outputs. This is particularly true in this system where the modules operate concurrently.
- *Switching:* Every module can be switched on and off without limiting the operation of other modules.
- *Monitoring:* The SM controls the activity and the flow of information through all the modules. A dedicated SM display has been designed, to be used by the CAMS supervisor on site.

The SM has been developed using the real-time expert system shell G2, by Gensym Corp., Reference [5].

4.2 Description

Figure 2 shows the logical connections among the CAMS modules and the SM. External processes (external to G2) used by each module to perform a task, are represented in circles. In this diagram the modular nature of CAMS is emphasized: each functional module has the same structure and all of them work concurrently under the supervision and coordination of the SM. Basically, each module has the following building blocks:

- an external process, performing the main task (for example a Picasso-3 process for the MMI module),
- a G2 module, that drives the external application and receives the results,
- a communication block (dac, svc,...), which contains the data that must be shared with the SM.

The CAMS prototype is built by using several different software packages and systems;

Fake plant	APROS, Reference [6]
System Manager	G2, Reference [5]
Data Acquisition	G2 and C++
Signal Validation	Matlab, Reference [7] and G2, Reference [5]
Tracking Simulator	APROS, Reference [6] and C
State Identification	GPS, Reference [8]
Predictive Simulator	APROS, Reference [6]
Probabilistic Safety Assessment	C/C++.
Man-Machine Interface	Picasso-3, Reference [9]
Strategy Generator	G2
Critical Function Monitor	G2.

All of these different systems are running on several computers in a network. For the software communication part of this hybrid system, programs written in C++ using a combination of Orbix[11], the Software Bus [10] and shared memory had to be made.

Figure 3 shows the internal structure of the SM. This structure (completely developed in G2) is also modular to facilitate maintenance and future expansion or changes to adapt CAMS to other processes.

Figure 4 shows one possible display from the SM console. Here the Data Acquisition main controls have been selected.

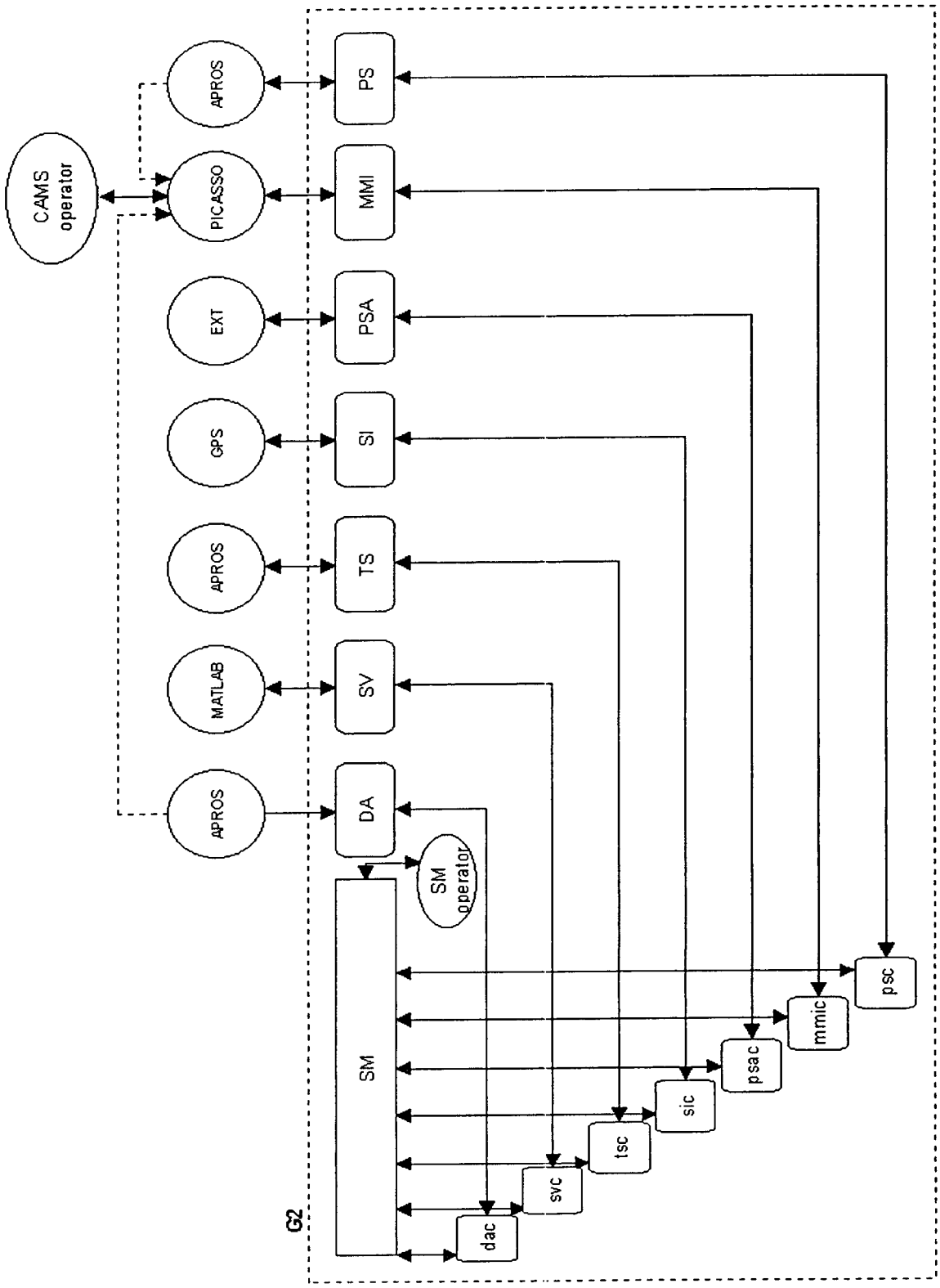

Figure 2. System Manager connections

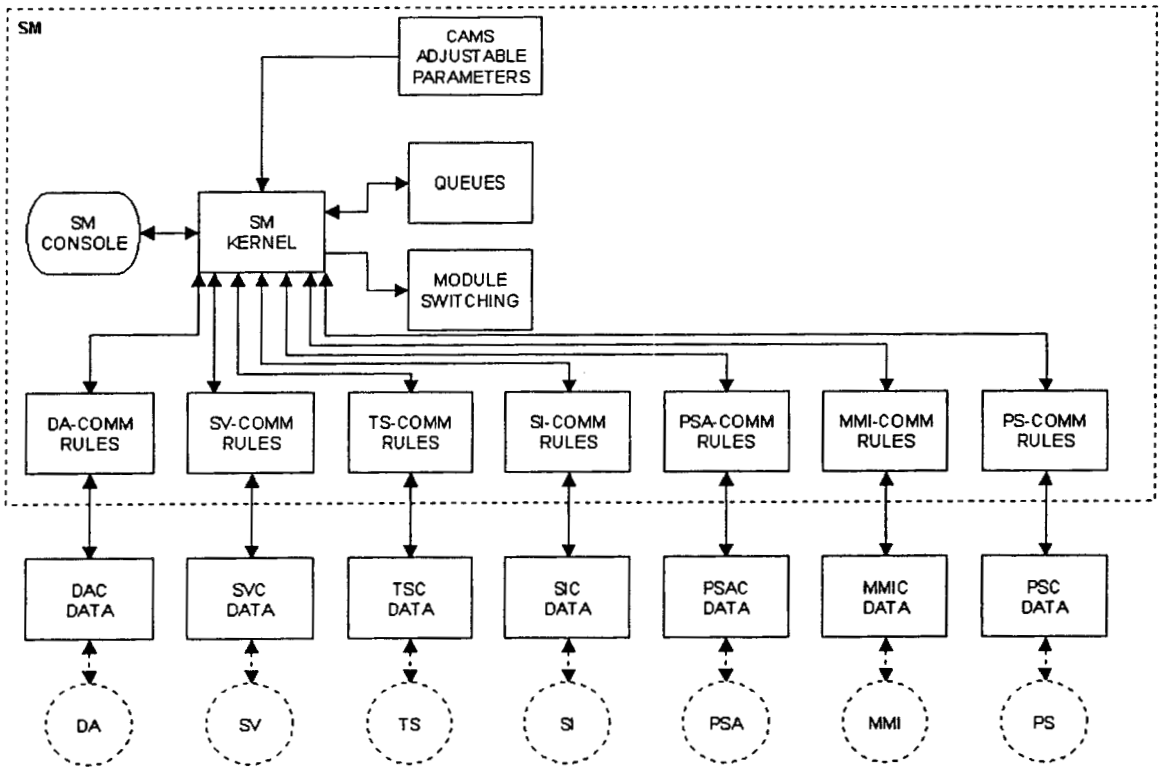

Figure 3. System Manager functional diagram

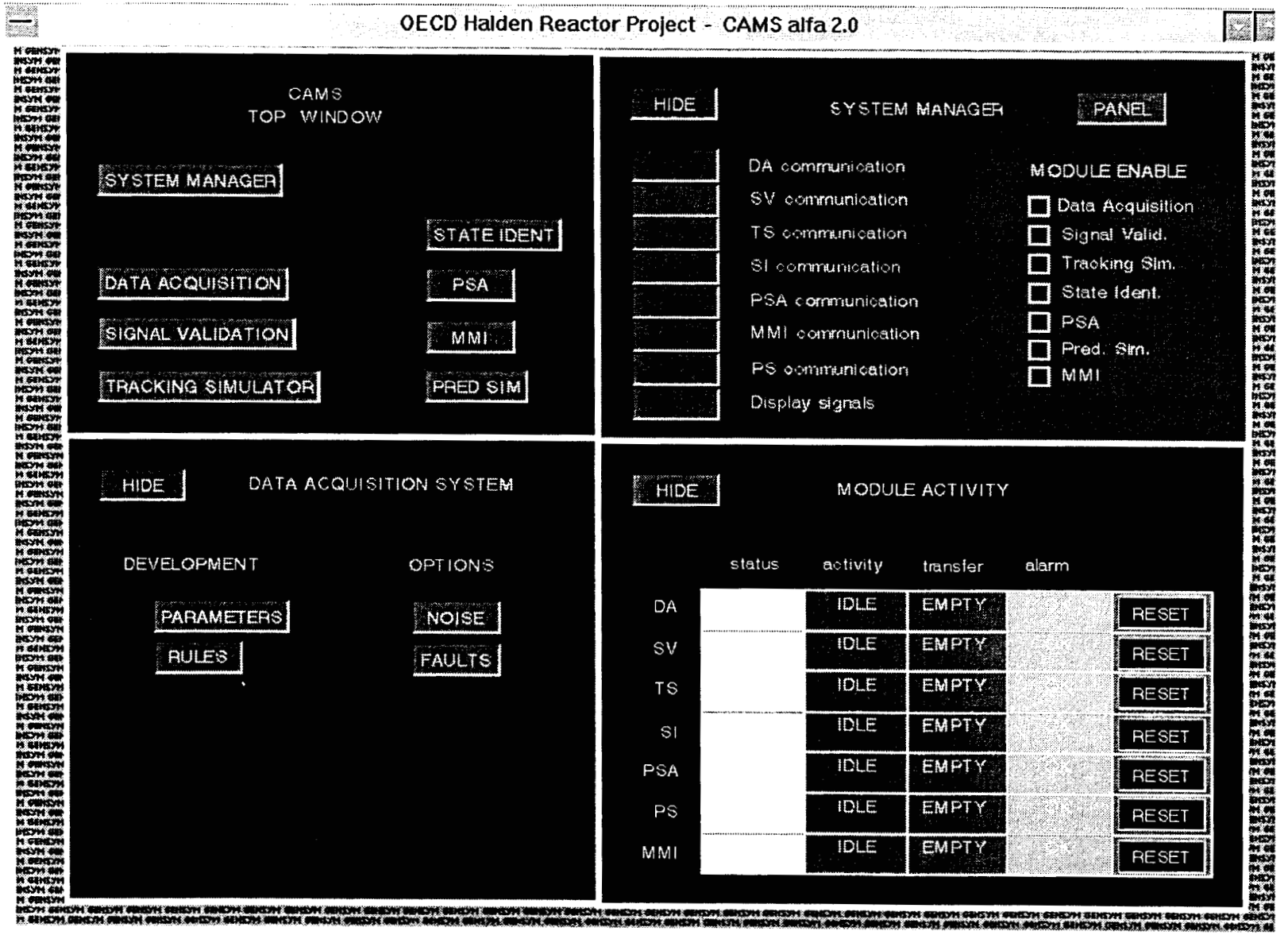

Figure 4. The System Manager console

5. THE DATA ACQUISITION MODULE (DA)

5.1 Purpose

This module operates as an interface between the CAMS and the monitored process. The main requirement in the development of the DA module was to avoid any dependency of other CAMS modules of the external data source, with only a few exceptions.

