

https://dspace.org/

Multilevel flow modelling as a generalized modelling technique

Bracher, Stephen

2016-08-18
http://hdl.handle.net/10539/20893

Downloaded from a DSpace repository. https://dspace.org/

Multilevel Flow Modelling as a Generalized Modelling Technique

Stephen Bracher

A project report submitted to the Faculty of Engineering, University of the Witwatersrand, Johannesburg, in fulfilment of the requirements for the degree of Master of Science in Engineering by course work and a project.

Johannesburg, 1993

Declaration

I declare that this project report is my own, unaided work. It is being submitted for the Degree Of Master of Science in Engineering at the University of the Witwatersrand, Johannesburg. It has not been Submitted before for any degree or examination in any other University.

S Bracken

8 day of AUGUST 1993

Abstract

With the increasing size and complexity of industrial plants there is a growing need for modelling techniques and tools that can be used to aid the operator in the daily running of the plant. This research investigates a modelling technique, known as multilevel flow modelling (MFM) and assesses its suitability for fault diagnostic applications. A rule base for measurement validation, alarm analysis and fault diagnosis is developed which is independent of the process structure and the number of sensors used, so as not to lose the generality of the modelling technique. A case study shows that it is feasible to use MFM for fault diagnosis and that this technique has the flexibility to accommodate alterations.

In memory of my grandfather

William Redhead

29 January 1918 - 8 August 1991

Acknowledgements

I would like to thank the following people for their support throughout my studies:

Professor Ian Macleod for his encouragement and guidance, and for creating a relaxed but creative and productive atmosphere in which this research was conducted.

My parents for all their help and support.

Natalie Sutton for the many times she proof read this project report.

The University for its financial assistance.

Contents

Declaration ii

Abstract iii

Acknowledgements v

Contents vi

List of Figures and Tables ix

Chapter 1 Introduction 1

1.1 Statement of the Problem....2

1.2 Related Work....3

1.3 Design Tool....5

1.4 Research Approach....6

1.5 Overview....7

Chapter 2 Process to be Modelled 9

Chapter 3 Multilevel Flow Modelling 12

3.1 Implementation of Flow functions in G2....14

3.2 Modelling the Plant with MFM....16

Chapter 4 Fault Diagnostic System 19

4.1 Measurement validation....20

4.2 Alarm Analysis....23

4.3 Fault Diagnostics....25

4.4 MFM Process Management....28

4.5 Applying MFM to Other Plants....29

Chapter 5 Conclusion 30

5.1 General....30

5.2 Assessment....31

5.2.1 G2 and Implementation Of MFM....31

5.2.2 MFM....32

5.2.3 Limitations and Significance....33

References 34

Appendix A Test Plant 37

Appendix B Attributes of Flow Functions 40

B.1 Mass Flow Functions....40

B.2 Management Flow Functions....42

Appendix C Measurement Validation Extrapolation 43

Appendix D Data Reflection 45

Appendix E Measurement Validation Rules 47

Appendix F Alarm Analysis Rules 51

F.1 Source Rules....51

F.2 Transport Rules....55

F.3 Storage Rules....72

F.4 Calculation Rules....76

F.5 Sink Rules....80

Appendix G Fault Diagnostic Rules 84

G.1 Source Rules....84

G.2 Transport Rules....88

G.3 Storage Rules.....109
G.4 Calculation Rules.....116
G.5 Sink Rules.....123

Appendix H Process Management Rules **127**

Appendix I Reset Rules **129**

Appendix J. Functions Used **131**

J.1 Function: CON.....131
J.2 Function: EXTRAPOLAT.....135
J.3 Function: REFLECTION.....136
J.4 Function: CHECK.....154

List of Figures and Tables

Figure 1.1 Operator and the Plant.....2

Figure 2.1 Physical arrangement of the Plant.....10

Figure 3.1 Basic flow functions & Connection rules.....13
Figure 3.2 Hierarchical Class Structure of MFM flow Functions.15
Figure 3.3 MFM representation of the modelled process.....18

Figure 4.1 Failure Detection Hierarchy.....19
Figure 4.2 Flow Diagram of the Fault Diagnostic process.....21

Table 4.1 Conditions for Fault Recognition.....27

Chapter 1

Introduction.

Many industrial processes are equipped with elaborate control systems which assist the operator in maximizing system performance. With the growing need for reliability and safety, these processes are usually equipped with a large number of sensors, which can activate alarms. In the event of a fault occurring, many alarms may be triggered. A small percentage of these alarms will be directly linked to the actual fault (primary fault) and the remainder are due to consequential effects of the primary fault; the domino effect (secondary alarms). This indicates the importance of having a supervisory and monitoring controller which can distinguish primary from secondary alarms. Alarm annunciation is used to pinpoint breakdowns in the process and to indicate that limits which will affect the final product, are being exceeded.

Fault diagnosis systems will become an important part of control systems where plants may extend hundreds of meters and have thousands of process variables. Due to the actual size of such plants it is unlikely that the operator will have "hands on" experience and detailed knowledge of the entire process. This makes it desirable for the operator to be assisted by a fault diagnostic system.

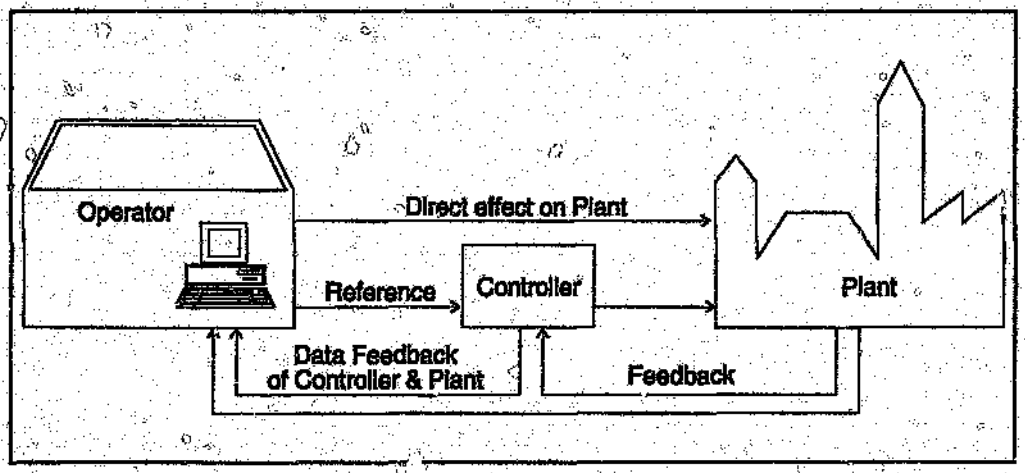


Figure 1.1 Operator and the Plant

The question arises, "Can fault diagnosis be performed with todays fault diagnostic theories and expert systems?". This project report will attempt to answer the question by discoursing fault diagnosis using a simple modelling technique, Multilevel Flow Modelling (MFM), which allows virtually any flow process to be modelled and provides a physical and functional view of the process.

This investigation aims to assess the effectiveness of MFM in performing tasks such as measurement validation, alarm analysis and fault diagnosis as well as generate a set of generic rules to perform these tasks.

1.1 STATEMENT OF THE PROBLEM.

Even the best operator will not always make accurate decisions concerning the process if a measuring instrument failure occurs. An experienced operator may be able to operate the plant for some time by ignoring some measurements and relying on his knowledge

of the past history of the plant. This method of operation however, cannot continue indefinitely and could lead to more serious failures, since the operator may oscillate between maximum and minimum limits of the plant. This causes strain to the equipment which could result in breakdowns which in turn, would increase the running cost of the plant.

In the Three Mile Island nuclear accident for example, the offending pressure relief valve which was stuck open indicated in the control room as being closed. This was because the valve indication showed only whether the valve operating solenoid was energized or not, and did not truly reflect the position of the valve[11]. This indicates the importance of monitoring all critical parts of the process.

An investigation is performed into Multilevel Flow Modelling (MFM) with the goal of determining if MFM could be used as a generalised modelling technique which could be applied easily to other processes and a generic rule base generated that could be used for measurement validation, alarm analysis and fault diagnosis on any flow process.

1.2 RELATED WORK.

Rolf Isermann [6] provides a survey of fault detection, with emphasis placed upon the monitoring of immeasurable quantities. Use was made of process models, parameters, state estimation and statistical decisions, resulting in an unique, mathematically complex solution.

Gertler [17] investigates and presents a survey of techniques to detect and isolate failures such as sensor based actuator malfunction, leaks and equipment deterioration in complex technological systems. Techniques such as analytical

redundancy, statistical testing and signature analysis were used. The major parts highlighted were: quality of the failure detection and isolation algorithms, isolatability, sensitivity and robustness. No consideration was given to a most important criteria of modern design, flexibility.

Paul M. Frank [9] discussed an analytical redundancy approach to fault detection and isolation (FDI) and points out that the case where only poor or imprecise analytical models are available, the model-based FDI approach is still problematical. It was concluded that "FDI is primarily a question of the quality of the available mathematical model of the system." This is not entirely true. The quality of the model used is very important, but it does not have to be mathematically intensive to be an effective modelling technique.

Danai and Chin [2] investigate fault diagnosis by representing the fault signature by a column of a multi valued influence matrix (MVIM) and use adaptations to cope with fault signature variability. MVIM is a commendable method for fault diagnosis when inadequate statistics preclude the use of a statistical pattern classification technique since it does not require any knowledge of the probabilistic structure of the systems. This proves that the ability to perform fault diagnosis is not entirely dependent on the quality of the mathematical model.

Medlock and Furness [14] paper gives a detailed and comprehensive review of the technique of mass flow measurements giving consideration to direct and indirect measurements.

Neil P. Piercy [7] describes a process of selecting filter residuals in order to detect and identify sensor failures. A valid generalised technique for sensor validation for a single class of sensor failure was supplied.

F.G. Shinskey [8] addresses the needed for a standard and proposes a series of tests which may be used to compare one controller against another. Shinskey does not put forward a standard against which performance can be judged since, most controller designs are unique and therefore, cannot be compared.

In all the papers considered there is no mention of whether fault diagnostic theory has reached a standard where it can be freely applied in industry. The majority have put forward complicated, unique solutions which are not easily understood, the result being that the theories have not been put into practice by plant engineers.