5.2 Description

Figure 5 shows how the DA module is connected to the rest of the system. A description of the blocks comprising the DA process follows:

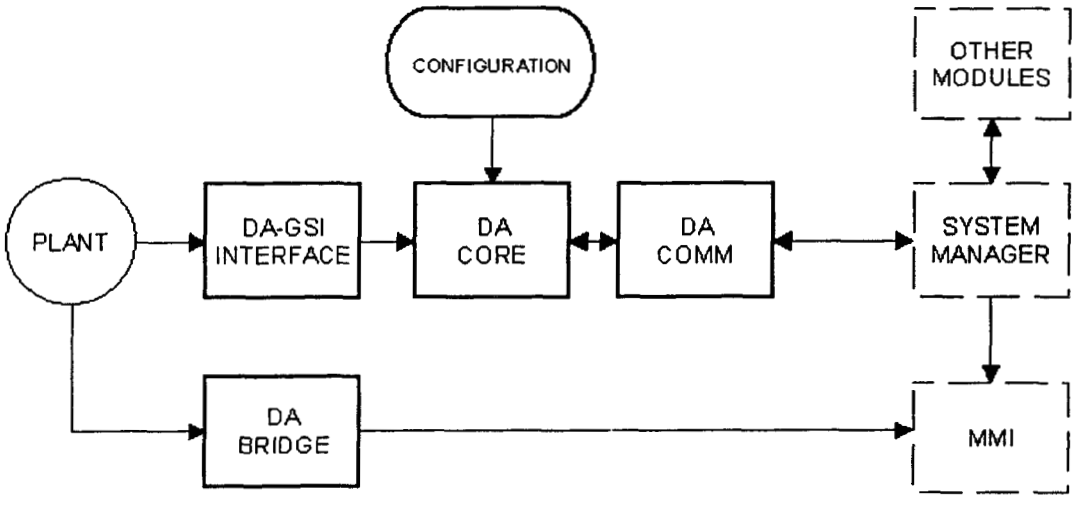

Figure 5. DA logical diagram

5.2.1 DA core

This is where the plant signals are acquired and stored to be processed on demand. At any time the last N samples are available for each signal. The length N of the memory can be adjusted.

Two additional functions are present in this block, to be used for test and training purposes:

- The noise simulator. It is used to add random Gaussian noise, at an adjustable level, to one or more incoming signals.
- The fault simulator. It is used to simulate drifts or failures in a limited set of signals.

This block has been implemented G2.

5.2.2 Configuration

This block contains information on the number, type and characteristics of the process (the plant). For each signal the following information is provided:

- Name. To be used in the other CAMS modules.
- Nominal value.
- Validation flag. Tells if this signal must go through the SV module or not.
- Rough signal validation parameters.

- Noise detection flag. To be used for detecting stuck signals.

For each signal, three queues are maintained and available to the rest of the system:

- The DA queue. A list of the last N samples (N is an adjustable parameter) as acquired from the plant.
- The SV queue. A list of the last N samples of the signal after the validation process.
- The TS queue. A list of the last N samples of the signal, as tracked by the TS module.

The configuration block has been implemented in G2.

5.2.3 DA comm

This block handles the information to be exchanged with the System Manager. A block like this exists in all the CAMS modules, to identify the link between each module and the SM. No link exists between DA and any other module in CAMS, except SM (the Bridge block is an exception).

This block is implemented in G2.

5.2.4 DA bridge

This is a direct link from the plant to the Picasso-3 display. It is used to update the display with information not relevant for other CAMS modules. This block might be removed in the future, because it is not consistent with the module-isolation concept that is the basis of the current CAMS design.

5.2.5 DA-GSI Interface

This is the software interface between the plant and the G2 process where DA has been implemented. A block like this exists in all the modules where there is a need to interface external processes.

6. THE SIGNAL VALIDATION MODULE (SV)

6.1 Purpose

The Signal Validation module (SV) in CAMS is currently based on neuro-fuzzy techniques. The complete algorithm and procedure can be found in Reference [12]. Currently, a set of reactor safety related signals has been used in the SV module (see Table I.) but the same design can be applied concurrently to other sets of process signals.

Sensor name	Range	Validated
Core power	0-100%	Yes
Control rods position	0-3.65 m	No
Core flow	0-10500 kg/s	Yes
Core pressure	0-7 MPa	Yes
Feedwater flow	0-1500 kg/s	Yes
Steam flow	0-1500 kg/s	Yes
Feedwater temperature	0-170 °C	Yes
Control valves position	0-100%	No
Bypass valves position	0-100%	No

Table I. Signals used in the validation model

6.2 Description

Figure 6 shows a simplified diagram of the neuro-fuzzy model. The possibilistic fuzzy classifier is used to identify one or more possible regions of the process operating point (as defined by the set of signals to be validated), to which the incoming sample could belong. The possibilistic nature of this classifier results in a prompt detection of patterns outside the module training volume (which can introduce unacceptable errors in the neural networks response).

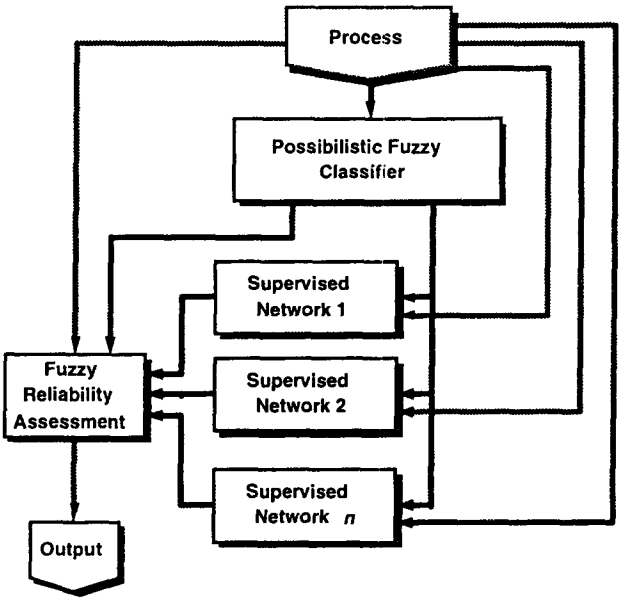

Figure 6. Neuro-fuzzy Signal Validation functional diagram

The neural networks (ANN) have been trained, each in a different region, in the set of the possible regions identified by the classifier. They work concurrently during the validation process and their output is averaged using the fuzzy membership value of the incoming pattern in each cluster (region). In this implementation, seven clusters have been identified, which cover the entire power-flow map of the reactor.

Figure 7 shows how the complete system works. Basically, the validation task is performed in the following steps:

- Get from the System Manager the last 6 samples for each input signal.
- Apply the pre-screening rules to the signals, where applicable. Information concerning applicability of pre-screening rules comes from the signal database, maintained by the SM and made visible to all the other modules. Currently, the following pre-screening rules are applied:
  - range check,
  - maximum positive derivative check,
  - maximum negative derivative check,
  - stuck signal check.
- Input the last set of samples to the classifier. The output from that is a vector of 7 possibilistic membership values in the 7 identified clusters.
- Activate the ANNs where the associated cluster has a membership value above an adjustable threshold, for the incoming pattern.

- Calculate the weighted average of the ANNs outputs, where the weights are the membership values calculated above. These are the validated signal values.
- Input the membership vector and the maximum mismatch value (the normalized absolute difference between the validated and input values for the samples) to the fuzzy reliability model, to calculate three membership values for the three fuzzy sets HIGH, MEDIUM and LOW. See Reference [12] for more details on this model.
- If the signal mismatch is above a user-adjustable threshold or one or more pre-screening rules are triggered, send a corresponding alarm to the SM. The three reliability membership values have three corresponding lamps in the CAMS SV display, to continuously inform the user about the current reliability of the model.

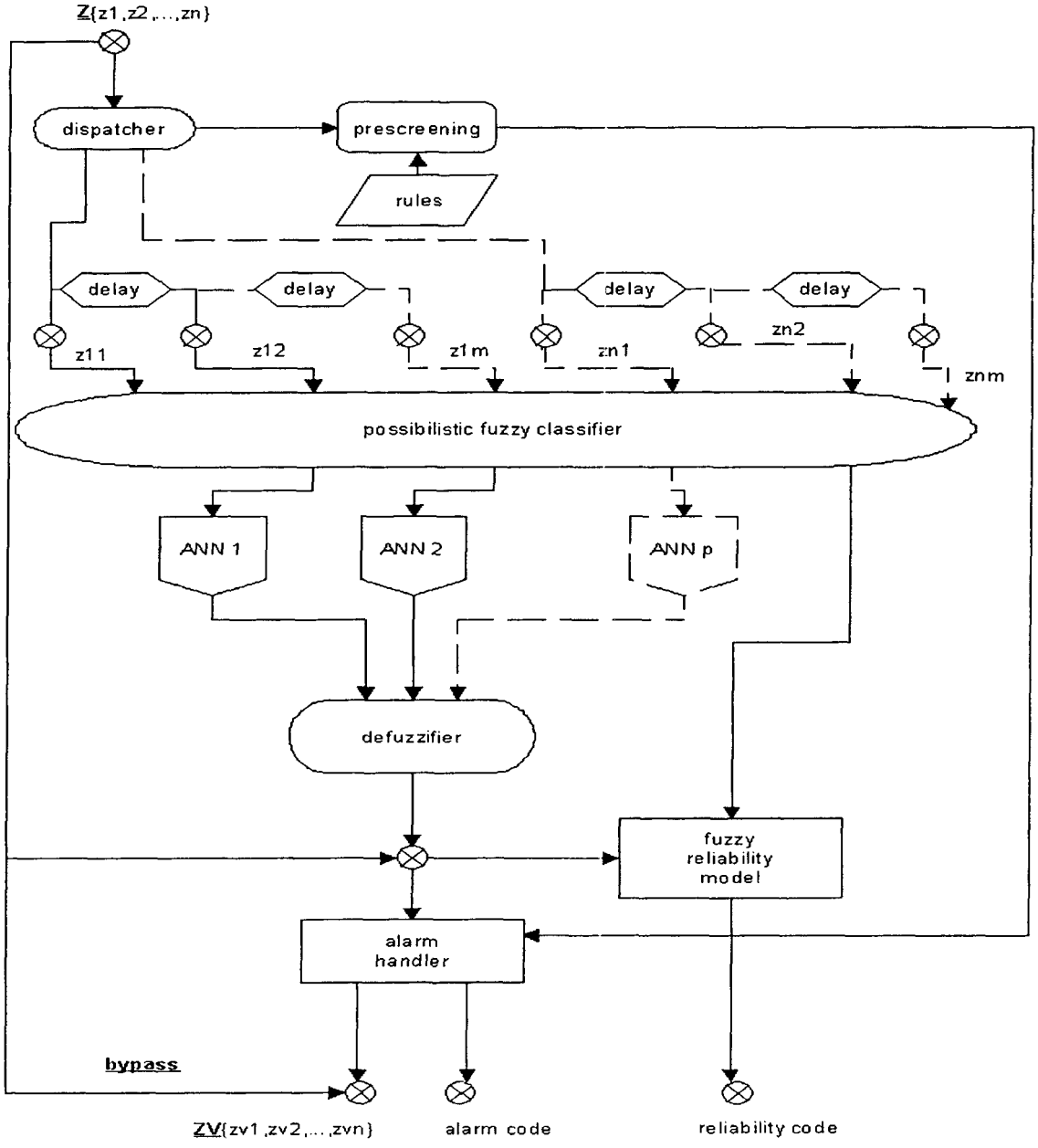

Figure 7. SV model flow chart

6.3 User interface

The SV communicates with the user through the MMI module and the SM. A new picture has been designed, see Figure 8, with the following features:

- The list of the signals with the validation flag "on" (that is, signals which go through the SV module) is always displayed in the picture. The status of each signal (confirmed, drift, stuck,...) is represented using text and colour information.
- The current reliability status is represented by three lamps (green, yellow and red) reflecting the current status of the validation process. The red lamp should always be off, for acceptable reliability levels.
- The user can bypass the module. In that case, the validation process is performed regularly and the alarms are displayed, but the TS and SI modules will use the non-validated data coming from DA, rather than the validated signals. Note that the bypass concept is different from the module enable/disable capability controlled by the SM. If the SV module is disabled, no validation check is performed.
- The user can set the alarm threshold parameter, which represents the maximum percentile signal mismatch (in per cent of the rated) that the system allows before issuing an alarm condition.

6.4 Testing

This model has been tested in normal and abnormal plant conditions, against single and double signal failures, with good results. Please see Reference [12] for a complete description of the tests performed and the results obtained.

6.5 Advantages and drawbacks of the model

The neuro-fuzzy validation model implemented in CAMS is the result of a research effort at the Halden Reactor Project in the last two years, to verify feasibility and limitations of Artificial Intelligence approaches in signal validation methods for nuclear applications, in comparison to other methods (mainly estimation methods using analytical redundancy algorithms). The following conclusions about this method have been derived so far:

- ANN models do not require exact knowledge of the monitored process. They are particularly suitable when the process behaviour is not known or is too complex to be modelled with enough accuracy for validation purposes.
- ANN models can be designed to adapt to process changes with no need to understand the causes which lead to a different behaviour in the process (fuel burnup, fouling factors, components efficiency, etc.).
- ANN models have a well-known robustness to noise. The efficiency of this model is almost unchanged when signal noise is artificially increased.
- This model is able to promptly detect plant conditions where it is not capable to perform its task in a reliable way. In other words, it has been designed to say "I do not know", when applicable.
- ANNs need a large amount of experimental data. If not available, the use of simulated data reduces considerably most of the above advantages. Moreover, experimental data in abnormal conditions are normally not available, so the use of ANN methods in a post accident situation can be limited. However, this is a general problem in the signal validation area, because of the limited knowledge of accident progression in nuclear power plants and the limited efficiency of the current computer codes in severe accident conditions.

- ANN models learn by examples, and they learn quite slowly. A huge computational effort and time may be required, depending on the complexity of the process. Moreover, their efficiency collapses outside the training volume, even if their generalization capability is high. As a consequence, they perform poorly in not-anticipated conditions (because they can be trained only in anticipated conditions).
- ANN models are black-box applications. Their knowledge is represented by the weights matrix and it is not possible yet to reverse the process to get information about the learned model. This can create an uncomfortable feeling in many users.

6.6 Future development

The neuro-fuzzy model currently implemented in CAMS can be enhanced in many ways. The following tasks have been identified for future expansion and/or enhancement:

- Adaptive operation. The training algorithm can be designed to use the experience during the monitoring time, to adapt the weights matrix to slow changes in the process.
- Adaptive reliability model. From the test results it seems that the membership functions of the three reliability fuzzy sets should change according to the plant condition. This could increase the robustness of the reliability model. In particular, this non-adaptive model tends to assign medium reliability tags, also in situations where the system is working quite properly, in some stressing conditions (presence of double signal faults, for example).
- Integration with other methods. A possible combination of this model with estimation methods can be considered. This will address, for example, the well-known problem of discriminating clearly a signal failure from a process failure.

Figure 8. SV user interface

7. THE TRACKING SIMULATOR (TS)

7.1 Purpose

The purpose of the TS (Tracking Simulator) is to calculate:

- Quantities that are not measured, but which the user should like to know.
  Used by: MMI.
- Quantities that are not measured, which are to be used as initial values when predicting what will happen.
  Used by: PS.
- Quantities that are measured, but which can also be calculated from other independent measurements.
  Used by: SV.

7.2 Requirements

For the prototype of May 1996, the required estimates are:

- Cladding temperature,
- Relief valve flow,
- Steam leakage inside the containment,
- Water leakage inside the containment.

More requirements may be added to this list later, for instance:

- Water level in the reactor tank (for the SV).
- Steam leakage outside the containment,
- Water leakage outside the containment.

Further, some estimates may be available at no extra cost.

7.3 Description

There are several tools available for building simulators, but they all appear to be made for building predictive estimators, rather than tracking ones. The facilities they offer are essentially this:

They have a set of ready-made components like pipes, valves, pumps, turbines, reactors and so on. You can describe the components of your plant by giving parameters to these ready-mades. I have a pipe, you say, the tool gives you its pipe component and ask you to describe your pipe. Its length is this, you say, and its diameter is that.

You then describe the topology of your plant. The water running out of this tank runs into the pipe, the water running out of the pipe runs into that valve. The modelling tool will establish and solve the corresponding algebraic equations.

Finally you give the initial conditions of your plant and ask what the situation will be a certain time from now. The tool will establish and solve the corresponding differential equations.

No suitable tool for making tracking simulator seems to be available on the market. How can we turn a predictive simulator into a tracking simulator? What we want is an estimation tool. And estimation has a mathematical relationship to prediction.

We receive new measurement data at regular intervals $t = k\Delta t$. But not all the interesting state variables are measured. We arrange all measured variables $y_1, y_2, \ldots, y_L$ into a vector y, an arbitrary one of these measured variables is denoted by y_l. Similarly, we arrange all non-meas-

ured variables $z_1, z_2, \ldots, z_M$ into a vector z, an arbitrary one of these non-measured variables is denoted by z_m. Finally, all parameters $p_1, p_2, \ldots, p_N$ that shall be updated are arranged into a vector p, an arbitrary parameter is denoted p_n. The complete state vector x is then the collection of all these:

$$x = \begin{bmatrix} y \\ z \\ p \end{bmatrix}. \tag{1}$$

Here, $1 \le l \le L$, $1 \le m \le M$, and $1 \le n \le N$. The number of adjustable parameters N should preferably be much smaller than the number of measurements L.

Non-measured and measured variables are for instance temperatures, pressures, positions, velocities, quantities for which we can establish equations describing the main part of their development with time, even though our knowledge is never precise.

Parameters are quantities that normally do not change, or at least change only slowly, like lengths and cross sections of pipes, the thicknesses of steel walls and electric resistances. The existence or non-existence of a leakage also is included in this category. These latter things, which we often call "constants", may also change, but we are not able to predict how, or we decide not to do so even if we could. We shall assume that a parameter does not jump wildly around, it displays some sort of continuity. Rather than making the parameter directly a Gaussian stochastic variable, we make its change between one measurement and the next a Gaussian stochastic variable. This does not mean that parameters cannot change, only that we cannot predict how it will change. The entire time development of these quantities is described by a noise term.

The classification into variables and parameters is not always obvious. If we for example enlarge our description of flow phenomena to also include the corrosion of a steel pipe, and establish equations for the change of the wall thickness with time, the wall thickness will be a variable rather than a parameter.

Given initial values for estimates of y, z and p, a prediction tool can tell you the corresponding estimates one time interval later. The p's are trivial, of course, as the prediction will be that they have not changed.

Then the measurements at $t = (k+1)\Delta t$ arrive, and we have new values for the y's. They will be similar to the estimated y's, but not identical. The differences between measurements and estimates are called *residuals* or *innovations* and they are denoted by r. They represent what we have learnt by the measurements, the amount of surprise. We can use them to improve our estimates of the measured variables y, the unmeasured variables z, and the parameters p.

In each time interval the state vector is therefore changed twice, by the equations of motion which are handled by the prediction tool, and the updates based on the residuals. This is illustrated in Figure 9.

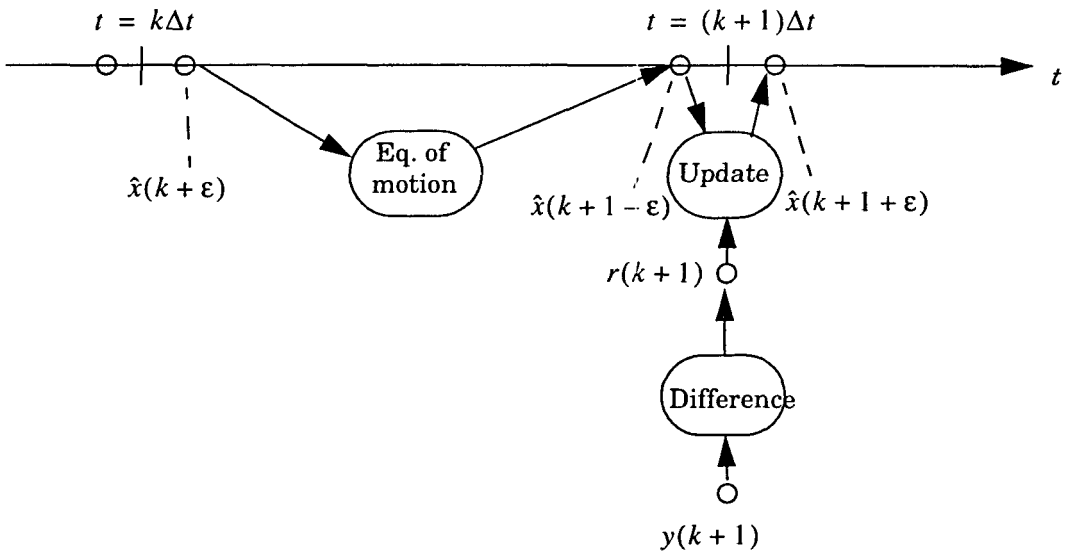

Figure 9. The state vector is modified alternatively by the equation of motion and the update equation. The update equation is fed from the residual equation.

There is the well-established theory of the Kalman filter. This theory gives a recipe for how to update the state vector which is optimal, provided certain conditions are fulfilled - which they are not in our case. First of all, the model of the dynamic system has to be a perfect model, and our model is by no means perfect. An additional difficulty is that the Kalman filter leads to numerically ill-conditioned equations when a large number of quantities shall be estimated. We therefore have to do something which is simpler, but non-optimal.

The variables will change in such a way that certain conservation laws are respected, when all such quantities are regarded together. If the amount of water increases in a tank, the amount of water will decrease in another tank, in such a way that the total amount of mass is conserved. Or it may turn up as a change in the amount of steam, again in such a way the total mass is conserved. A conservation law usually involves masses, momenta, or energies. Parameters, on the other hand, describe lengths and cross sections of pipes, friction factors, resistances and similar things and they usually do not obey conservation laws. We conclude that it is easier to update parameters than variables. We shall use measurements of variables to adjust parameters so that the model fits the measurements.

We do not update any variables, neither measured nor non-measured, when new measurements arrive. The experimental information is used for updating the parameters only. This means that the estimates of the variables are changed once every time interval by the prediction tool, and the parameters are also changed once every time interval, by the update mechanism. The parameters are then modified to make the residuals as small as possible according to a least-squares criterion. This simplified process is illustrated in Figure 10.

A preliminary version of this procedure has been implemented in the present prototype (May 1996).

A more complete description with all the mathematics is given in Appendix I.

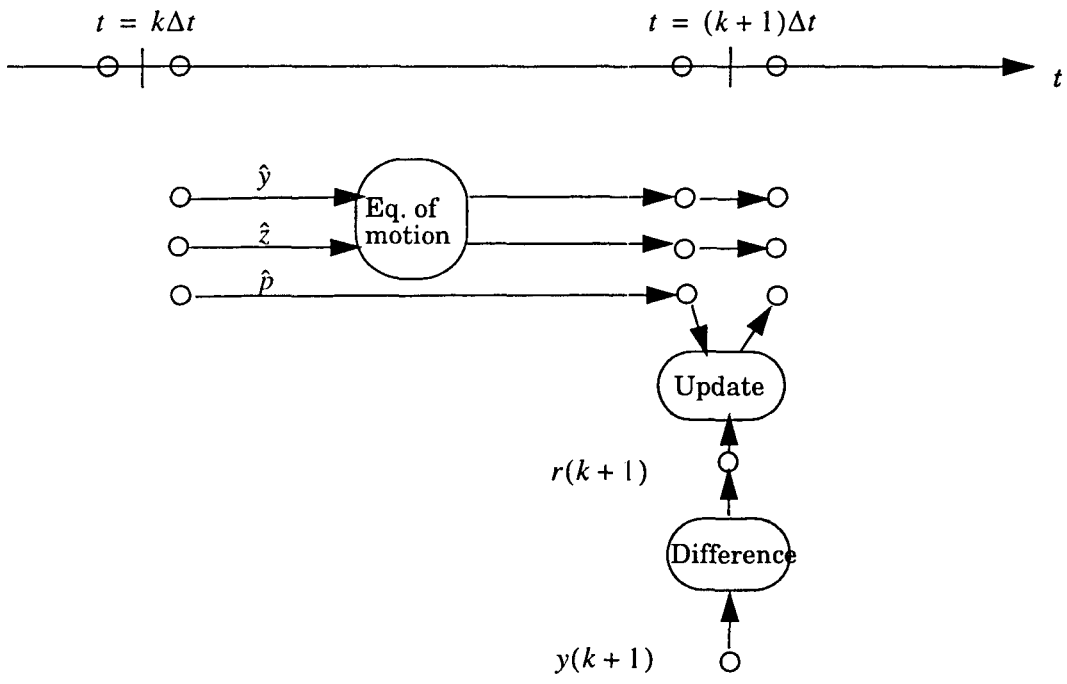

Figure 10. The estimates of the variables y and z are modified by the equations of motion. The estimates of the parameters p are modified by the update equation.

7.4 Future development

The method outlined here and described in more detail in Appendix I implies finding a set of parameters such that a given criterion is a minimum. The standard procedures for finding such a set of parameters include working out the derivatives of the criterion with respect to the parameters. In our case these relationships are only known numerically. Numerical derivation is a difficult thing: using too large differences, the nonlinearity of the relationships corrupt the result, using too small differences, and you have numerical inaccuracy.

Procedures have been described in the literature that avoid working out the derivatives. Such methods should be studied to see if they can be applied to our problem.

8. THE STATE IDENTIFICATION MODULE (SI)

8.1 Purpose

The purpose of the State Identification (SI) module is to identify the state of the plant and to communicate information to the user and to other CAMS modules. In the current version, the outputs of the SI module are the following

- output to the user: textual information describing the plant state, status of critical safety functions and availability of safety systems (see the MMI chapter for more details),

- output to the TS: presence of a steam or water leakage inside or outside the containment and the occurrence of steam release through relief valves,
- output to the PSA: the occurrence of initiating events and the availability of front-line and support systems.

8.2 Description

The state identification module is a knowledge based system with the classical main components: a knowledge base and an inference engine. It is developed using a specific tool: GPS (Goal Planning System, Reference [8]) which provides the inference engine and the knowledge-acquisition tool.