1.3 DESIGN TOOLS.

A commercial real time expert system called Gensym G2 (G2) [19] is used, since G2 is a commercially available expert system used in industry, allowing for technology transfer, bridging the gap between academic research and the requirements of industry. G2 is used to simulate the process and develop the multilevel flow modelling technique, with a generic rule base.

G2 is a tool for developing real time expert systems for complex applications that require continuous and intelligent monitoring, diagnosis and control [19].

G2 has the following capabilities:

* Scan an application and focus on key areas
* Reason about and control events in a continuously changing environment
* Respond to events when they occur
* Apply both procedural knowledge and rule-based heuristics
* Express and make use of relationships between objects

A high-level language is provided for writing rules, functions and procedures in a graphical environment.

1.4 RESEARCH APPROACH.

The aim of this investigation is to determine how effective Multilevel Flow Modelling (MFM) is as a tool for fault diagnosis. The concepts and rules developed may be applied to different processes with a minimum amount of modification and without the need to generate a new rule base. This is achieved by using a modular approach and is aided by G2's programming method, since the rules developed do not refer to particular objects but rather, to classes of objects.

The approach taken can be described by the following steps.

1. An investigation into the theory of MFM with the addition of components to the modelling technique which were found to be lacking.
2. The implementation of the process to be modelled in G2.
3. The implementation of MFM in G2 to develop the required knowledge base.

4. The testing and refinement of the generic rules developed for fault diagnostics.

Initially it was necessary to obtain a reasonable working knowledge of G2. A simple two tank process was simulated. A model of the process was generated and some fault diagnostics were performed on the process. This indicated possible problems that could occur.

A suitable process was chosen for the purpose of the study (see Chapter 2). The knowledge base that was to be developed had to be independent of the actual process and thus not dependent on the number of sensors and actuators in this particular plant.

1.5 Overview.

The First three chapters (1-3) of the project report are introductory in nature. Chapter 4 demonstrates how MFM can be used for fault diagnosis. The last chapter (5) contains concluding remarks.

Chapter 2, Physical Plant Configuration used.

The requirements of the process to be modelled are set out and the functionality of the test plant is described.

Chapter 3, Multilevel Flow modelling.

MFM is introduced, with a description of its flow functions and how they connect to each other. The implementation of MFM in G2 is demonstrated by modelling the process described in chapter 2.

Chapter 4, Fault Diagnostic System.

The development of the fault diagnostic system is proposed. The various steps for fault diagnosis are discussed. They are then implemented in G2 to construct a fault diagnostic system using MFM.

Chapter 5, Conclusion.

The last chapter provides a summary of the investigation and illustrates the significance of the research.

Chapter 2

Process to be Modelled.

A process was required that met the following specifications:

* The process should incorporate two or more types of flow in order to demonstrate the different flow types that can be modelled.
* It should incorporate the majority of the flow functions.
* It should be of reasonable complexity.
* It should be typical of processes found in industry.

Figure 2.1 shows the example created for this investigation. It is a simple system of buffer tanks which mixes two liquids and produces the resultant liquid at a constant rate, re-circulating any excess liquid.

Product A is produced in small quantities and stored in tank-A2, while product B is produced at a constant rate and stored in tank-A1. The liquids are mixed together to produce C, which is stored in tank-B1. Tank-A1 and tank-A2 are situated above tank-B1, with the outflow of each tank being dependent on gravity, described by the following equation:

$$v = \sqrt{(2gh)} \quad \text{Torricelli's Theorem}$$

$$\text{outflow} = vA$$

where outflow (m^3/s)

A is the area of the outlet (m^2)

v is the velocity of the liquid at the outlet (m/s)

g is the gravity constant (9.81 m/s^2)

h is the height of liquid (m)

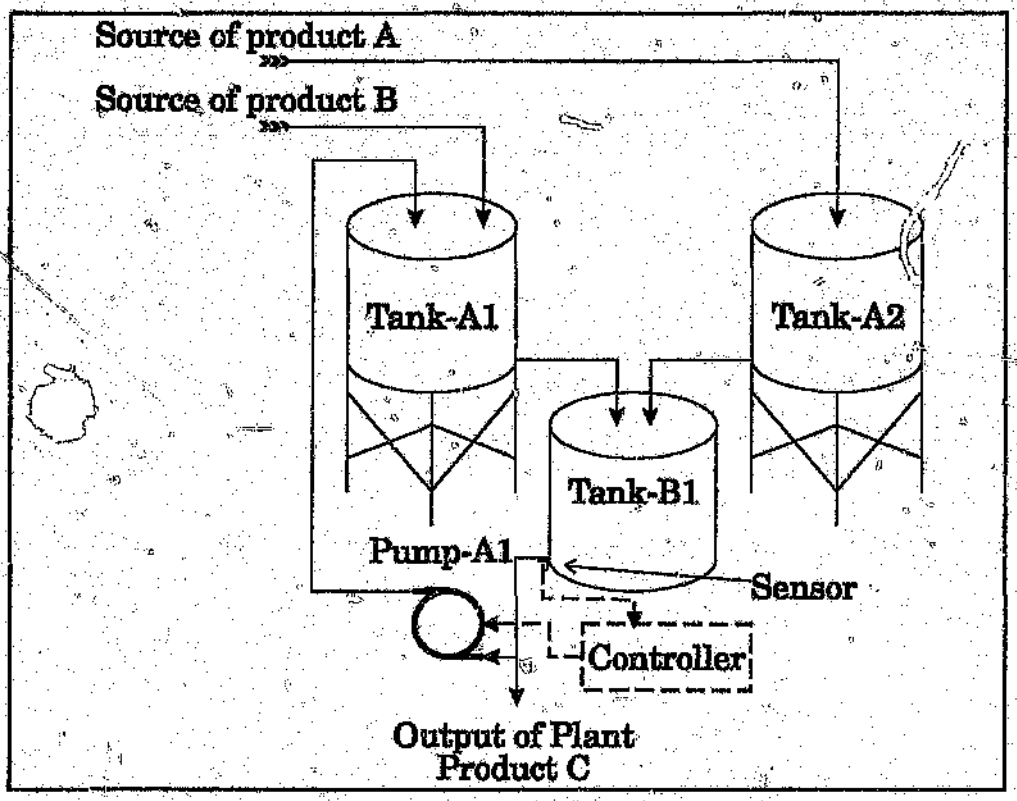


Figure 2.1 Physical arrangement of the Plant

The objective of this plant is to keep the output flow rate of tank-B1 constant. This is achieved by re-circulating the excess of product C into tank-A1 via pump-A1. The controller varies the speed of the pump according to the output of tank-B1. This example was implemented using G2's simulation facility.

The example used could represent a dilution process - by monitoring the level of tank-A2 and controlling the inflow into tank-A1, ignoring pump-A1. It could also represent a fermentation process or a chemical reaction where a catalyst is added and the concentration of C is monitored. The outflow is re-circulated or exits the plant, depending on the state of the reaction.

Chapter 3

Multilevel Flow Modelling.

The existing contributions to multilevel flow modelling have been made by M. Lind at the Technical University of Denmark, J. Rusmussen at the Reso National Laboratory and J. E. Larsson at the Department of Automatic control, Lund Institute of Technology, Sweden. The description of MFM in Larsson's paper [1] "Model-Based Alarm Analysis Using MFM" is used in this project.

MFM provides a physical and functional view of the process being modelled. The physical view describes the components that are present in a process, the way in which they connect together and their dependency upon subsystems. The functional view represents the goals of the process and the functions provided.

MFM models are built up of basic flow functions, which can be connected together to form virtually any type of flow process. The basic flow functions are illustrated in figure 3.1. This provides a graphical representation of the process, easily understood by the operator.

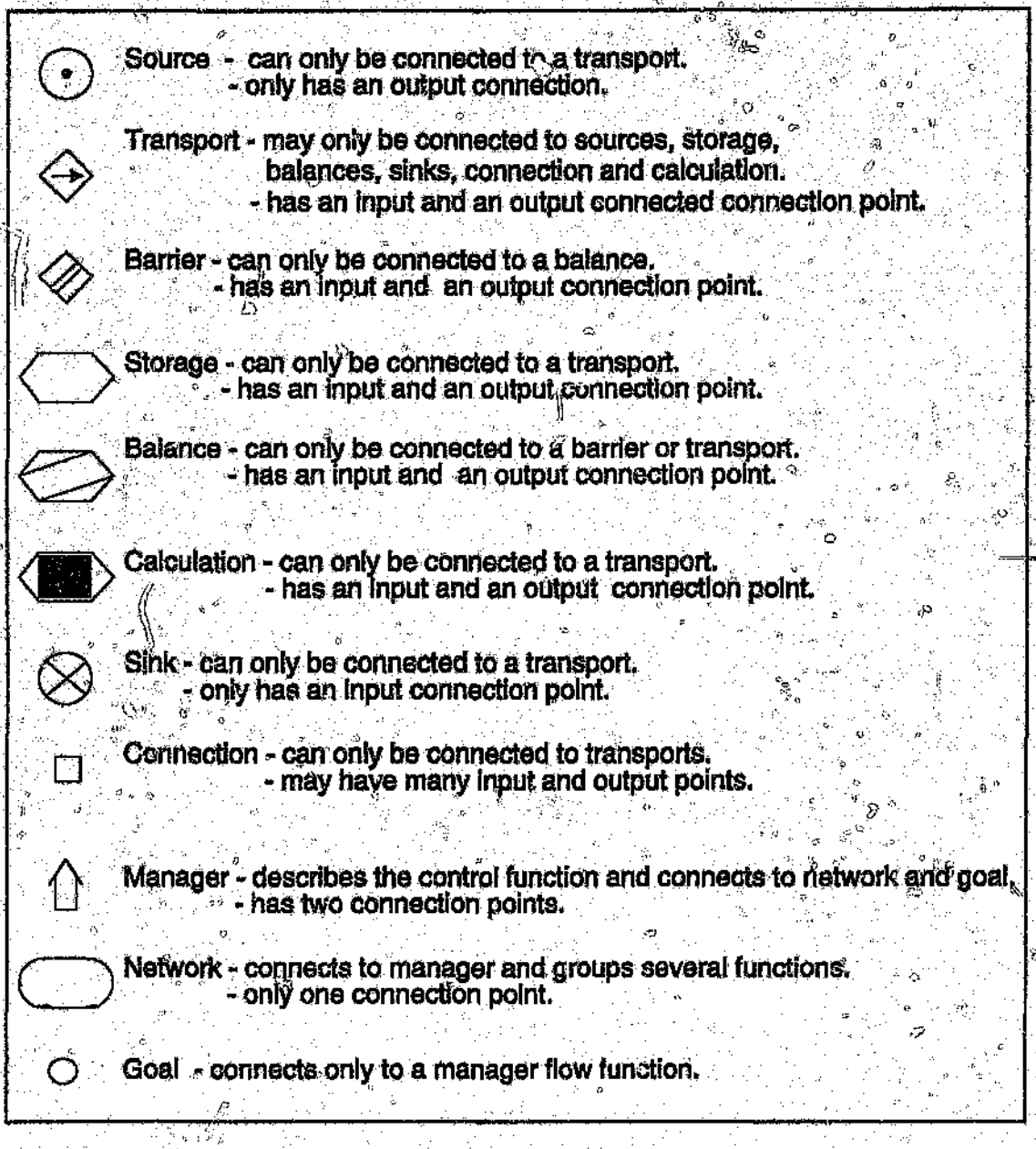


Figure 3.1. Basic flow functions & Connection rules.