8.2.1 The tool

GPS is a specific tool designed to manage "process related" knowledge and aimed at process supervision via real-time acquisition of the process variables.

GPS works by following a repeated read-process-wait cycle. During each cycle, the program acquires data, processes this data and then waits for the following cycle. During the processing phase, the program matches the acquired data with its knowledge and gives information to the user. No user interaction is needed. The cycle duration is now fixed to 5 seconds, but it could be changed if necessary.

The knowledge used by the program is produced through a dedicated knowledge-acquisition interface. The knowledge is represented by an acyclic oriented graph of knowledge units called *goals*. This graph defines a father-sons hierarchy that represents the logical structure of the knowledge. In the graph, goals can be linked by different kinds of operators, characterised by a logical aspect (AND - OR) and by a temporal aspect (sequential - parallel). The graph is divided in subgraphs to make the knowledge base easier to consult.

The detailed knowledge is described inside goals. These are characterised by several attributes among which the success criterion, which is a function of some process variables, and the information that have to be sent to the user. Goal attributes are expressed in the "Knowledge Acquisition Language", a small subset of the LISP programming language, added with special functions.

When the inference engine is running, the goals are activated according to the logical structure of the graph. Goals can succeed or fail depending on their success criterion and the current value of process variables and information is sent to the user in accordance with their success or failure.

The GPS software is written in LISP and CLOS (Common Lisp Object System).

This tool has been developed at Tractebel Energy Engineering (Belgium) in the framework of a previous project: OPA (Operator Advisor, Reference [8]), a support system for nuclear power-plant operators. The knowledge base was then derived from the operating procedures. The same tool is now integrated in CAMS and reused with a new knowledge base.

8.2.2 The knowledge base

The current knowledge base contains 3 parts:

1. Safety objective trees

State Identification is based on Safety objective trees as defined by the NRC, Reference [13].

The safety objective trees identify the relationships between plant safety objectives, challenges to the safety objectives, mechanisms that cause the challenges and strategies that would mitigate or prevent the mechanisms using a hierarchical tree structure. We have chosen to represent safety objective trees, from the safety objective level to the mechanism level. The strategy level will be the purpose of the future SG module.

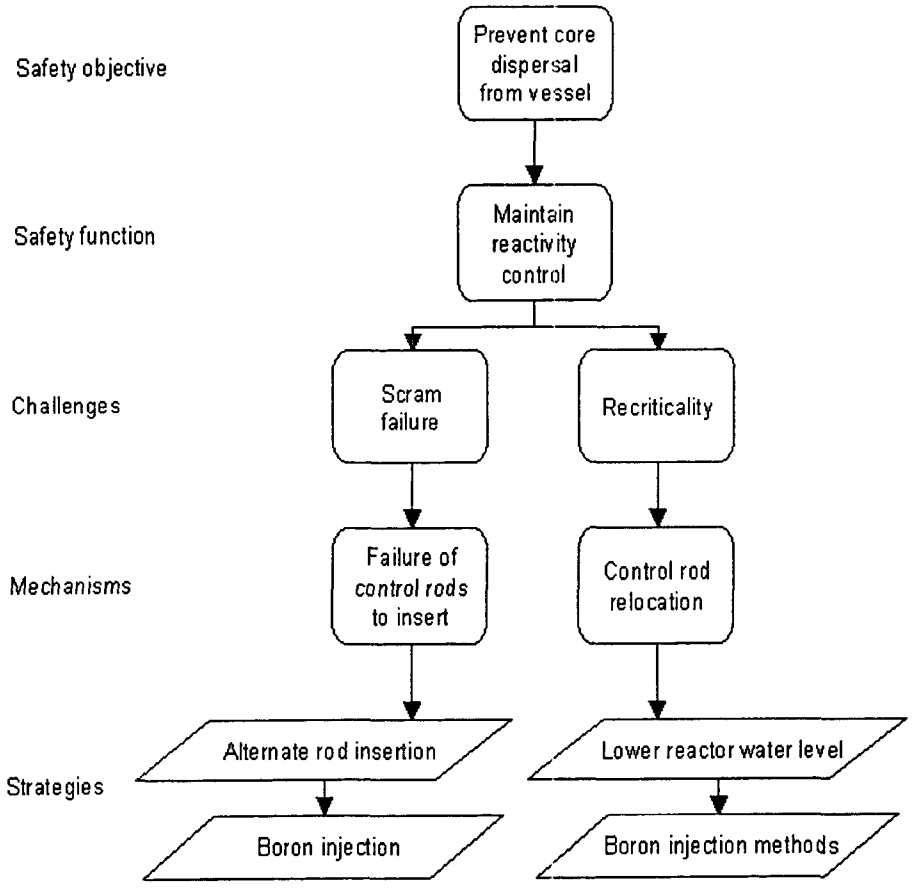

Figure 11. Example of a safety objective tree

So far, we have implemented the trees related to the following safety objectives:

- prevent core damage,
- maintain containment integrity.

2. *Leakage detection and steam release*

The current knowledge base only detects leakage inside the containment by monitoring the evolution of the containment pressure and temperature and the water level in the reactor vessel.

Steam release through relief valves is monitored by checking the position of the relief valves but also the temperature in steam relief lines to prevent errors due to wrong information about the position of the relief valves.

3. *Safety systems availability and initiating events*

The safety systems monitored by the SI module are automatically started under given plant conditions. When the starting conditions of a safety system are fulfilled, the SI module checks if the system is really working and if not, it tries to identify the reason why the system does not work.

For example, when the starting conditions for the low pressure cooling system are met, the flows in the low pressure cooling lines are checked. If there is no flow, the system is declared unavaila-

ble. Then the speed of the pumps and the position of the related valves are verified and the user is informed of abnormal situations.

The considered initiating events are LOCA of different sizes (large, medium, small), manual or automatic shutdown of the reactor and loss of important functions like heat sink, feedwater or external power. The occurrence of these events is already detected in the previously described parts of the knowledge base. The information is transmitted to the user and to the PSA module.

8.3 The user interface

The state identification picture contains 3 parts: the critical safety functions, the unavailable safety systems and the messages. This screen is mainly an information screen. There is no need for the user to control the work of the state-identification module.

Figure 12. State Identification picture

8.3.1 Critical safety functions

Each one of the 4 critical safety functions (heat sink, core cooling, reactivity control and containment integrity) is represented by a coloured square, the colour corresponding to the status of the function: green for normal, yellow for abnormal and red for dangerous.

8.3.2 Unavailable safety systems

This window shows a list of the unavailable safety systems with the time when this information was found out. When an unavailable system comes back to normal state, it is not erased from the list, but the background colour changes and the time is updated. The user can erase all the systems back to normal state by clicking on a button.

8.3.3 Messages

This window contains textual messages describing the state of the plant. Each message is preceded by its generation time. Messages are displayed in chronological order. The user can clear the window by clicking on a button.

8.4 Future development

The SI module is still at an early prototype stage. Here are some areas for future work:

- The knowledge base should be extended to cover more of the safety objective trees, to monitor more safety systems and to detect more faults. The missing safety objective trees are
  - prevent core dispersal from the vessel,
  - mitigate fission product release from containment.
- One of the problems faced when filling the knowledge base is that threshold values have to be chosen that fix the border between normal and abnormal states. In the real world, this distinction is not so clear. The introduction of fuzzy logic concepts could be very helpful concerning this problem (see Reference [4]).
- The SI module could also use numerical information from the SV and the TS in addition to its own qualitative information to discriminate a signal failure from a process failure.

9. THE PREDICTIVE SIMULATOR (PS)

9.1 Purpose

The purpose of the Predictive Simulator is to tell the CAMS user what the future state of the plant will be, by running a model of the plant faster than real time. In the case of an accident, the user will see if the safety systems of the plant are sufficient as the accident evolves, or if some interference is needed to reach a safe state. If a safe state cannot be reached, the simulator will give indication of when the plant reaches a critical state.

When the user wants to run a prediction, he will initialize the simulator with the current state of the plant (which is a major task of the Tracking Simulator). The user can then let the simulator run by itself to see what will happen if no mitigation strategy is tried, or by manipulating the controls of the simulator he can test different strategies to see which effect they have and choose the better one.

The simulator can also be used to check what *might* happen before it happens (if ever). Say, we have lost all auxiliary feedwater pumps but one. If the last one should also fail, how much time do we have before the core is uncovered? With this information the user can be prepared if so should happen (this sequence of events happened at the safety exercise at the Swedish Nuclear Inspectorate in May 1995, see Reference [2]).

9.2 Description

No modules use the output from the simulator. The output goes directly to the relevant predictor pictures and the trend system within Picasso-3. But the control of the simulator goes through the System Manager. In this way the System Manager knows the commands to the simulator, without being loaded with all the display data.

Operating the simulator is done using three pictures, Figure 13 "Predictor Control", Figure 14 "Predictor Panel" and Figure 15 "Predictor Setup".

"Predictor Control" is similar to the process picture, which gives an overview of the process, regarding layout and placement of components. This has been done so that the user will recognize and be familiar with the picture during an accident. There is one important difference; in "Predictor Control" one can control the Predictive simulator. This is done using a mimic style interface where one can point and click on components. A small window will pop up and the state of that component can be modified (a pump can be started, a valve opened etc.).

"Predictor Panel" has a more traditional layout with buttons and sliders to mimic what operators are used to from the control room. This layout gives a more detailed control of the systems in the plant, like manual scram and suppression pool cooling.

"Predictor Setup" is used to start, stop and initialize the predictor. *Start* and *stop* are self-explanatory, and *initialize* is used to make the predictor reflect the state of the plant. Normally initializing will be activated when the plant has changed its state, or when the user wants to try a new mitigation strategy. It is not necessary to stop the predictor to control and modify components, it can be done while running.

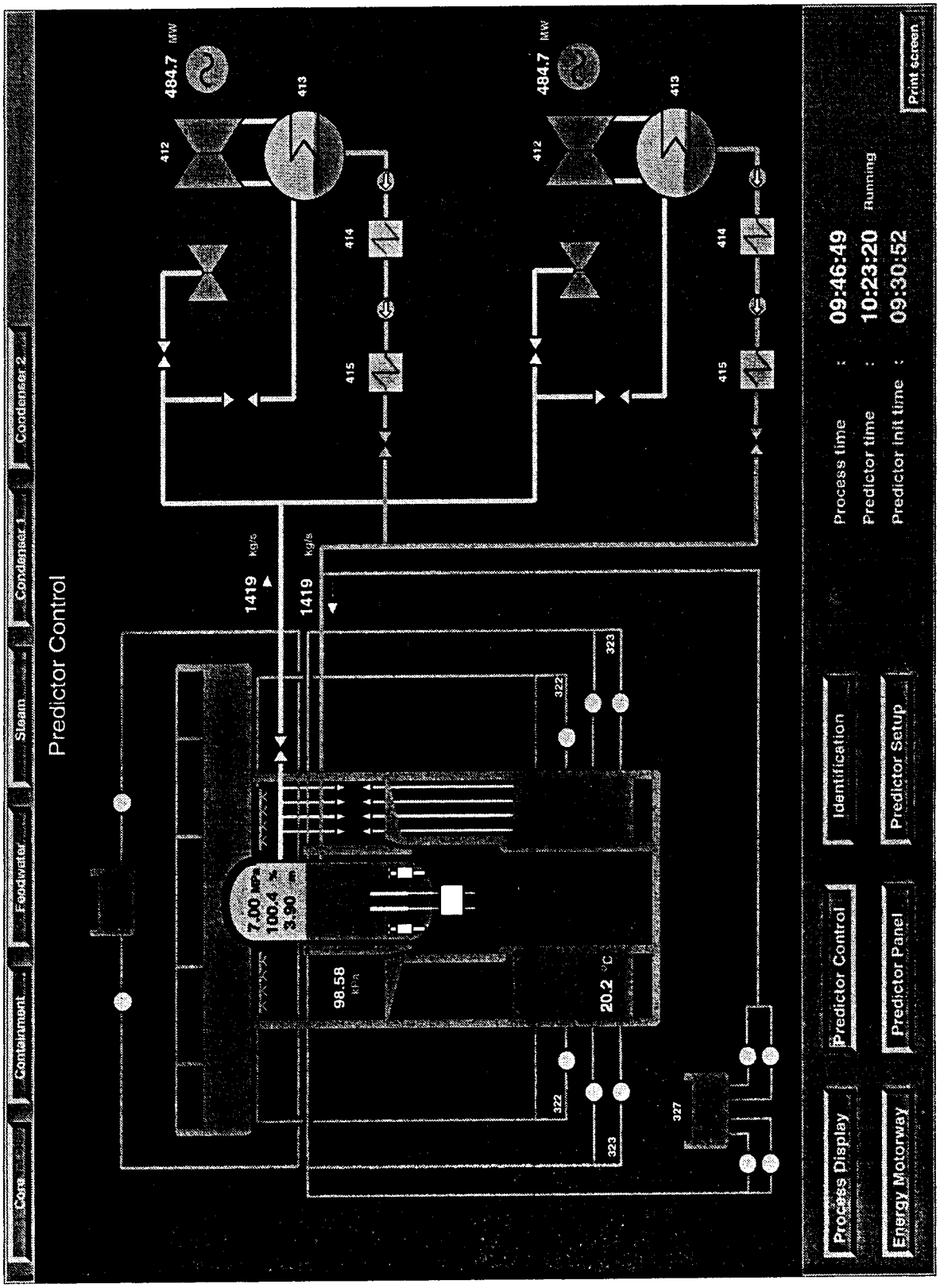

Figure 13. Predictor Control

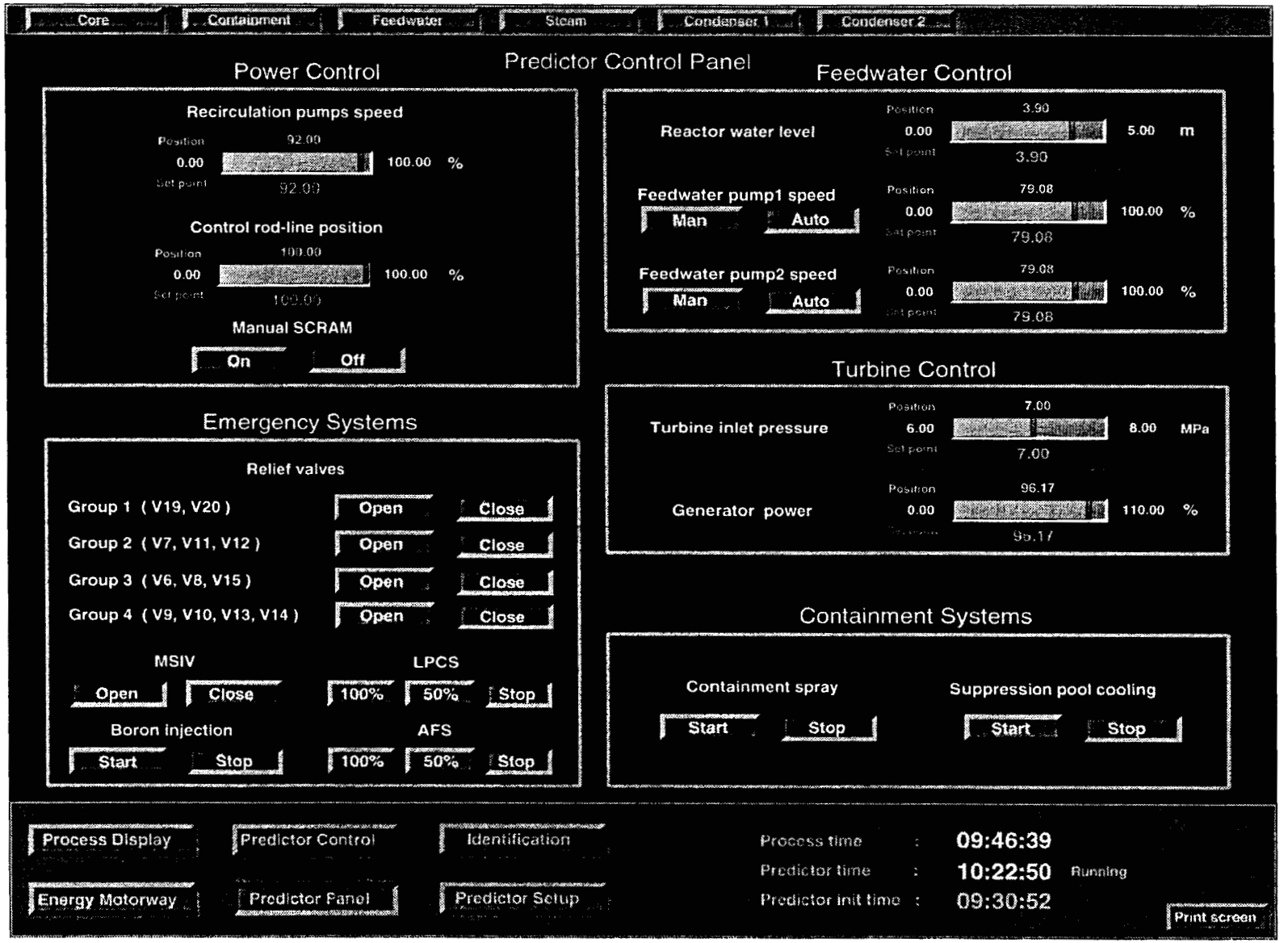

Figure 14. Predictor Panel

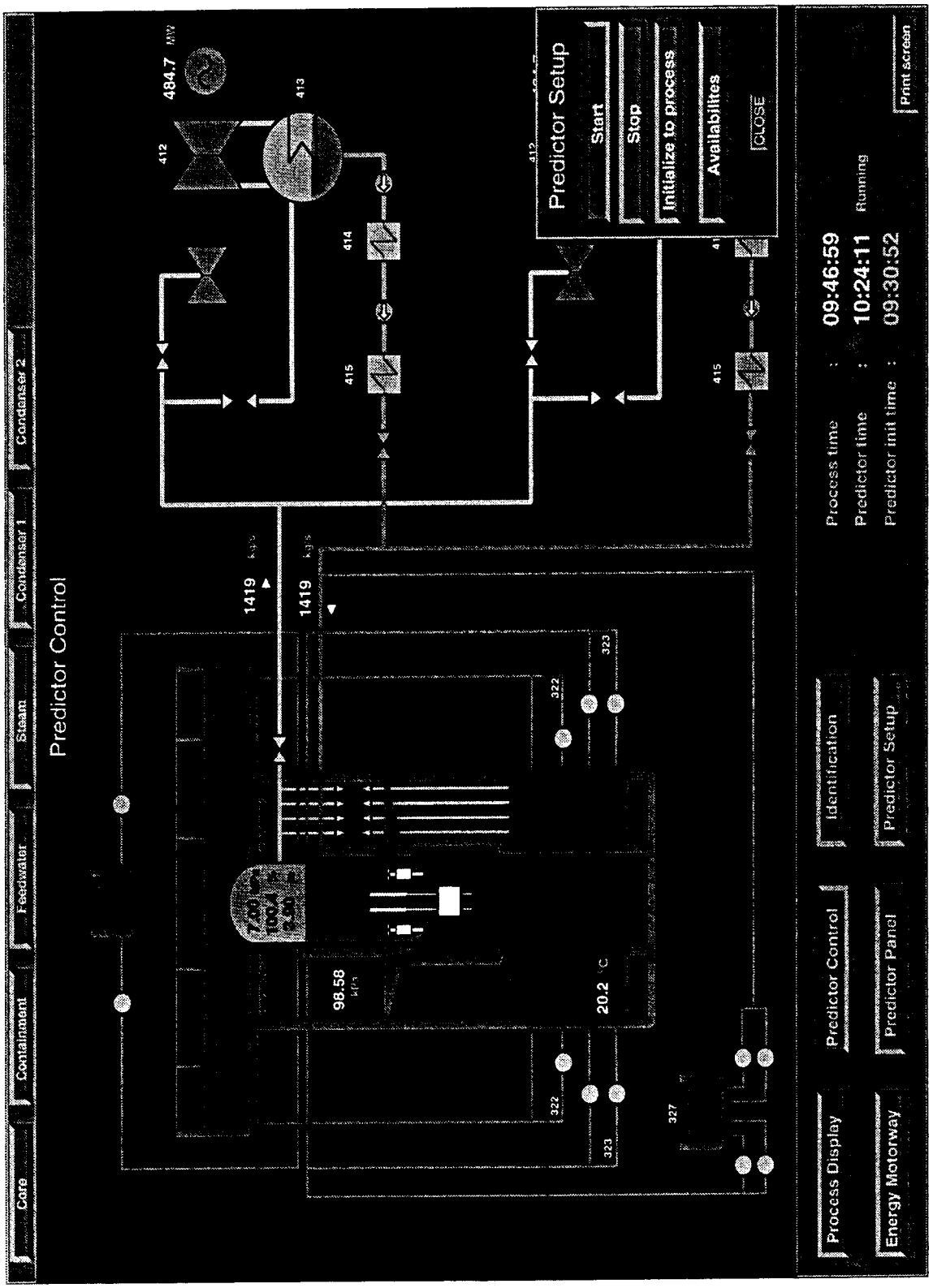

Figure 15. Predictor Setup

By using the new trend system of Picasso-3, prediction trends are superimposed on the historic trends. This approach offers a good visual indication on the evolution of plant versus prediction. History as well as prediction can be shown together (helps to see if a prediction is correct) and there will be click-sensitive areas on the curves (for instance, to bring up the mitigation strategy employed on this particular prediction). This last point is important because it can be used by the user to retrace his actions to an earlier point in the prediction and start from there. Figure 16 shows how we are thinking to display this. Figure 17 shows the trend system as it is today.

APROS [6], developed by VTT, has been chosen as the simulator building tool for CAMS. For a technical description of the model, see Reference [1].

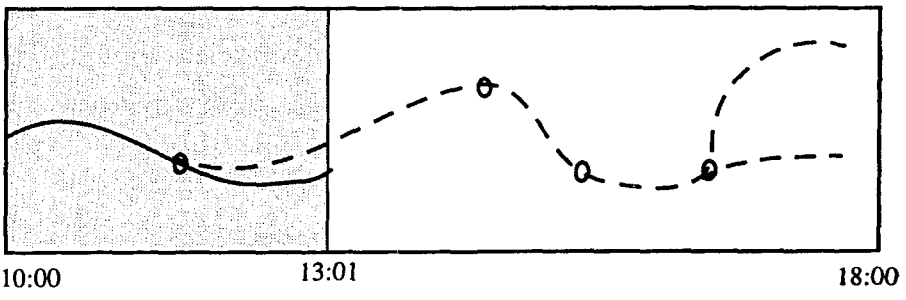

Figure 16. Planned trend diagram.
The present time is the border between dark and light background colour. History to the left (start of prediction is already history). The circles are click-sensitive and indicate where the user has interfered with the controls of the simulator. In the end two different strategies have been tested, with quite different outcome.

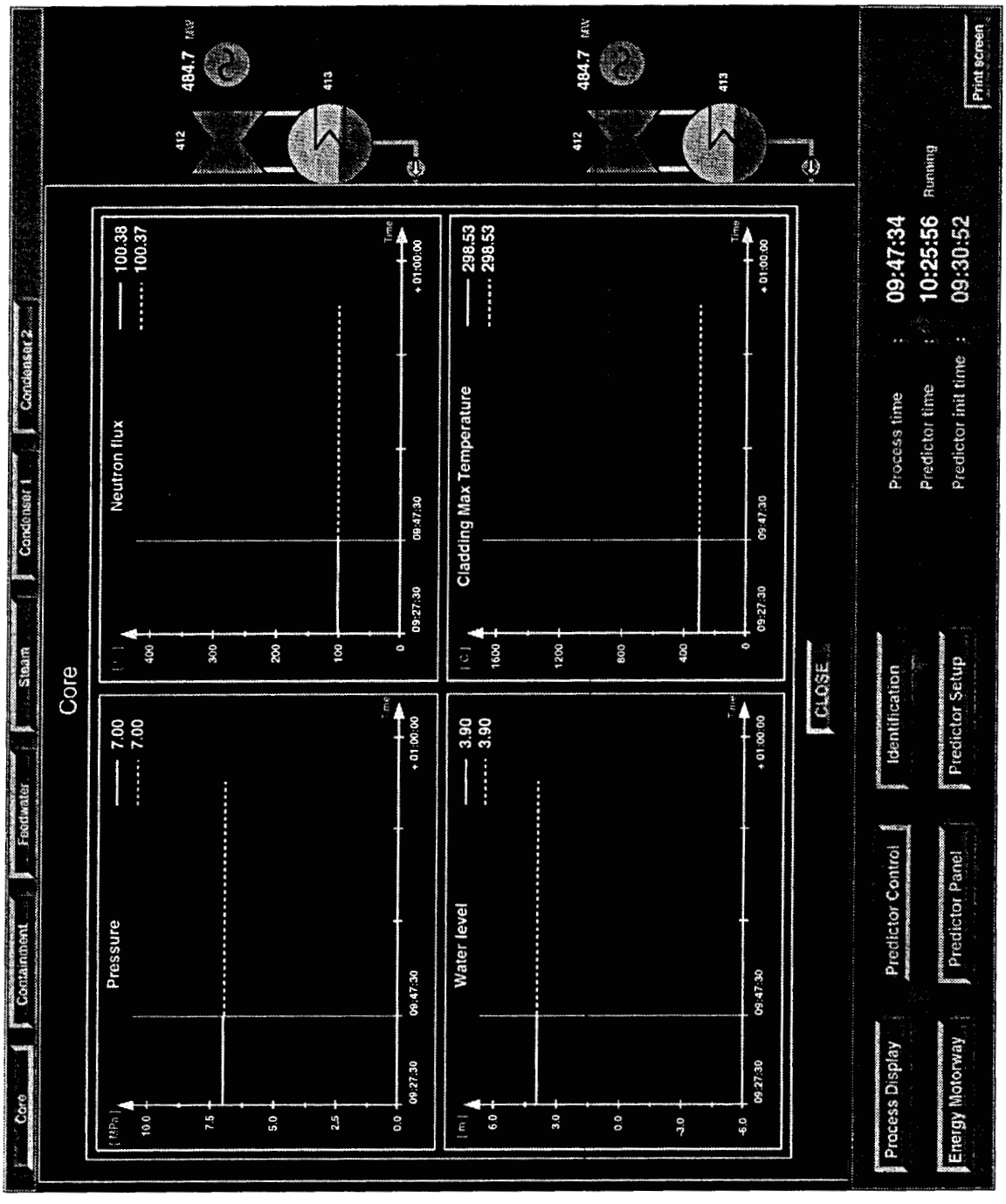

Figure 17. Present trend system.
History is shown to the left of the vertical bar which represents current time. To the right of this bar the future is shown as a dotted line. In the upper right corner of each trend diagram the numerical value of the present time and of the prediction is shown.