Two additional flow functions have been added to Larsson's (1) list, namely:

Connection: This overcomes the problem of three or more objects being connected together in G2 and aids in fault diagnostics. The connection has no sensory data. Its only function is to combine the attributes of the transports connected to it. This is because G2 will not allow identical objects connected to a common object to be referenced by the physical connection between those objects and the common object, since there are two objects of the same class from which to choose.

Calculation: This is due to the fact that in information flow, the information can be changed, i.e. the output is some function of the inputs, not necessarily the sum of the inputs.

Each of the flow functions can be used to represent either mass, energy or information flow with the exception of calculation which can be used only to represent information flow. Note, the flow functions can only be connected according to the connection rules in figure 3.7. The most important rule is that: Flow function of a particular type may only be connected to functions of the same type. [1]

3.1 Implementation of Flow functions in G2.

Each of the flow functions as described above are defined as objects with their own unique icon (figure 3.1), belonging to an MFM class, within the class hierarchy of G2. Note, in G2 an object can be referenced by its name or, as an instance of a

class or, by a physical connection to it.

The flow functions are divided into two groups: mass flow functions and management flow functions. This is illustrated by the hierarchical structure shown in figure 3.2.

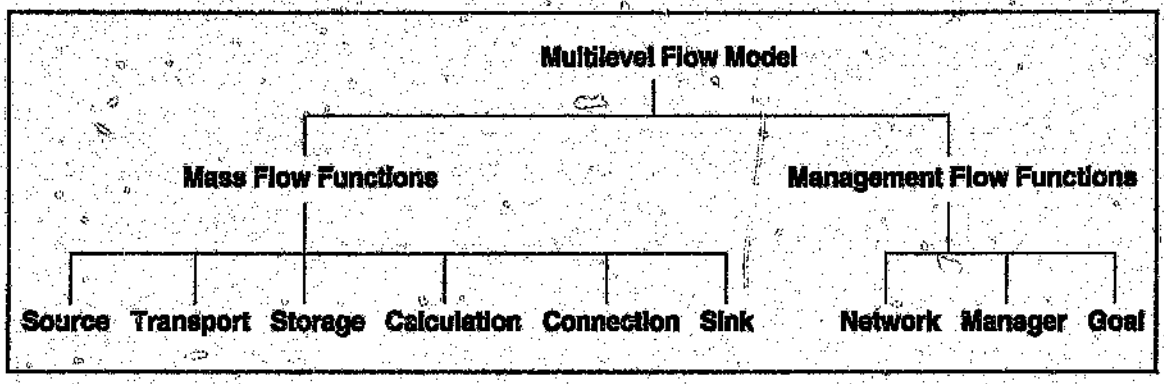


Figure 3.2 Hierarchical Class Structure of MFM flow functions.

Mass flow functions are those functions which relate directly to a physical aspect of the plant and have the following attributes (variables):

Limitations: describe the limits which may not be exceeded by the component being modelled and are set by the user, e.g. low flow, low level, high pressure etc.

Sensory: the type of sensor connected and the reading receiv from the sensor, e.g. flow, level or pressure.

Management: describes the current condition and state of the flow function and the time since the last failure involving the flow function, occurred.

Each management flow function has unique attributes which are

described below:

Network: has no attributes since its purpose is only to indicate which mass flow functions belong to the managem to which it is connected.

Manager: has three attributes, fault1, fault2 and type. Fault1 and fault2 represent the closest upstream and downstream mass flow functions from which the fault can be identified, i.e. the position of the fault. Type, describes the type of fault that has occurred.

Goal: has two attributes object and system. Object describes the goal of the process and system describes whether the goal of the process has been achieved and if not, why?

For further information about the attributes, see Appendix B.

3.2 Modelling the Plant with MFM.

Considering the different types of flow that have to be modelled in the process described in chapter 2, the MFM model shown in figure 3.3 can be obtained. The plant is separated into three flow types, mass, energy and information(control signals).

The mass flow, eg. flow of products A, B and C is described by the flow network called 1-NET and is constructed as follows. 1-SOURCE1 and 1-SOURCE2 represent the source of product B and A respectively, which in turn flow into tanks (1-STOR1 and 1-STOR2) via transports. The outflow of 1-STOR1 and 1-STOR2 are combined to form a single input into tank-B1 (1-STOR3). The outflow of 1-

STOR3 is split into two, to form the output of the process (1-SINK1) and to be re-circulated back into 1-STOR1 (TANK-A1) via connection 1-CON1, with the aid of pump-A1 (1-TRANS8). The transports (1-TRANS) are used to complete the flow path.

Transport, 1-TRANS8 is used to model a pump, which is dependent on the flow of electricity, described by 2-NET. The electricity supply is modelled by a source flowing into a sink. The goal of this network is to provide power for the pump (1-TRANS8).

The amount of power supplied to the pump is controlled by the control network (3-NET). This is modelled by a source (3-SOURCE1, a sensor) with its measurement flowing into a calculation flow function (3-CALC1). 3-CALC1 represents the control function and can be used to check the controller. The signal is then sent to a sink, in this case an actuator. The goal of this network is to check that the actuator is receiving a signal.

The mass flow (1-NET) is the most significant, since it describes the actual mass flow around the process and is the overall goal of the process. Transport 1-TRANS8 is dependent on 2-NET's goal being achieved, making the goal of 2-NET a subgoal of 1-NET since the goal of 1-NET cannot be accomplished if the goal of 2-NET is not achieved. The electric flow is in turn controlled by the controller, described by 3-NET. This forms a hierarchy with the achievement of the overall goal being dependent upon the subgoals.

Note, the numeric value in front of the flow function objects name e.g. 1-NET, indicates its position in the hierarchy

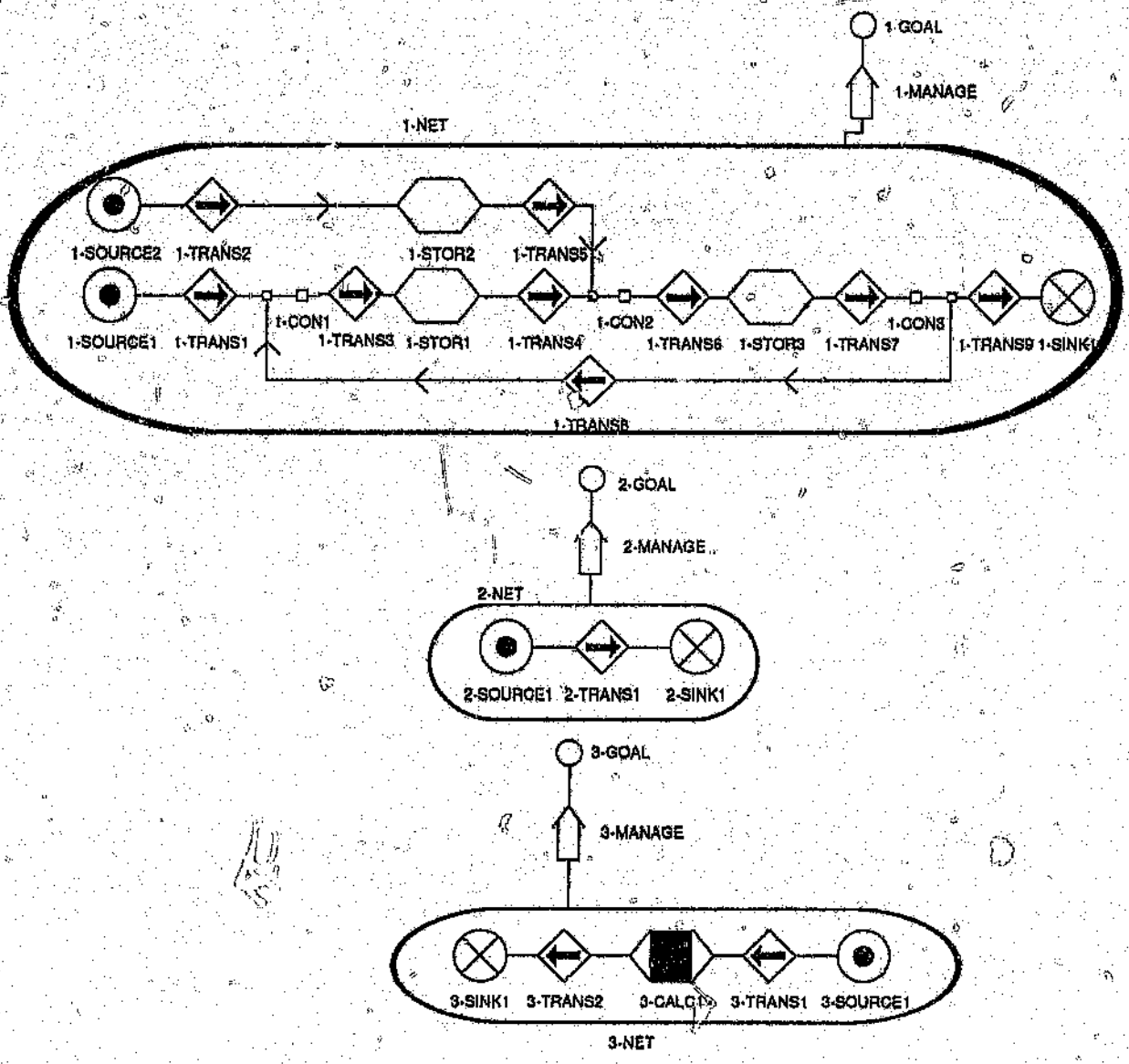


Figure 3.3 MFM representation of the modelled process

Chapter 4

Fault Diagnostic System.