9.3 Future development

The current version was made to be a model of the plant with emphasizing simulation of the as-built Forsmark-2, and does therefore not include many extra features to allow the external control of components and systems that are needed for simulation of faults.

For the model to be able to simulate a faulty system or a component, the model needs to be expanded with features that will override the normal automatic control system of the plant. Example; if it is detected from the real plant that a relief valve is stuck, we must tell our model that this particular valve is stuck, set the valve in a given position (to simulate stuck open) and keep it there regardless of the automation system that will try to close or open it.

Some faults needs extra modelling if they are to be simulated. Leakages belongs in this category as they need a fake valve to be inserted where we want the leak.

Availabilities of systems and components is also something that needs expansion.

Some systems are still missing, like shutdown cooling.

10. THE PROBABILISTIC SAFETY ASSESSMENT MODULE (PSA)

10.1 Purpose

The purpose of this PSA module is to provide on-line accident prevention and mitigation strategies for a nuclear power plant (NPP) as one module of the CAMS (Computerized Accident Management Support).

10.2 Description

This module contains plant specific PSA data, comprising event trees, failure probabilities etc. It has several event trees categorized according to the initiating events (IEs). Each event tree has an initiating event frequency and a branching probability. The various support systems for branches are considered and their dependencies are calculated logically.

The *risk* or *Core Damage Frequency* (CDF) is re-calculated based on the current state of the plant and the pre-calculated level 1 PSA. The CDF is relatively low when the plant is operating normally. However, if a component or a system becomes unavailable for one reason or another, say failure or maintenance, the failure probability is changed and the current risk is re-calculated and displayed. This function can be activated by data from the state identification (SI) module of CAMS.

If an initiating event occurs, the event tree is re-calculated and the PSA module shows which systems of the plant that should operate normally. If the plant responds to the event in the normal way, the plant will be shut down and come to a safe state. However, if some functions do not work, the PSA module generates another path and gives the information about the critical systems to the Strategy Generator (SG) module. The new path is checked by the SG and if the state of the plant is changed, either by the operators or automatically by the control system, the PSA module follows the new route.

Not only the operational support is considered. We are also planning to provide support for the maintenance or test condition considering risk or/and cost. For this purpose, a plant availability analysis (PAA) should be performed.

10.3 Basic methods

10.3.1 Event tree model

The first goal of the accident management is prevention of a *Core Damage Accident* (CDA). Therefore, the development of the first version of the PSA module of CAMS focuses on the level 1 PSA, which estimates the CDF. Usually for level 1 PSA, the method of small event trees and large fault trees is used. In the PSA module, we have adopted this method.

The event tree is a sequential accident tree as shown in Figure 18. There are 5 branches, which correspond to safety functions called "front-line systems" in this sample event tree. System failure probability (unavailability) is given to each front-line system. In the level 1 PSA, this unavailability is calculated by fault tree analysis. The fault trees comprise a large number of gates and basic events, which have associated failure probabilities.

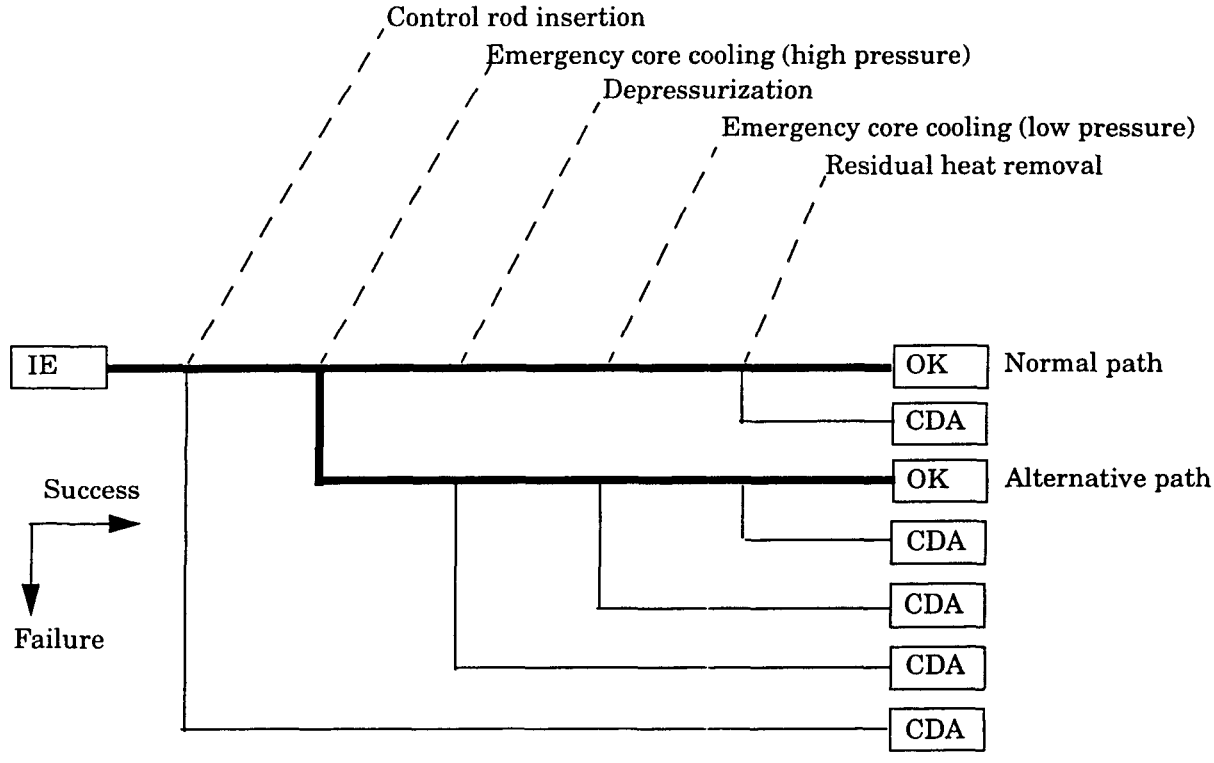

Figure 18. An event tree of the probabilistic safety assessment.
IE = initiating event, OK = safe state, CDA = core damage accident.

The tree branches according to the systems availability, success or failure. The number of end state means the number of sequences. Since not all the branching points produce trees, the number of sequences is less than 2^n (n = number of front-line systems).

The frequency of each sequence is calculated with the failure probabilities of the front-line systems, but if a system fails or succeeds, the probability becomes 1 or 0 respectively. In Figure 18, the success branches go to the right and failures go down. If a system always fails, then the branch goes down only and for the success, goes to the right only. If we know the systems availability in plant, the event tree branches are truncated. Thereby, we can obtain only probable sequences. The

probable paths of the sequences can be checked by the SG or the PS in CAMS, and a better strategy may be obtained.

10.3.2 Front-line system model

The front-line systems have their own fault trees. In the level 1 PSA, the analysis of a fault tree is a time consuming work both for the analyst and the computer. In a living PSA, it is impossible to calculate the complex fault tree very fast, therefore a pre-calculated and simplified fault tree should be adopted. The structure of a simplified fault tree consists of minimal cut sets (MCSs) only. A MCS is a minimized combination of components failure that makes a failure of the system. Moreover, trivial or low probability MCSs are discarded, so that the calculation is fast enough. We think that carefully selected MCSs are accurate enough for the living or dynamic PSA.

10.3.3 Support system model and dependency calculation

For the static PSA, every sequence is calculated as one big fault tree. The system dependencies are calculated automatically by the fault tree analysis code. However, this calculation needs much time even with a fast computer. In many cases, support systems affect several front-line systems, for example, a power supply system or a cooling water system. If these support systems fail, one or more front-line systems become unavailable.

In many cases, the support systems are extracted from the front-line systems, and the front-line systems are divided into truncated parts and support system parts. If a support system fails, the failure probability of the front-line systems becomes 100%. But if the support system succeeds, the failure probability of the front-line systems still exists. Moreover, the status of a support system influences other front-line systems. Therefore, we should calculate the contribution of every support system for every front-line system.

For instance, suppose there are four front-line systems, F_1, F_2, F_3 and F_4, and two support systems S_1 and S_2. Moreover, say if S_1 fails, F_2 and F_3 fails, and if S_2 fails, F_3 and F_4 fails. The truncated part of each front-line systems are T_1, T_2, T_3 and T_4. The Boolean equations are as follows:

$$F_1 = T_1 \tag{2}$$

$$F_2 = T_2 + S_1 \tag{3}$$

$$F_3 = T_3 + S_1 + S_2 \tag{4}$$

$$F_4 = T_4 + S_2 \tag{5}$$

Suppose a Boolean equation of a sample sequence $A=F_1\neg F_2F_3F_4$ (F_1, F_3 and F_4 fail but F_2 succeeds). The logical calculation gives

$$A = T_1T_3T_4\neg T_2\neg S_1 + T_1S_2\neg T_2\neg S_1 \ . \tag{6}$$

The failure probability of this equation is expressed by the following equation because probability of negation equals nearly 1 in most cases.

$$P(A) = P(T_1)P(T_3)P(T_4) + P(T_1)P(S_2) \tag{7}$$

Here, the term $P(T_1)P(T_3)P(T_4)$ is the contribution of only the truncated part of the front-line systems, and $P(T_1)P(S_2)$ is a contribution from the support system S_2, because if S_2 and T_1 fail, A fails. However, the contribution of support system S_1 is omitted, because the success of F_2 means the success of S_1. Therefore, the term S_1 cannot contribute to A in this case.

Generally, one front-line system is represented as follows:

$$F_i = T_i + \sum_{j=1}^{N_s} R_{ij} S_j . \quad (8)$$

Here, F_i is a Boolean expression of each front-line system, T_i is a main body of the front-line system, S_j is each support system and R_{ij} is a matrix that expresses the relationship between the front-line systems and the support systems (the value is true or ø). Moreover, each T_i and S_j is a combination of several MCSs.

One sequence is a combination of front-line systems as follows:

$$A_k = \prod_{i=1}^{N_F} C_{ki} F_i , \quad (9)$$

where A_k is a Boolean equation of each accident sequence (not safety state sequence) and C_{ki} is a matrix that combines front -line systems and accident sequences (the value is true, false or ø). The probability of each accident sequence is calculated according to this Boolean equation.

One event tree has several accident sequences. The CDF of each event tree E_n is given by the following equation, where I_n is an initiating event frequency:

$$E_n = I_n \sum_{k=1}^{N_A} P(A_k) . \quad (10)$$

Approximately 10 event trees are used for a normal PSA. The sum of all the CDFs gives the final result.

10.3.4 MCS model and basic events

Each system has more than one MCS. Each MCS is a multiplication of failure probabilities of basic events. Different MCSs can have the same basic events, but the same MCS must not appear in other front-line or support systems in our simplified method. Otherwise, failure probabilities are miscalculated because the dependency between systems becomes inconsistent.

The basic event has its own failure probability. The failure probability is calculated according to the component status and failure rate. For this calculation, mission time, test interval and other data are required. In the living PSA, these data are changing dynamically.

10.4 Functions of the PSA module

The PSA module calculates the core damage frequency using the latest status of the plant periodically (the period is 10 seconds now). The status of each safety functions is obtained from the SI module. The module has a calculation part implemented in the C++ language and a display part made with Picasso-3. The user can interact with the module, i.e. the system status can be modified temporarily. However, the data from the SI override the user input.

Important information is displayed on the screen, see Figure 19, as described here.

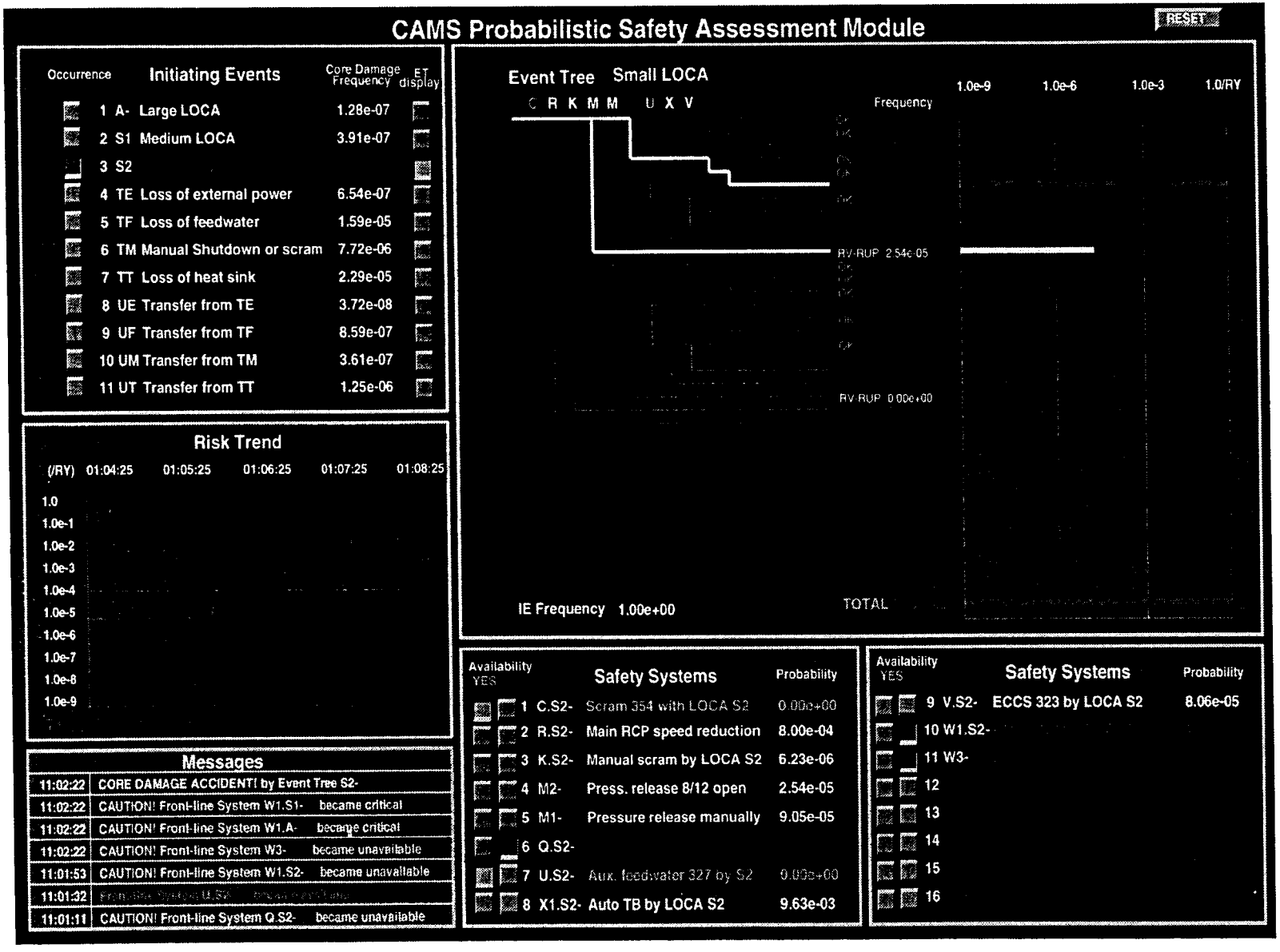

Figure 19. The PSA Console

Initiating event list

A list of initiating events with names and CDFs is shown on the screen. If the user inputs *Occurrence of the initiating events*, the CDF is re-calculated.

Front-line system list

On this list, names and failure probabilities of the front-line systems of the selected event tree are displayed. The user can change the availability of the front-line systems temporarily.

Event tree display

If the user chooses one initiating event, the module shows the event tree diagram and the CDF bar chart. If some systems are unavailable (failure probability equals 1) or completely available (failure probability equals 0), only the probable paths are shown with thick lines. Therefore, you can easily find the success path.

Risk trend

The user can input the initiating event occurrence and systems availability arbitrary or the SI module may change them. The trend screen shows the history of the CDF according to the changes.

Message screen

The message screen shows failed systems, completely available systems and critical systems. If a critical system fails, there is no way to avoid the core damage accident when the related initiating event occurs.

10.5 Future development

The functions of the current version of the PSA module are still limited and only the basic methods are available so far. The plan is to extend the module as described below.

10.5.1 Fault Tree Analysis

This module presently uses only system failure probabilities for the calculation of the CDF. If we want more detailed information, for example a maintenance method for one particular component, we need to use component level data as well as system level data. For this purpose, the FTA (Fault Tree Analysis) and the MCSs calculation are required.

If some information of a component is transferred from the plant or the SI module, the probability of the related system is modified. The calculation is performed with the fault tree of the systems. The fault tree should have only MCSs to keep the calculation speed fast enough.

A MCS is a combination of basic events which have failure probabilities. Most of the basic events are divided into two categories as follows.

Stand-by component event

The failure probability of a stand-by component is given by demand failure rate or failure rate and test interval. The probability changes according to the elapsed time after the test. If the component fails or becomes unavailable, the probability becomes 100%.

Running component and mission time calculation

The failure probability of a running component is given by failure rate and mission time (required operation time of the system for the accident situation at hand). When an initiating event occurs, the mission time decreases if the component keeps running. If the component fails or becomes unavailable, the probability becomes 100%.

The data for calculation of each basic event are changing according to the plant status.

10.5.2 Connection with plant data

The failure rates are given by some generic data base of the PSA in the past or other experiences. But in the real plant, the failure rate data must be changed by the plant specific-failure data.

Moreover, we need on-line component status for real-time estimation of CDF. The available data of components is limited in the real plant. We think some functions are necessary, that identify the failed components judging from a few data. In CAMS, we have the State Identification (SI) module. This will help in the identification of failed components.

Furthermore, operation data like tests or maintenance works affect the failure probability. These data should be linked with the PSA module. If on-line data is unavailable, the user should input the data in this module.

10.5.3 Operational Support

One purpose of Living PSA is to assist operators and maintenance staff in the nuclear power plants. The system should provide suggestions about *What to do next.*

Success path searching and operational support

The PSA module shows the probable path of sequences. This means which systems or components should normally work. Therefore the operator knows what operation should be taken. If the event trees are made based on the EOP of the plant, the system can work as an operational guidance system during the emergency. However, the order of operations or guidance timing should be well organized.

Importance calculation of systems and components

The operator and the maintenance staff should know which systems or components are crucial to the safety, during normal operation or emergency situations. The system should show the quantitative importance of the systems or components. The possibility of automatic display of the importance of systems should be studied. Such a function will show the important systems and components in an emergency situation, such that the operator can keep his attention on the most important tasks.

Maintenance and testing suggestions

During normal operation, it is important to minimize the plant risk. The maintenance and surveillance tests affect the risk remarkably. In many cases, during maintenance or testing, the component becomes unavailable, so that the risk increases. However, if the component returns to the normal state, the risk is reduced compared to the state before the maintenance or the test. In this way, we can plan the best interval of maintenance and testing to minimize the CDF. Moreover, we should consider maintenance and testing cost. For this purpose, we should perform cost analysis using CDF and other factors.

10.5.4 Combination with Plant Availability Assessment (PAA)

Minimizing the maintenance cost of a NPP is crucial, because the maintenance of NPPs is expensive. One reason for high cost of maintenance is safety. We should perform cost-benefit analysis as mentioned, but this is not enough. In normal operation, the most important factor relevant to cost is not safety but plant availability, because a plant shut-down means a big loss of income. From our experience, this factor is 1000 times larger than the cost caused by the safety factor. So we should execute another fault tree analysis called Plant Availability Assessment (PAA), see Reference [14]. The combination of PSA and PAA will give a well balanced proposal of a maintenance plan.

10.5.5 Other tasks

Low power and shut down risk analysis

Some studies show that the risk at low-power and shut-down states is relatively high. We should take into account these factors in the future. A workshop is planned on this subject in the autumn of 1996.

Consequence analysis

In this study so far, we have used CDF as a risk indicator. But we should also consider the consequence of an accident. Therefore the results of level 2 and 3 PSA should be taken into account in the future, in addition to CDF.

Uncertainty analysis

The PSA has various kinds of uncertainty. We should consider the uncertainty of CDF, but it is in many cases difficult in an on-line system, because the uncertainty analysis sometimes requires time-consuming calculation like the Monte Carlo Method.

11. CONCLUSION

This report shows the actual status of the CAMS project. The first phase of the project (1992 - 94), which is described in Reference[1], [2], and [3], focused mainly on the following tasks:

- information needs during normal and accident conditions in a NPP,
- methods that can be successfully applied to a CAMS system,
- MMI and human factors requirements.

In the second phase of the project the development of those ideas into a working prototype has begun. The prototype is by no means completed yet. Its main purpose is to test those methods in a simulated environment, to verify that the many developed functions, using different techniques, can work together producing the desired result in an efficient way. In other words, this prototype can be considered a test platform to do the following:

- Develop and integrate modules like Tracking Simulator, Signal Validation, State Identification, PSA, and Predictive Simulator. Evaluate how each task is performed, and identify advantages and drawbacks in the methods used.
- Design a functional structure able to coordinate and synchronize the activities that take place in all those modules. The overall output of the CAMS system should then be able to satisfy the information needs requirement mentioned above.
- Design and test an appropriate user interface for different kinds of users, with different needs and knowledge.

During the CAMS design, a considerable effort has been made to maintain the generality of the CAMS concept; although the referenced process has so far been a BWR plant, the use of this structure and design can be applied to other processes, including non-nuclear processes. An important feature of the system, in this respect, is that all the functions or modules (TS, SI, SV, etc.) are completely independent of each other and that modules can be deleted, added or changed without affecting the rest of the system. Moreover, the external tools here used (APROS, GPS, Picasso-3, etc.) are just plugged into CAMS, so that other tools can be easily used, depending on the application. This solution is also in line with the "testing platform" concept cited above, where the need to test different solutions for a single module within the general CAMS framework (with other functions turned off) can be anticipated.

The development phase of CAMS (phase 2) is in a fairly early stage. Some modules are in a more advanced condition than others and even the general structure controlled by the System Manager

is still evolving and is not completed. Nevertheless, the prototype in its current state demonstrates that the direction in which the CAMS project is heading is promising, and that the prototype is a useful tool for studying important aspects of accident management.

12. ACKNOWLEDGEMENT

APROS was received from VTT and IVO (Finland). They have also provided excellent support. A large amount of PSA data for the Forsmark-2 plant was received from Forsmarks Kraftgrupp and Vattenfall (Sweden). EdF (France) has participated in technical discussions. SKI (The Nuclear Inspectorate, Sweden) has provided data for the Barsebäck plant, and they provided staff to evaluate the predictive part of the system. Tractebel (Belgium) has permitted us to use their tool GPS.

We are also grateful to VTT, IVO, PNC (Japan), and Tractebel, who have sent delegates working on the project.

We have also received good help from an advisory group during the early parts of the CAMS project.

We are also grateful to Miki Sirola (VTT), Pekka Nurmilaukas (IVO), Alessandro Mazzola (Politecnico di Milano), and Takashi Iijima (PNC), in addition to several HPR staff members, for their important contribution in earlier phases of the project.

13. REFERENCES

[1] Paolo Fantoni, Geir Meyer, Pekka Nurmilaukas, Miki Sirola, Aimar Sørenssen:
Description of the computerized accident management support system (CAMS) prototype and design.
HWR-390. Halden, October 1994.

[2] Lars Berglund, Paolo Fantoni, Magnhild Kaarstad, Michael Lindström, Geir Meyer, Jan Olsén, Aimar Sørenssen:
The use of CAMS during a safety exercise at the Swedish nuclear inspectorate.
HWR-423. Halden, July 1995.

[3] Øivind Berg, Paolo Fantoni, Yukihiro Iguchi, Takashi Iijima, Geir Meyer, Svein Nilsen, Pekka Nurmilaukas, Aimar Sørenssen:
On-line simulation and estimation: Lessons learned for CAMS.
HWR-429. Halden, December 1995.

[4] Takashi Iijima:
Application study of fuzzy logic methods for plant-state identification.
HWR-432. Halden, December 1995.

[5] G2 is a trademark of Gensym Corporation.

[6] APROS has been developed by IVO and VTT in Finland.

[7] Matlab is a trademark of The MathWorks, Inc.

[8] M. De Vlaminck, E. Martin, C. Van Dyck, F. Leboutte, F. de Viron:
A knowledge-based system to support nuclear power plant operators.
EPRI conference, December 1993.

[9] K. A. Barmsnes, A. Hornæs, Ø. Jakobsen, R. Storkås:
Picasso-3 system design.
HWR-288. Halden, May 1991.

[10] Tord Akerbæk, Michael Louka:
The software bus, an object-oriented data exchange system.
HWR-446. Halden, April 1996.

[11] ORBIX is a trademark of IONA Technologies Ltd.

[12] Paolo F. Fantoni, Neuro-Fuzzy Models Applied to Full Range Signal Validation in Nuclear Power Plants, NPIC&HMIT '96, The Pennsylvania State University, May 6-9, 1996.

[13] Idaho Nat. Engineering Lab., EG&G Idaho Inc.:
Accident Management Information Needs.
NUREG/CR_5513, EGG-2592 Vol. 1.

[14] Yukihiro Iguchi, Masutake Sotsu, Kazutoshi Isomura:
PSA related activities and an application to the maintenance in Fugen.
ICONE-3, April 1995, Kyoto, Japan.