The main aim of this investigation is determine if MFM is a feasible method for fault diagnosis (not to develop a fault diagnostic system). Each of the topics mentioned below are in-depth fields of study in their own right. They are only touched upon in this project report, to show that MFM can be used for fault diagnosis.

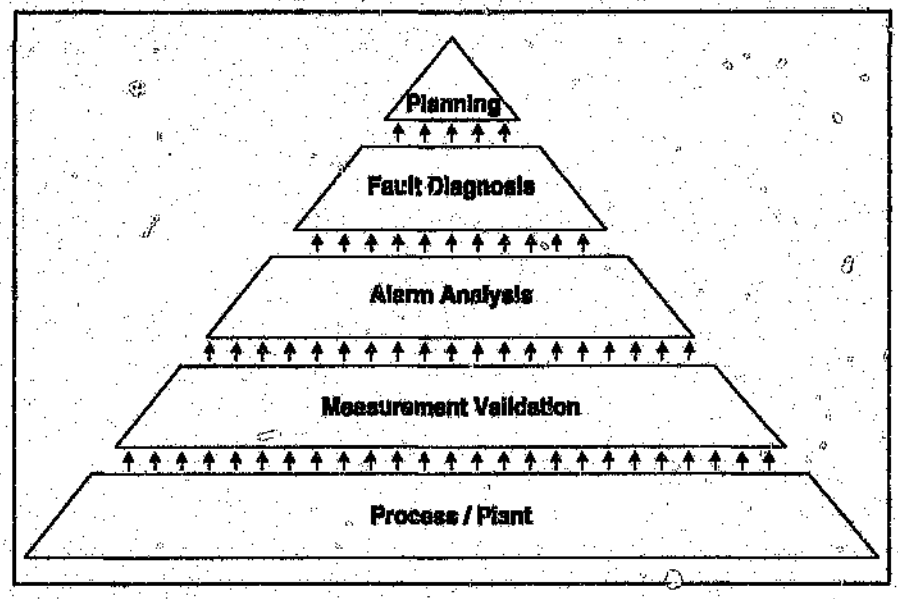


Figure 4.1 Failure Detection Hierarchy.

A fault diagnostic System must check if pre-defined limits are exceeded as well as detect more subtle changes in the relationships between variables. At the same time, in order to have a high degree of reliability, it is essential that there is a way of checking for instrumentation errors. A structured view to fault detection, may been seen in Figure 4.1.

The amount of information that is passed up the failure detection hierarchy from layer to layer decreases with increased meaning to the operator. The accuracy of any layer to perform its function is dependent on the accuracy of the preceding layer. A fault cannot be correctly identified if the correct alarm was not identified as the primary failure.

The fault diagnosis process used in this investigation can be described by the flow chart diagram shown in Figure 4.2.

4.1 Measurement validation.

The first part of any fault detection system is measurement validation, since a correct diagnosis of the fault can't be formulated on incorrect data. There are various methods of measurement validation, ranging from sensor redundancy to analytical redundancy. A measurement validation system must be able to indicate when a sensor has failed as well as when it is about to fail. Since a sensor can fail in unpredictable ways, it is necessary to have a variety of detection methods.

An effective method of measurement validation is hardware redundancy, using three or more sensors to compare the measured values, in order to determine the correct measurement. A type of

analytical redundancy has been considered here since no hardware was actually used is this investigation.

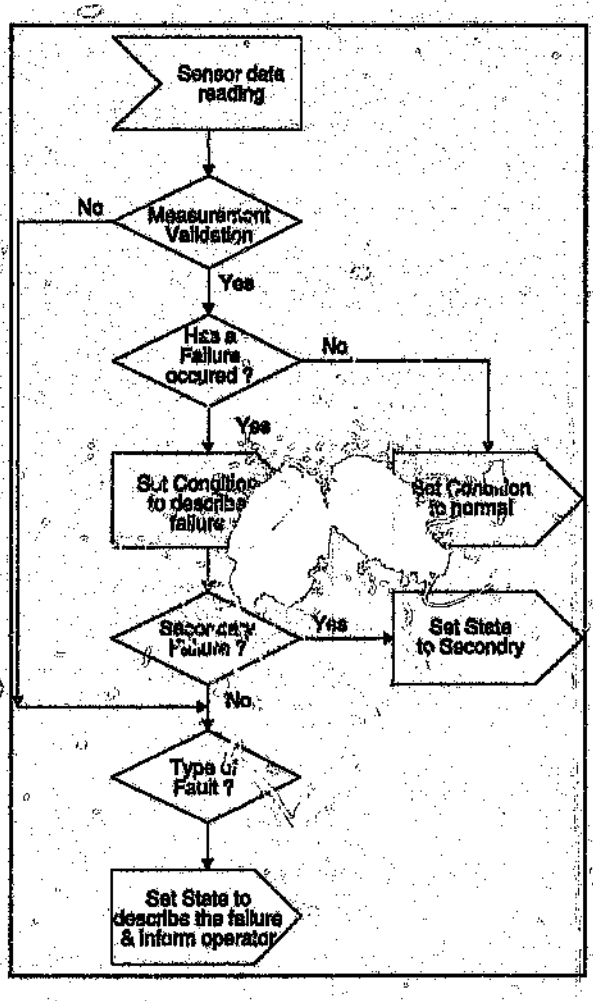


Figure 4.2 Flow Diagram of the Fault Diagnostic Process.

An effective method of analytical redundancy is when all possible flow measurements are taken and by then considering the physical relations between them, a faulty sensor can be identified. This method is not always possible since sensors are expensive, so only the minimal amount of sensors required are used in the controller.

The analytical redundancy method must be able to identify sudden and gradual failures and check for inconsistencies with neighbouring sensors. Two methods of measurement validation have been used to identify these failures:

In method one, with G2's ability to keep a history of the measured value, it is possible to predict, with some degree of accuracy the next measured value. A predictor is usually used in the control environment but has not been used here since it requires the current and previous inputs as well as the previous outputs. This requires complex equations, losing the generality and flexibility of the modelling technique. Extrapolation is used since it is dependent on the previous measurements of the sensor and the time at which the measurement was taken. A sampling rate of one second is used to simplify the equations and reduce the number of variables required. This approach does not affect the generality of MFM since it is independent of the plant being modelled, where a predictor is based on the type of plant being modelled.

Using the Newton-Gregory Forward Difference interpolation formula and using only 5 terms in order to minimize the amount of computer memory used and maximize speed, the following formula derives an approximate figure of the next reading from the sensor.

Predicted measured value = $X_0 - 5X_1 + 10X_2 - 10X_3 + 5X_4$

Where X_0 is the measured value 5 seconds ago
X_1 is the measured value 4 seconds ago etc.
(See Appendix C for derivation.)

If the actual measured value does not equal the predicted measured value ± the error in extrapolation, then the sensor is

presumed to be faulty. The state of the flow object is set to faulty_type_sensors, which describes the type of sensor that has failed. The sensor is disabled if two consecutive readings are found to be faulty. The sensor can only be reactivated by the operator.

The above method will identify sudden sensor failures but not gradual sensor drift or failure.
The second method is designed to identify sensor drift. This is done by comparing the condition of each flow function with the condition of its neighbouring flow functions. If the condition is not the same as its neighbours then the sensor has failed. Again, the failure is logged and the state of the object is set to faulty_type_sensor.

The analytical redundancy used, provides solid grounds for presuming that the sensor data is valid and allows for accurate alarm analysis. (Rules in Appendix E)

4.2 Alarm Analysis.

Alarm Analysis is a primary step towards the development of an intelligent controller. At this stage the distinction between primary and secondary faults is made. This distinction is required since alarms almost never occur singly and have a domino effect, triggering off other alarms. In the Three Mile Island accident, within a minute of the first event, there was a cascade of 100 alarms [4]. This evidence points out the importance of showing the operator the primary faults only, so as not to clutter the operator with unnecessary information in a crisis.

By analysing the relationships between flow functions, it is possible to distinguish primary from secondary alarms. This is done by considering the symptoms of each flow function e.g. low flow, high pressure, which are stored in the condition attribute of each flow function. The symptoms are generated when predefined limits are exceeded or, a sensor is found to be faulty.

It is important that the alarm limit be set accurately in order to prevent false alarms or, alarm conditions to exist but not be recognised; for this could be detrimental to the process.

Once a limit of the flow, level or pressure has been exceeded for any flow function and the measurements are valid, the condition of the flow function is set to a meaningful name which describes the failure (symbolic variables of condition, see Appendix B). By comparing the condition of the flow function with the condition of surrounding flow functions, it is possible to make a distinction between which alarms are primary and which are secondary.

A secondary fault can be identified from primary failures by using the rules in Appendix F. These can be condensed into the following rule:

IF the symptoms of any flow function are similar to the symptoms of the relevant neighbouring flow function
THEN the failure occurred by the flow function is a secondary failure.

e.g.

IF a flow function has high pressure and low flow
AND it's downstream neighbour has high pressure and low flow
THEN the flow functions alarm state is secondary
*** since the fault is down stream.

Once a secondary failure has been identified the state of the flow function is set to secondary and the clear_state_counter is set to zero. Ten seconds after the symptoms have disappeared i.e. condition is normal and no new symptoms have appeared, then it is assumed that the fault has been cleared. The state of the flow function will automatically be set to normal.

4.3 Fault Diagnostics.

At this point the data from the sensors has been filtered and the knowledge base is focused around the primary faults. When constructing a plant it is physically and economically unlikely that all flow, pressure and other relevant readings may be measured. This makes it impossible to identify the exact point in the process where the fault occurred.

A method for identifying the region where the fault occurred and the type of fault, has been developed. Two fault variables are used which identify the region in the process where the fault occurred. These are the nearest upstream and downstream flow functions. From these, the fault can be identified. A third variable, fault_type is used to describe the type of fault e.g. leak or blockage. All three variables are attributes of the manager function. An error message is also displayed on the message board to notify the operator that an error has occurred.

The accuracy of the fault diagnosis is dependent on the number of sensors used. Since sensors can be expensive, often only the minimal number of sensors needed for control and safety are used. Consequently, not all flow functions will have sensory data, making it impossible for these flow functions to have any

Once a secondary failure has been identified the state of the flow function is set to secondary and the clear_state_counter is set to zero. Ten seconds after the symptoms have disappeared i.e. condition is normal and no new symptoms have appeared, then it is assumed that the fault has been cleared. The state of the flow function will automatically be set to normal.