Appendix I

The estimation problem in tracking simulation

The estimation problem is usually formulated about as follows.

Let the state variables $x_1, x_2, \ldots, x_N$ be collectively denoted by x. An arbitrary state variable is denoted by x_n. The controls $u_1, u_2, \ldots, u_M$ are collectively denoted by u, and an arbitrary one of them is denoted by u_m. The state variables at the time $t = k\Delta t$ are denoted by $x(k)$. Here, Δt is the time between measurements. Their values at $t = (k+1)\Delta t$ can be written as

$$x(k+1) = \Phi[x(k), u(k)] + w(k). \tag{11}$$

The functions Φ contain integrals of the differential equations of the problem. $w(k)$ describes the process noise, which we assume to be a Gaussian random variable with zero mean and covariance Q:

$$w(k) \propto N(0, Q) \tag{12}$$

The measurements are not necessarily the states themselves, but functions of the states:

$$y(k) = C(x(k)) + v(k). \tag{13}$$

Here $y(k)$ denotes the set of measurements $y_1, y_2, \ldots, y_L$. An arbitrary member is denoted by y_l.

The functions C can be regarded as the spectacles through which we watch the state. $v(k)$ describes the measurement noise, which we assume to be a Gaussian random variable with zero mean and covariance R:

$$v(k) \propto N(0, R) \tag{14}$$

We shall only work with cases where C picks out a subset of the states, even though the theory may be developed for the more general case.

Note that the state vector x is permitted to include constant parameters. It is up to us whether to include a non-changing quantity in the state vector, or not. Let us imagine that all quantities in the model are included in the state vector.

Assume we have estimates $\hat{x}$ of the state variables at $t = k\Delta t$. These estimates we denote $\hat{x}(k)$. We shall distinguish between two types of such estimates: those based only on the information available at $t = (k-1)\Delta t$, which we shall call $\hat{x}(k-\varepsilon)$, and those that also take the information available at $t = k\Delta t$ into account, which we shall call $\hat{x}(k+\varepsilon)$.

For a linear system, the estimates follow the same equations of motion between measurements as the states themselves, apart from noise. This is not strictly true for a nonlinear system, but we shall assume that it holds with sufficiently good approximation. We put

$$\hat{x}(k+1-\varepsilon) = \Phi[\hat{x}(k+\varepsilon), u(k)]. \tag{15}$$

From the estimates of the states we calculate the estimates of the measurements:

$$\hat{y}(k+1-\varepsilon) = C(\hat{x}(k+1-\varepsilon)). \tag{16}$$

This transformation from just after the measurement at $t = k\Delta t$ to just before the measurement at $t = (k+1)\Delta t$ is performed by the modelling tool, in our case APROS. The dynamics model essentially works out Φ and C. In order to obtain sufficient accuracy, the model will have to use a fine nodalization, which means that there will be many more internal state variables x in the model than there are measured quantities y in the plant.

Then the measurements at $t = (k+1)\Delta t$ arrive, and we shall use them to transform $\hat{x}(k+1-\varepsilon)$ to $\hat{x}(k+1+\varepsilon)$. First we work out the so-called *residuals* or *innovations*

$$r(k+1) = y(k+1) - \hat{y}(k+1-\varepsilon). \tag{17}$$

There will be L residuals, one for each measurement. They represent what is learned by the measurements, the amount of surprise. We use this knowledge to update some of the estimates. Then we put

$$\hat{x}(k+1+\varepsilon) = \hat{x}(k+1-\varepsilon) + K(k+1)r(k+1). \tag{18}$$

The gain matrix K is not necessarily diagonal. The estimate $\hat{x}_n$ of the state variable x_n may be modified also by the measurement y_l of another state variable $x_{n'}$. In component form we may write

$$\hat{x}_n(k+1+\varepsilon) = \hat{x}_n(k+1-\varepsilon) + \sum_{l=1}^{L} K_{nl}(k+1)r_l(k+1). \tag{19}$$

Here y_l is the measurement of $x_{n'}$, where $n' \neq n$.

For reasons given in Chapter 7., we shall have to find a method that is simpler than the Kalman filter.

Equation of motion for variables and parameters

Non-measured and measured variables are things like temperatures, pressures, positions, velocities, quantities for which we can establish equations describing the main part of their development with time, even though our knowledge is not precise. A quantity of the first type may typically be described by an equation of motion like for instance

$$x_{position}(k+1) = x_{position}(k) + x_{velocity}(k)\Delta t + w_{position}(k). \tag{20}$$

The noise term $w(k)$ describes our lack of knowledge of the details of the motion.

Updated parameters are quantities that normally do not change, or at least change only slowly, like lengths and cross sections of pipes, the thicknesses of steel walls and electric resistances. The existence or non-existence of a leakage also is included in this category. These latter quantities, which we often call "constants", may also change, but we are not able to predict how, or we decide not to do so even if we could.

We shall assume that a parameter does not jump wildly around, it displays some sort of continuity. Rather than making the parameter directly a Gaussian stochastic variable, we make its change between one measurement and the next a Gaussian stochastic variable:

$$x_{parameter}(k+1) = x_{parameter}(k) + w_{parameter}(k). \tag{21}$$

This equation does not mean that parameters cannot change, only that we cannot predict how it will change. The entire time development of these quantities is described by the noise term.

The classification into variables and parameters is not always obvious. If we for example enlarge our description of flow phenomena to also include the corrosion of a steel pipe, and establish equations for the change of the wall thickness with time, the wall thickness will be a variable rather than a parameter.

The variables will change in such a way that certain conservation laws are respected, when all such quantities are regarded together. If the amount of water increases in a tank, the amount of water will decrease in another tank, in such a way that the total amount of mass is conserved. Or

it may turn up as a change in the amount of steam, again in such a way that the total mass is conserved. A conservation law usually involves masses, momenta, or energies. Parameters, on the other hand, describe things like lengths and cross sections of pipes, friction factors, resistances and similar things and they usually do not obey conservation laws. We conclude that it is easier to update parameters than variables. We shall use measurements of variables to adjust parameters so that the model fits the measurements.

We introduce the following notation. We arrange all measured variables $y_1, y_2, \ldots, y_L$ into a vector y, an arbitrary one of these measured variables is denoted by y_l. Similarly, we arrange all non-measured variables $z_1, z_2, \ldots, z_M$ into a vector z, an arbitrary one of these non-measured variables is denoted by z_m. Finally, all parameters $p_1, p_2, \ldots, p_N$ that shall be updated are arranged into a vector p, an arbitrary parameter is denoted p_n. The complete state vector x is then the collection of all these:

$$x = \begin{bmatrix} y \\ z \\ p \end{bmatrix}. \tag{22}$$

Here, $1 \le l \le L$, $1 \le m \le M$, and $1 \le n \le N$. The number of adjustable parameters N should preferably be much smaller than the number of measurements L.

The equation of motion for the estimates, Equation (15), is rewritten into component form:

$$\hat{y}_l(k+1-\varepsilon) = \Phi_l[\hat{y}(k+\varepsilon), \hat{z}(k+\varepsilon), \hat{p}(k+\varepsilon), u(k)], \tag{23}$$

$$\hat{z}_m(k+1-\varepsilon) = \Psi_m[\hat{y}(k+\varepsilon), \hat{z}(k+\varepsilon), \hat{p}(k+\varepsilon), u(k)], \tag{24}$$

$$\hat{p}_n(k+1-\varepsilon) = \hat{p}_n(k+\varepsilon). \tag{25}$$

We do not have a priori information to predict how the parameters p develop between $t = k\Delta t$ and $t = (k+1)\Delta t$, therefore the estimates $\hat{p}$ are unchanged until we receive new experimental information.

The residual equation, Equation (17), is

$$r_l(k+1) = y_l(k+1) - \hat{y}_l(k+1-\varepsilon) \tag{26}$$

as before. $z(k+1)$ and $p(k+1)$ are not measured, and there are no residuals corresponding to them.

And finally the update equation, Equation (18), is rewritten as

$$\hat{y}_l(k+1+\varepsilon) = \hat{y}_l(k+1-\varepsilon), \tag{27}$$

$$\hat{z}_m(k+1+\varepsilon) = \hat{z}_m(k+1-\varepsilon), \tag{28}$$

$$\hat{p}_n(k+1+\varepsilon) = \hat{p}_n(k+1-\varepsilon) + \sum_{l=1}^{L} K_{nl}(k+1) r_l(k+1). \tag{29}$$

Equations (27) and (28) mean that we do not update any variables, neither measured nor non-measured, when new measurements arrive. The experimental information is used for updating the parameters.

Determination of the gain matrix

We want to minimize a cost function of the least-squares type. The sum of squared residuals

$$J(k+1) = \sum_{l=1}^{L} [r_l(k+1)]^2 \tag{30}$$

will not do. Some terms in this sum may have the dimension m^2, other terms may have the dimension $degrees^2$, and so on, and it does not make sense to add their numerical values. A better choice is

$$J(k+1) = \sum_{l=1}^{L} [a_l r_l(k+1)/\sigma_l]^2 . \tag{31}$$

Here, σ_l is the standard deviation of measurement number l. a_l is an attention factor, we may for instance want to put little emphasis on a certain measurement even though it is very accurately measured, or much emphasis on something we feel is particularly important.

We shall minimize the cost function by a proper choice of estimates $\hat{p}_n$ of the adjustable parameters p_n.

Suppose that we have estimates of y, z, p at $t = k\Delta t$ based on the information available at $t = k\Delta t$. In the notation we have used these are denoted $\hat{y}(k+\varepsilon)$, $\hat{z}(k+\varepsilon)$ and $\hat{p}(k+\varepsilon)$ respectively.

Then the measurements for $t = (k+1)\Delta t$ arrive. We then predict y and z at $t = (k+1)\Delta t$ for several sets of the parameters p, and from the y's we work out the residuals r and the cost function J. Finally we choose the set of y, z, p which has the smallest J. This set we take as our estimates at $t = (k+1)\Delta t$, and we denote them $\hat{y}(k+1+\varepsilon)$, $\hat{z}(k+1+\varepsilon)$, $\hat{p}(k+1+\varepsilon)$.

Distribution:

Denmark:

Beredskabsstyrelsen
attn: Björn Thorlaksen
Louise Dahlerup
P.O.Box 189
DK-3460 Birkeröd
Denmark

Forskningscenter Risö
attn. Povl Ölgaard
Frank Höjerup
Knud Ladekarl Thomsen
Erik Nonböl
P.O.Box 49
DK-4000 Roskilde
Denmark

Finland:

IVO International Ltd
attn: Petra Lundström
Harri Tuomisto
FIN-01019 IVO
Finland

Teollisuuden Voima Oy
attn: Markku Friberg
Heikki Sjövall
Seppo Koski
Risto Himanen
FIN-27160 Olkiluoto
Finland

Finnish Center of Radiation and
Nuclear Safety (STUK)
attn. Lasse Reiman
Timo Karjunen
Juhani Hyvärinen
P.O.Box 14
FIN-00881 Helsinki
Finland

VTT Energy
attn: Lasse Mattila
Risto Sairanen
Jaakko Miettinen
Esko Pekkarinen
Ari Silde
Ilona Lindholm (5 copies)
P.O.Box 1604
FIN-02044 VTT
Finland

VTT Automation
attn. Björn Wahlström (3 copies)
Pekka Pyy
P.O.Box 13002
FIN-02044 VTT
Finland

Iceland:

Tord Walderhaug
Geislavarnir Rikisins
Laugavegur 118 D
IS-150 Reykjavik
Iceland

Norway:

Institutt for Energiteknikk (IFE)
attn: Oivind Berg (10 copies)
Paolo Fantoni
Yukihiro Iguchi
Geir Meyer
Aimar Sörenssen
Claude Van Dycke
P.O.Box 173
N-1751 Halden
Norway

Sverre Hornkjöl
Statens Strålevern
P.O.Box 55
N-1345 Österås
Norway

Sweden:

SKI
Wiktor Frid (5 copies)
S-10658 Stockholm
Sweden

Studsvik Eco & Safety AB
Lars Nilsson (3 copies)
S-61182 Nyköping
Sweden

Vattenfall Energisystem AB
attn. Veine Gustavsson
Klas Hedberg
P.O.Box 528
S-16216 Stockholm
Sweden

Gustaf Löwenhielm (3 copies)
Forsmarks Kraftgrupp AB
S-74203 Östhammar
Sweden

Emil Bachofner
OKG Aktiebolag
S-57093 Figeholm
Sweden

Sven Jacobsson
Vattenfall AB
Ringhalsverket
S-43022 Väröbacka
Sweden

Erik Söderman (2 copies)
ES-Konsult AB
P.O.Box 3096
S-16103 Bromma
Sweden

Kjell Andersson
Karinta Konsult HB
P.O.Box 6048
S-18306 Täby
Sweden

Barsebäck Kraft AB
attn: Christer Palm
Pekka Skogberg
Ulf Soldèus
P.O.Box 524
S-24625 Löddeköpinge
Sweden

Torkel Bennerstedt
NKS
PL 2336
S-76010 Bergshamra
Sweden