4.3 Fault Diagnostics.

At this point the data from the sensors has been filtered and the knowledge base is focused around the primary faults. When constructing a plant it is physically and economically unlikely that all flow, pressure and other relevant readings may be measured. This makes it impossible to identify the exact point in the process where the fault occurred.

A method for identifying the region where the fault occurred and the type of fault, has been developed. Two fault variables are used which identify the region in the process where the fault occurred. These are the nearest upstream and downstream flow functions. From these, the fault can be identified. A third variable, fault_type is used to describe the type of fault e.g. leak or blockage. All three variables are attributes of the manager function. An error message is also displayed on the message board to notify the operator that an error has occurred.

The accuracy of the fault diagnosis is dependent on the number of sensors used. Since sensors can be expensive, often only the minimal number of sensors needed for control and safety are used. Consequently, not all flow functions will have sensory data, making it impossible for these flow functions to have any

symptoms. In this case the flow function merely reflects the condition of the opposite neighbouring flow function from which condition was requested, having a recursive effect if required. The primary fault is diagnosed only by the condition of the flow function and it's immediate neighbours, even if the neighbouring flow functions do not have sensory data, due to the reflection of data. (See Appendix D for workings of reflection).

A primary failure is diagnosed with the same method used in alarm analysis. The two types of failure which are considered here are leaks and blockages (See Appendix G for rules). These failures can be identified by the following rules and by reading across the table 4.1 below:

IF the condition of any flow function is 1
AND the condition of the immediate downstream neighbour is 2
THEN the failure is between the two flow functions and the flow function is the upstream failure point and the type of failure is 3.

IF the condition of any flow function is 2
AND the condition of the immediate upstream neighbour is 1
THEN the failure is between the two flow functions and the flow function is the upstream failure point and the type of failure is 3.

1	2	3
Low_Flow_High_Pressure or High_Level_High_Pressure or High_Pressure or High_Level	Low_Flow_Low_Pressure or Low_Level_Low_Pressure or Low_Flow or Low_Level or Low_Pressure or Normal	Blocked
High_Flow_Low_Pressure or High_Flow_High_Pressure or Low_Level_Low_Pressure or low_Level or High_Flow	Low_Flow_Low_Pressure or Low_Level_Low_Pressure or Low_Flow or Low_Level or Normal	Leak

Table 4.1 Conditions for Fault Recognition

The upstream and downstream failures cannot be identified in the same rule since the data may be reflected. For example, if a blockage occurred in the storage tank (1-STOR3) in Figure 3 and the storage tank had no sensory data, the following would be found to be at fault:

Upstream : 1-TRANS6
Downstream : 1-TRANS7
Type : blockage

by the manager function.

Once the fault has been identified, the fault region and fault type variables of the manager function of the network are set as shown in the example above. This allows the MFM modelling technique to indicate the failure to the operator, quickly and accurately.

4.4 MFM Process Management.

Dividing a problem up into smaller problems and in turn solving the smaller problems, is a simple and effective way of solving a problem as a whole. MFM uses this approach by dividing the flow model into different types of flow, which are separately managed.

MFM process management is built up from three flow functions, network,manager and goal functions. The network groups together flow functions of the same type with a common objective. The manager describes the control function and reports any failure in the process. The goal displays whether the control function of the system is achieved and if not, why not?

The manager flow function receives data from its flow network. The data stored in the manager's attributes is supplied from the diagnostic rules described earlier in this chapter. Manager flow functions have a knowledge base called manager rules (Appendix H). Manager rules take into account the values of the manager attributes and compile an intelligent message describing the predicament of the process. Conclusions from this knowledge base are reported to the system goal.

The system goal provides a quick reference to indicate to the operator the state of the process. A process may have more than one goal, depending on the number of different types of flow being considered and whether the flow paths cross.

The management process forms a hierarchy, whereby an object in a network may be dependent upon the goal of another flow network. In the example used, the flow of the pump (1-TRANS8) is dependent upon the goal of 2-NET which is the electricity supply. If the goal of 2-NET is not achieved, the goal of 1-NET cannot be

achieved. This forms an important interface with the operator since it is only necessary for him to observe the highest process goals, to determine if the process is achieving its objectives.

MFM process management provides a concise source of data for the final part of an intelligent supervisory system, namely planning, which has not been attempted in this investigation, due to lack of time.

4.5 Applying MFM to Other Plants.

The concepts and rules that have been developed in this chapter can easily be applied to other plants. This is achieved by copying the generalised modelling technique with the generic fault diagnostic system to a new work space and constructing the plant with the flow functions of MFM, by considering the different flow types involved. All that is then required is that the flow functions be connected together, named, a path set to their sensory data and the sensor limits set. This will provide a description of the structural and functional relations between objects, expressed in a graphical form and provide a fault diagnosis facility.

In this chapter it has been demonstrated that MFM can be used as a generalised modelling technique to perform the task of fault diagnosis.

Chapter 5

Conclusion.

The project report has investigated a fault diagnosis system and introduced MFM, showing how MFM can be used for fault diagnostics. It has been found, during this investigation that MFM can be used as a generalised modelling technique and that MFM can be used successfully for fault diagnosis.

5.1 General.

The main goal of this investigation was to determine whether MFM could be used to perform tasks such as measurement validation, alarm analysis and fault diagnosis. Secondly, to generate a set of generic rules to perform these tasks.

A fault diagnostic rule base was constructed around MFM, to assess the effectiveness of MFM to perform the task of fault diagnosis. With the limitations of the simple test process simulated in G2, it has been demonstrated that MFM can be used for fault diagnosis with current fault diagnostic theory. A set of generic rules was developed to perform this task.

While testing the rule base, the advantages and disadvantages of generic rules were highlighted. Generic rules have the disadvantage of being slower than specific rules to determine the failure, since a generic rule will not only be fired for the faulty flow function but also all other flow functions in the same class. Alternatively, there are certain advantages to using generic rules, such as reduced programming time, and a reduction in the number of rules and the amount of memory needed.

5.2 Assessment.

5.2.1 G2 and Implementation of MFM.

G2 is an excellent tool for developing real-time expert systems. Its ability to express and make use of the relationships between objects (actual connection between objects and the hierarchical class structure of the objects) allows for easy programming and a reduced number of rules, since an object can be referenced by a connection, by the class (generically) or directly by the object's name. The ability, dynamically to create and destroy objects and rules allows the process modelled with G2 to adapt itself to change in the process.

The following problems were encountered while using G2:

1) G2 would crash when an attribute was referenced which did not exist, without giving any explanation of the error.
2) G2 has a limited print out facility, which only allows the user to print on letter size paper, using post script

language.

3) G2 cannot reference an object by a connection from a common object, if two objects from the same class are connected to the common object.

These problems encountered using version 3.0 Beta may be addressed in a later version of G2.

5.2.2 MFM.

MFM is made up from a few basic flow functions, from which more complicated flow systems can be constructed. This limits the knowledge base to a few functions around which a concise knowledge base for fault diagnosis is developed, using G2.

Virtually any flow process can be broken down into the basic flow functions available in MFM. With the aid of the generic rule base and the inherent nature of the programming style used in G2, the flow process can be modelled easily. The power that this approach has over conventional modelling, is the ease with which process changes and new designs can be implemented while retaining the ability to perform fault diagnostics.

5.2.3 Limitations and Significance.

As with all theories, there are limitations. The limitations of this research project can be summarised as follows:

1) Parameters associated with the given state of a flow function (component) cannot be defined precisely.
2) The accuracy of the knowledge base is dependent on the number of sensors used.
3) Flow functions can fail in unexpected ways, making it impossible to satisfy the conditions required to identify all possible failures.
4) Due to the use of generic rules the fault diagnostic process will not perform as well as a dedicated technique in a time critical process.
5) The limitations encountered while using G2, will most likely be overcome in a later release of G2.
6) Only flow and pressure related failures are considered.
7) The components in the example used have constant limits. This is not always the case since the limits may be dependant upon time or the state of other components in the plant.

The limitations experienced could be overcome with further research. This research project has therefore indicated that with the technology currently available and present fault diagnosis theories, it would be feasible to implement a fault diagnosis system in industry.

REFERENCES:

1. Jan Eric Larsson(1991):"Model-Based Alarm Analysis Using MFM", Department of Automatic Control, Lund Institute of Technology.

2. Kourosh Danai, Hsinyung Chin (1991): "Fault Diagnosis With Process Uncertainty" Journal of Dynamic Systems, Measurement and Control, September 1991,Vol 113, pp.339 -343.

3. Hilding Elmquist and Sven Erik Mattsson: "Simulator for Dynamical Systems Using Graphics and Equations for Modelling" IEEE Control Magazine, January 1989.

4. W.F. Kaemmerer and S. Zvolner: "Control-Systems Fault Diagnosis Using Models" Scientific Honeyweller, Summer 1988 pp. 94 - 100.

5. G.Bruno, A.Elia and P.Laface: "A Rule-Based System to Schedule Production" Computer, July 1986.

6. Isermann, Rolf (1984) "Process Fault Detection Based on Modeling and Estimation Methods - A Survey" Automatica, Vol 20, No 4, pp. 387 - 404.

7. Piercy N.P. (1992) "Sensor Failure Estimators for Detection Filters", IEEE Transactions on Automatic Control, vol 37 No. 10 , October 1992, pp. 1553 - 1558.

8. F.G. Shinskey (1990) "How good are our controllers in absolute performance and robustness?" Measurement and Control, Volume 23, May 1990, pp. 114 - 121.

9. Frank P.M. (1990) "Fault Diagnosis in Dynamic Systems Using Analytical and Knowledge-based Redundancy - A Survey and Some New Results", Automatica, Vol. 26, No. 3, pp. 459 - 474.

10. Watanabe k and Tzafestas S.G. (1990) "A Hierarchical Multiple Model Adaptive Control of Discrete-time Stochastic Systems for Sensor and Actuator Uncertainties" Automatica, Vol. 26, No. 5, pp. 875 - 886.

11. Romanica M.F. (Feb 1986) "Dynamic modelling and fault detection, using a rule - based system, in a milling circuit" Department of Electrical Engineering, University of the Witwatersrand, MSc project.

12. Prock J. (1991) "A New Technique for Fault Detection Using Petri Nets" Automatica, Vol. 27, No. 2, pp..239 - 245.

13. Jones P. (1992) "Computer control of industrial processes", Measurement + Control, Volume 25, October 1992, pp.243 - 247.

14. Medlock R.S. and Furness R.A. (1990) "Mass flow measurement -a state of the art review", Measurement and Control, Volume 23, May 1990, pp. 100 - 112.

15. Isermann R and Freyermuth B (Dec 1991) "Process Fault Diagnosis Based on Process Model Knowledge - Part I: Principles for Fault Diagnosis With Parameter Estimation" Transactions of the ASME, Journal of Dynamic Systems, Measurement, and Control, Vol. 113, pp. 620 - 626.

16. Isermann R and Freyermuth B (Dec 1991) "Process Fault Diagnosis Based on Process Model Knowledge - Part II: Case Study Experiments" Transactions of the ASME, Journal of

Dynamic Systems, Measurement, and Control, Vol. 113, pp. 627 -633.

17. Gertler J.J. (Dec 1988) "Survey of Model-Based Failure Detection and Isolation in Complex Plants." IEEE Control Systems Magazine, Dec. 1988, pp. 3 - 11.

18. Smith R "Structuring a Knowledge Base for Real - Time Diagnosis " Gensym Corporation.

19. "G2 Reference Manual" (August 1990) Gensym Corporation 125 Cambridge Park Drive, Cambridge, MA 02140.

20. Franklin G.F., Powell J.D. and Emami-Naeine A. "Feedback Control of Dynamic Systems", Addison-Wesley Publishing Company June 1988.

21. Giancoli D.C. "Physics for Scientists and Engineers" Prentice-Hall International Editions, Second Edition 1988.

Appendix A

Test Plant

This appendix describes how the test process was implemented in G2.

Using the simulation facility in G2 the process was implemented. The plant consisted of three tanks, a pump and a controller, created as objects in G2. Each object has the same specifications:

Validity Interval : 1 second
Data server : G2 Simulator

The data server for all plant variables is the G2 simulator. Each variable will seek data from its respective data server. The variable may not cause forward chaining when it receives a new value. The variable is simulated using a simulation formula in the variable's simulation subtable.

Tanks:

All three tanks have the same attributes and are governed by similar equations.

Simulation Details:

Inflow:

Tank-A1: The sum over each pump pump-A1 connected to tank-A1 of (the flow of pump-A1) + 6.5

Tank-A2: (If random(100) > 98 then random(100) else 0

Tank-B1: The sum over each tank tank-A1 connected to tank-B1 of (the outflow of tank-A1)

Outflow:

(the level of tank-XX * 2 * 9.81) ^0.5

Volume:

state variable: d/dt = (the inflow of tank-XX - the outflow of tank-XX) with initial value 50

Level:

max(0,the volume of tank-XX/20)

Pressure:

(the outflow of tank-XX)^2 * 0.5 * 1000

Pump simulation details:

Flow:

state variable: next value = max (0, (the outflow of tank-b1)) with initial valve 0

Pressure:

0.5 * 1000 * (the flow of pump-A1 ^ 2)

Controller:

Control_signal:

(the outflow of tank-b1 - 6.75)

Note, XX denotes either A1, A2 or B1.

By arranging the objects on a work space and connecting them together, the process was implemented in G2.

Appendix B

Attributes of Flow Functions

This appendix describes the attributes of the different flow functions.

B.1 Mass Flow Functions

The following attributes may exist for each mass flow function.

Low flow, High flow, Low pressure & High pressure :

To set limits, so that abnormal flow or pressures can be detected for different types of objects, since not all storage objects are identical. These values are set by the user.

Flow:

Stores the actual measured value from the flow sensor of this particular flow function. A history is kept with five data points. The value is obtained from the sensor (G2 simulator).

Pressure:

Stores the measured pressure reading. A history is kept with five data points. The value is obtained from the sensor (G2 simulator).

Condition:

A symbolic variable that stores the condition of the object and may have values; normal, faulty_flow_sensor, faulty_pressure_sensor, low_flow, low_pressure, high_flow, high_pressure, low_flow_low_pressure, low_flow_high_pressure, high_flow_low_pressure, high_flow_high_pressure and no_sensory_data. Value is obtained for con and con_level functions. (Appendix J.1)

In_condition & Out_condition:

These attributes contain the condition of the closest upstream or downstream neighbours from the connection.

Clear_state_counter:

A counter to determine the length of time that has past since the last failure occurred.

State:

A symbolic variable showing the overall state of the object in relation to other objects and may have the value; Ok, secondary, fault_flow_sensor, faulty_pressure_sensor, Fault_blockage, Fault_leakage and no_sensory_data. Value is obtained from the rules in Appendix E through G.

Note :

(1) flow, pressure, condition, clear_state_counter and state have a validity of one second.
(2) Storage objects don't have low flow, high flow or flow, but low level, high level and level. Similar for the condition of the object.
(3) When considering information flow, ie. electrical flow where only current and voltage can be measured, flow refers to current and pressure to voltage.

B.2 Management Flow functions

Network: has no attributes since it's purpose is only to indicate which mass flow functions belong to the manager connected to it.

Manager: has three attributes, fault1, fault2 and type. Fault1 and fault2 represent the closest upstream and downstream mass flow functions, from which the fault can be identified, i.e. the position of the fault. Type describes the type of fault that has occurred. Value is obtained from the rules in Appendix E through G).

Goal: has two attributes object and system. Object describes the goal of the process and system describes whether the goal of the process has been achieved and if not, why? Value is obtained from the rules in Appendix H.

Appendix C

Measurement Validation Extrapolation

This appendix contains the calculations in deriving the equation for predicting the measured value.

Using the Newton-Gregory Forward Difference interpolation formula:

$$P_N(x) = y_0 + \frac{\Delta y_0}{h}(x-x_0) + \frac{\Delta^2 y_0}{2h^2}(x-x_0)(x-x_1) + \ldots + \frac{\Delta^N y_0 \Pi_{i=0}^{N-1}}{N!h^N}(x-x_1)$$

h = 1 since the intervals between samples is one second.

i	xi		Δy_i	$\Delta^2 y_i$	$\Delta^3 y_i$	$\Delta^4 y_i$
0	-5	y_0				
1	-4	y_1	y_1-y_0			
2	-3	y_2	y_2-y_1	$y_2-2y_1+y_0$		
3	-2	y_3	y_3-y_2	$y_3-2y_2+y_1$	$y_3-3y_2+3y_1-y_0$	
4	-1	y_4	y_4-y_3	$y_4-2y_3+y_2$	$y_4-3y_3+3y_2-y_1$	$y_4-4y_3+6y_2-4y_1+y_1$

$$P_4(0) = y_0 + 5*(y_1-y_0) + \frac{5*4*}{2}(y_2-2y_1+y_0) + \frac{5*4*3*}{2*3}(y_3-3y_2+3y_1-y_0)$$

$$+ \frac{5*4*3*2*}{}(y_4-4y_3+6y_2-4y_1+y_1)$$

$$2*3*4$$

$$= y_0 + 5*(y_1-y_0) + 10*(y_2-2y_1+y_0) + 10*(y_3-3y_2+3y_1-y_0)$$
$$+ 5*(y_4-4y_3+6y_2-4y_1+y_1)$$

$$P_4(0) = y_0 - 5y_1 + 10y_2 - 10y_3 + 5y_4$$

Appendix D

Data Reflection

This appendix describes how data is reflected when there is no sensory data available. Reflection and reflection_secondary work on similar principals so only reflection is described.

A flow function needs to be informed of whether it has sensory data or not. This is done by setting the maximum and minimum (high and low) limits equal to each other i.e. both equal to zero.

eg.

low_flow = 0 & high_flow = 0

Each flow function determines its condition by calling the con function. No_sensory_data will be returned if the flow function has no sensory data.

First few lines of con function:

con(who) =

```
IF the low_flow of who = the high_flow of who
  AND the low_pressure of who = the high_pressure of who
THEN the symbol no_sensory_data
  AND the state of who is no_sensory_data
```

ELSE .....etc.

Note: "who" is the name of the flow function being considered. The state of the flow function is automatically set since no conclusion can be made about the state, if there is no sensory data.

If a flow function requires the condition of a neighbouring flow function which has no sensory data then the reflection function is called.

eg. from rule base:

IF ...
OR (IF the condition of the neighbouring flow function connected at the output of the flow function is no_sensory_data
THEN reflection(the flow function connected at the output of the flow function,output,blockage)) is faulty
THEN ... etc.

The reflection function will return the symbol faulty if the down stream neighbours satisfy the condition being checked, making the IF statement true. The reflection function works on the principals of Table 4.1. The process may be recursive (see Appendix J.3 for listing of the reflection function).

Appendix E

Measurement Validation Rules

This appendix contains the rules used for measurement validation.

MEASUREMENT_VALIDATION

IF the condition of any mfm_flow_functions is
faulty_flow_sensor
AND the state of the mfm_flow_functions is not
faulty_flow_sensor
THEN conclude that the state of the mfm_flow_functions is
faulty_flow_sensor
AND inform the operator for the next 5 seconds that "[the
name of the mfm_flow_functions] has a faulty flow
sensor. Check the sensor and once the sensor is
operational again, then reset the state of [the
name of the mfm_flow_functions]."

IF the condition of any mfm_flow_functions is
faulty_pressure_sensor
AND the state of the mfm_flow_functions is not
faulty_pressure_sensor
THEN conclude that the state of the mfm_flow_functions is
faulty_pressure_sensor
AND inform the operator for the next 5 seconds that " [the
name of the mfm_flow-functions] has a faulty
pressure sensor. Check the sensor and once the

sensor is operational again, then reset the state of [the name of the mfm_flow_functions]."

IF the condition of any mfmstorage is faulty_flow_sensor
AND the state of the mfmstorage is not faulty_flow_sensor
THEN conclude that the state of the mfmstorage is faulty_flow_sensor
AND inform the operator for the next 5 seconds that " [the name of the mfmstorage] has a faulty flow sensor. Check the sensor and once the sensor is operational again, then reset the state of [the name of the mfmstorage]."

IF the condition of any mfmstorage is faulty_pressure_sensor
AND the state of the mfmstorage is not faulty_pressure_sensor
THEN conclude that the state of the mfmstorage is faulty_pressure_sensor
AND inform the operator for the next 5 seconds that " [the name of the mfmstorage] has a faulty pressure sensor. Check the sensor and once the sensor is operational again, then reset the state of [the name of the mfmstorage]."

IF the condition of any mfmsource is not normal
AND the condition of the mfmtransport connected to the mfmsource is normal
AND ((IF there exists a mfmstorage connected to the mfmtransport the condition of the mfmtransport is normal)
OR (IF there exists a mfmsink connected to the mfmtransport
THEN the condition of the mfmsink connected to the mfmtransport is normal))
THEN inform the operator for the next 2 seconds that " source [the name of the mfmsource] has a faulty sensor"
AND invoke transport rules

```
    AND conclude that the state of the  mfmsource is fault
    AND conclude that fault1 = "[the name of the mfmsource]"
    AND conclude that fault2 = fault1

IF the condition of any mfmtransport is not normal
AND (IF there exists a mfmsource connected to the
        mfmtransport
     THEN the condition of the mfmsource connected to the
        mfmtransport is normal)
AND ((IF there exists a mfmstorage connected to the
     mfmtransport
     THEN the condition of the mfmstorage connected to the
        mfmtransport is normal)
   OR (IF there exists a mfmsink connected to the
        mfmtransport
     THEN the condition of the mfmsink connected to the
        mfmtransport is normal))
   THEN inform the operator for the next 2 seconds that
      "Transport [the name of the mfmtransport] has low flow
      and low pressure -     k sensors"
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is fault
   AND conclude the fault1 = "[the name of the mfmtransport]"
   AND conclude that fault2 = fault1


IF the condition of any mfmstorage is not normal
AND the condition of the mfmtransport connected at the input
   of the mfmstorage is normal
AND the condition of the mfmtransport connected at the output
   of the mfmstorage is normal
   THEN  inform the operator for the next 2 seconds "storage
      [the name of the mfmstorage] has low level - check
      sensor
   AND invoke transport rule
   AND conclude that the state of the mfmstorage is fault
```

```
    AND conclude that fault2 = "[the name of the mfmstorage]"
    AND conclude that fault1 = fault2


IF the condition of any mfmsink is not normal
AND the condition of the mfmtransport connected to the
    mfmsink is normal
AND ((IF there exists a mfmsource connected to the
            mfmtransport
        THEN the condition of the mfmsource connected to the
            mfmtransport is normal)
    OR (IF there exists a mfmstorage connected to the
        mfmtransport
        THEN the condition of the mfmstorage connected to the
            mfmtransport is normal))
    THEN inform the operator for the next 2 seconds that" sink
        [the name of the mfmsink] has a errOR in the sensor
        reading"
    AND invoke transport rules
    AND conclude that the state of the mfmsink is fault
    AND conclude that fault2 = "[the name of the mfmsink]"
    AND conclude that fault1 = fault2
```

Appendix F

Alarm Analysis Rules

This appendix contains the rules used for alarm Analysis.

F.1 Source Rules

Rules used to determine secondary alarms with mfm source function.

SECONDARY_SOURCE

```
IF the condition of any mfmsource is low_flow
AND (the condition of the mfmtransport connected to the
        mfmsource is low_flow_high_pressure
   OR the condition of the mfmtransport connected to the
      mfmsource is high_pressure
   OR  (IF the condition of the mfmtransport connected to the
      mfmsource is no_sensory_data
      THEN reflection_secondary(the condition of the
      mfmsource,the flow_function connected at the output of
      the mfmsource ,output)) is faulty))
   THEN inform the operator for the next 2 seconds that "
      source [the name of the mfmsource] has low flow "
   AND invoke transport rules
   AND concluded that the state of the mfmsource is secondary
```

```
IF the condition of any mfmsource is high_flow_high_pressure
AND (the condition of the mfmtransport connected to the
        mfmsource is high_flow_high_pressure
   OR the condition of the mfmtransport connected to the
      mfmsource is high_pressure
   OR the condition of the mfmtransport connected to the
      mfmsource is high_flow
   OR the condition of the mfmtransport connected to the
      mfmsource is high_flow_low_pressure
   OR  (IF the condition of the mfmtransport connected to the
      mfmsource is no_sensory_data
      THEN reflection_secondary(the condition of the
      mfmsource,the flow_function connected at the output of
      the mfmsource ,output)) is faulty)
   THEN inform the operator for the next 2 seconds that "
      Source [the name of the mfmsource] has high flow and
      high pressure"
   AND invoke transport rules
   AND conclude that the state of the mfmsource is secondary

IF the condition of any mfmsource is high_flow
AND (the condition of the mfmtransport connected to the
        mfmsource is high_flow_high_pressure
   OR the condition of the mfmtransport connected to the
      mfmsource is high_flow_low_pressure
   OR the condition of the mfmtransport connected to the
      mfmsource is high_flow
   OR the condition of the mfmtransport connected to the
      mfmsource is high_pressure
   OR  (IF the condition of the mfmtransport connected to the
      mfmsource is no_sensory_data
      THEN reflection_secondary(the condition of the
      mfmsource,the flow_function connected at the output of
      the mfmsource ,output)) is faulty)
   THEN inform the operator for the next 2 seconds that
      "source [the name of the mfmsource] has high flow"
   AND invoke transport rules
```

AND conclude that the state of the mfmsource is secondary

IF the condition of any mfmsource is low_flow_high_pressure
AND (the condition of the mfmsource connected to the
mfmsource is low_flow_high_pressure
OR the condition of the mfmtransport connected to the
mfmsource is high_pressure
OR (IF the condition of the mfmtransport connected to the
mfmsource is no_sensory_data
THEN reflection_secondary(the condition of the
mfmsource,the flow_function connected at the output of
the mfmsource ,output)) is faulty)
THEN inform the operator for the next 2 seconds that
"source [the name of the mfmsource] has low flow and
high pressure"
AND invoke transport rules
AND conclude that the state of the mfmsource is secondary

IF the condition of the mfmsource is high_pressure
AND (the condition of the mfmtransport connected to the
mfmsource is low_flow_high_pressure
OR the condition of the mfmtransport connected to the
mfmsource is high_flow
OR the condition of the mfmtransport connected to the
mfmsource is high_flow_high_pressure
OR the condition of the mfmtransport connected to the
mfmsource is high_pressure
OR (IF the condition of the mfmtransport connected to the
mfmsource is no_sensory_data
THEN reflection_secondary(the condition of the
mfmsource,the flow_function connected at the output of
the mfmsource ,output)) is faulty)
THEN inform the operator for the next 2 seconds that
"source [the name of the mfmsource] has high pressure"
AND invoke transport rules

```
    AND conclude that the state of the mfmsource is secondary


IF the condition of the mfmsource is high_flow_low_pressure
AND (the condition of the mfmtransport connected to the
      mfmsource is high_flow_low_pressure
   OR the condition of the mfmtransport connected to the
      mfmsource is high_flow
   OR the condition of the mfmtransport connected to the
      mfmsource is low_pressure
   OR  (IF the condition of the mfmtransport connected to the
      mfmsource is no_sensory_data
      THEN reflection_secondary(the condition of the
      mfmsource,the flow_function connected at the output of
      the mfmsource ,output)) is faulty)
   THEN inform the operator for the next 2 seconds that
      "source [the name of the mfmsource] has high flow and
      low pressure"
   AND invoke transport rules
   AND conclude that the state of the mfmsource is secondary
```

F.2 Transport Rules

Rules used to determine secondary alarms with mfm transport function.

SECONDARY_TRANSPORT

IF the condition of any mfmtransport is low_pressure
AND (IF there exists a mfmsource connected to the
mfmtransport
THEN (the condition of the mfmsource connected to the
mfmtransport is low_pressure
OR the condition of the mfmsource connected to the
mfmtransport is low_flow_low_pressure
OR the condition of the mfmsource connected to the
mfmtransport is low_flow))
THEN inform the operator for the next 2 seconds that
"Transport [the name of the mfmtransport] has low pressure"
AND invoke source rules
AND conclude that the state of the mfmtransport is secondary

IF the condition of any mfmtransport is low_flow
AND (IF there exists a mfmsource connected to the
mfmtransport
THEN (the condition of the mfmsource connected to the
mfmtransport is low_flow_low_pressure
OR the condition of the mfmsource connected to the
mfmtransport is low_pressure))
THEN inform the operator for the next 2 seconds that
"transport [the name of the mfmtransport] has low flow - check sensors"
AND invoke source rules

```
   AND conclude that the state of the mfmtransport is
       secondary

IF the condition of any mfmtransport is low_flow_low_pressure
AND (IF there exists a mfmsource connected to the
          mfmtransport
      THEN (the condition of the mfmsource connected to the
             mfmtransport is low_flow_low_pressure
          OR the condition of the mfmsource connected to the
             mfmtransport is low_flow
          OR the condition of the mfmsource connected to the
          mfmtransport is low_pressure))
   THEN inform the operator for the next 2 seconds that
       "transport [the name of the mfmtransport] has low flow"
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
       secondary


IF the condition of any mfmtransport is low_flow
AND (IF there exists a mfmstorage connected at the input of
             the mfmtransport
   THEN (the condition of the mfmstorage connected at the
       input of the mfmtransport is high_level
   OR the condition of the mfmstorage connected at the input
       of the mfmtransport is high_pressure
   OR the condition of the mfmstorage connected at the input
       of the mfmtransport is high_level_high_pressure
   OR  (IF the condition of the mfmstorage connected at the
       input of the mfmtransport is no_sensory_data
       THEN reflection_secondary(the condition of the
       mfmtransport,the flow_function connected at the input
       of the mfmtransport ,input)) is faulty))
   THEN inform the operator for the next 2 seconds that
       "transport [the name of the mfmtransport] has low flow
   AND invoke sink rules
```

AND conclude that the state of the mfmtransport is secondary

IF the condition of any mfmtransport is low_flow
AND (IF there exists a mfmstorage connected at the output of the mfmtransport
THEN (the condition of the mfmstorage connected at the output o the mfmtransport is low_level
OR the condition of the mfmstorage connected at the output of the mfmtransport is high_pressure
OR the condition of the mfmstorage connected at the output of the mfmtransport is low_level_high_pressure
OR (IF the condition of the mfmstorage connected at the output of the mfmtransport is no_sensory_data
THEN reflection_secondary(the condition of the mfmtransport,the flow_function connected at the output of the mfmtransport ,output)) is faulty))
THEN inform the operator for the next 2 seconds that "transport [the name of the mfmtransport] has low flow
AND invoke sink rules
AND conclude that the state of the mfmtransport is secondary

IF the condition of any mfmtransport low_flow_high_pressure
AND (IF there exists a mfmstorage connected at the input of the mfmtransport
THEN (the condition of the mfmstorage connected at the input of the mfmtransport is high_pressure
OR the condition of the mfmstorage connected at the input of the mfmtransport is high_level_high_pressure
OR the condition of the mfmstorage connected at the input of the mfmtransport is high_level
OR (IF the condition of the mfmstorage connected at the input of the mfmtransport is no_sensory_data
THEN reflection_secondary(the condition of the

mfmtransport,the flow_function connected at the input of the mfmtransport ,input)) is faulty))

THEN inform the operator for the next 2 seconds that "transport [the name of the mfmtransport] has low flow and high pressure "

AND invoke sink rules

AND conclude that the state of the mfmtransport is secondary

IF the condition of any mfmtransport is high_flow

AND (IF there exists a mfmstorage connected at the output of the mfmtransport

THEN (the condition of the mfmstorage connected at the output of the mfmtransport is high_level

OR the condition of the mfmstorage connected at the output of the mfmtransport is high_pressure

OR the condition of the mfmstorage connected at the output of the mfmtransport is high_level_high_pressure

OR the condition of the mfmstorage connected at the output of the mfmtransport is high_level_low_pressure

OR (IF the condition of the mfmstorage connected at the output of the mfmtransport is no_sensory_data

THEN reflection_secondary(the condition of the mfmtransport,the flow_function connected at the output of the mfmtransport ,output)) is faulty))

THEN inform the operator for the next 2 seconds that "transport [the name of the mfmtransport] has high level"

AND invoke source rules

AND invoke storage rules

AND conclude that the state of the mfmtransport is secondary

IF the condition of any mfmtransport is high_flow

AND (IF there exists a mfmstorage connected at the input of the mfmtransport

```
    THEN (the condition of the mfmstorage connected at the
        input of the mfmtransport is high_level
        OR the condition of the mfmstorage connected to the
            input of the mfmtransport is high_pressure
        OR the condition of the mfmstorage connected at the
        input of the mfmtransport is high_level_high_pressure
        OR the condition of the mfmstorage connected at the
            input of the mfmtransport is
            high_level_low_pressure
        OR  (IF the condition of the mfmstorage connected at
            the input of the mfmtransport is no_sensory_data
            THEN reflection_secondary(the condition of the
                mfmtransport,the flow_function connected at the
                input of the mfmtransport ,input)) is faulty))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has high
        level"
    AND invoke source rules
    AND invoke storage rules
    AND conclude that the state of the mfmtransport is
        secondary


IF the condition of any mfmtransport is
high_flow_low_pressure
AND (IF there exists a mfmstorage connected at the output of
            the mfmtransport
    THEN (the condition of the mfmstorage connected at the
        output of the mfmtransport is low_pressure
        OR the condition of the mfmstorage connected at the
            output of the mfmtransport is high_level
            low_pressure
        OR the condition of the mfmstorage connected at the
            output of the mfmtransport is high_level
        OR the condition of the mfmstorage connected at the
            output of the mfmtransport is
            high_level_high_pressure
```

```
    OR  (IF the condition of the mfmstorage connected at
        the output of the mfmtransport is no_sensory_data
        THEN reflection_secondary(the condition of the
            mfmtransport,the flow_function connected at the
            output of the mfmtransport ,output)) is
            faulty))
  THEN inform the operator for the next 2 seconds that
    "transport [the name of the mfmtransport] has high flow
    and low pressure "
  AND invoke sink rules
  AND conclude that the state of the mfmtransport is
    secondary


IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmstorage connected at the output of
    the mfmtransport
    THEN (the condition of the mfmstorage connected at the
                output of the mfmtransport is low_level
    OR the condition of the mfmstorage connected at the
        output of the mfmtransport is high_pressure
    OR the condition of the mfmstorage connected at the
        output of the mfmtransport is
        low_level_high_pressure
    OR the condition of the mfmstorage connected at the
        output of the mfmtransport is
        high_level_high_pressure
    OR  (IF the condition of the mfmstorage connected at
        the output of the mfmtransport is no_sensory_data
        THEN reflection_secondary(the condition of the
            mfmtransport,the flow_function connected at the
            output of the mfmtransport ,output)) is
            faulty))
  THEN inform the operator for the next 2 seconds that
    "transport [the name of the mfmtransport] has high
    pressure "
  AND invoke sink rules
```

```
    AND conclude that the state of the mfmtransport is
        secondary


IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmstorage connected at the input of
        the mfmtransport
        THEN (the condition of the mfmstorage connected at the
                        input of the mfmtransport is high_level
        OR the condition of the mfmstorage connected at the
            input of the mfmtransport is high_pressure
        OR the condition of the mfmstorage connected at the
            input of the mfmtransport is
            high_level_high_pressure
        OR  (IF the condition of the mfmstorage connected at
            the input of the mfmtransport is no_sensory_data
            THEN reflection_secondary(t' condition of the
                mfmtransport,the flow_function connected at the
                input of the mfmtransport ,input)) is faulty))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has
        high_pressure "
    AND invoke sink rules
    AND conclude that the state of the mfmtransport is
        secondary

IF the condition of any mfmtransport is
    high_flow_low_pressure
AND (IF there exists a mfmstorage connected at the input of
            the mfmtransport
        THEN (the condition of the mfmstorage connected at the
                        input of the mfmtransport is
                        low_pressure
        OR the condition of the mfmstorage connected at the
            input of the mfmtransport is
            high_level_low_pressure
        OR the condition of the mfmstorage connected at the
```

input of the mfmtransport is high_level
OR (IF the condition of the mfmstorage connected at the input of the mfmtransport is no_sensory_data
THEN reflection_secondary(the condition of the mfmtransport,the flow_function connected at the input of the mfmtransport ,input)) is faulty))
THEN inform the operator for the next 2 seconds that "transport [the name of the mfmtransport] has high flow and low pressure"
AND invoke sink rules
AND conclude that the state of the mfmtransport is secondary

IF the condition of any mfmtransport is low_flow_high_pressure
AND (IF there exists a mfmstorage connected at the output of the mfmtransport
THEN (the condition of the mfmstorage connected at the output of the mfmtransport is high_pressure
OR the condition of the mfmtransport is low_level_high_pressure
OR the condition of the mfmstorage connected at the output of the mfmtransport is low_level
OR (IF the condition of the mfmstorage connected at the output of the mfmtransport is no_sensory_data
THEN reflection_secondary(the condition of the mfmtransport,the flow_function connected at the output of the mfmtransport ,output)) is faulty))
THEN inform the operator for the next 2 seconds that "transport [the name of the mfmtransport] has low flow and high pressure"
AND invoke sink rules
AND conclude that the state of the mfmtransport is secondary

```
IF the condition of any mfmtransport is
   high_flow_high_pressure
AND (IF there exists a mfmstorage connected at the output of
     the mfmtransport
   THEN (the condition of the mfmstorage connected at the
              output of the mfmtransport is high_pressure
      OR the condition of the mfmstorage connected at the
          output of the mfmtransport is high_level
      OR the condition of the mfmstorage connected at the
          output of the mfmtransport is
          high_level_high_pressure
      OR  (IF the condition of the mfmstorage connected at
          the output of the mfmtransport is no_sensory_data
      THEN reflection_secondary(the condition of the
          mfmtransport,the flow_function connected at the
          output of the mfmtransport ,output)) is faulty))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has high
      level and high pressure"
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      secondary


IF the condition of any mfmtransport is low_flow_low_pressure
AND (IF there exists a mfmstorage connected at the input of
         the mfmtransport
      THEN (the condition of the mfmstorage connected at the
                        input of the mfmtransport is
                        low_pressure
      OR the condition of the mfmstorage connected at the
          input of the mfmtransport is low_level
      OR the condition of the mfmstorage connected at the
          input of the mfmtransport is low_level_low_pressure
      OR  (IF the condition of the mfmstorage connected at
```

```
        the input of th  mfmtransport is no_sensory_data
        THEN reflection_secondary(the condition of the
            mfmtransport,the flow_f...tion connected at the
            input of the mfmtransport ,input)) is faulty))
    THEN conclude that the state of the mfmtransport is
        secondary

IF the condition of any mfmtransport is
    low_flow_high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
    THEN (the condition of the mfmsink connected to the
                mfmtransport is high_pressure
        OR the condition of the mfmsink connected to the
            mfmtransport is low_flow_high_pressure
        OR the condition of the mfmsink connected to the
            mfmtransport is low_flow))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has low flow
        and high pressure"
    AND invoke sink rules
    AND conclude that the state of the mfmtransport is
        secondary

IF the condition of any mfmtransport is
    high_flow_high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
    THEN (the condition of the mfmsink connected to the
                mfmtransport is high_pressure
        OR the condition of the mfmsink connected to the
            mfmtransport is high_flow
        OR the condition of the mfmsink connected to the
            mfmtransport is high_flow_high_pressure))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has high flow
        and high pressure"
```

```
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      secondary


IF the condition of any mfmtransport is
   high_flow_low_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
                        mfmtransport is low_pressure
            OR the condition of the mfmsink connected to the
                  mfmtransport is high_flow_low_pressure
            OR the condition of the mfmsink connected to the
                  mfmtransport is high_flow))
   THEN inform the operator for the next 2 seconds that
      transport [the name of the mfmtransport] has high flow
      and low pressure"
   AND invoke sink rules
   AND conclude that the state of the mfmtransport is
      secondary

IF the condition of any mfmtransport is low_flow
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
                        mfmtransport is low_flow
            OR the condition of the mfmsink connected to the
                  mfmtransport is high_pressure
            OR the condition of the mfmsink connected to the
                  mfmtransport is low_flow_high_pressure))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has low flow
   AND invoke sink rules
   AND conclude that the state of the mfmtransport is
      secondary
```

```
IF the condition of any mfmtransport is low_flow
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
                  mfmtransport is low_flow_low_pressure))
  THEN inform the operator for the next 2 seconds that
     "transport [the name of the mfmtransport] has low flow"
  AND invoke storage rules
  AND invoke source rules
  AND conclude that the state of the mfmtransport is
     secondary


IF the condition of any mfmtransport is high_flow
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
                  mfmtransport is high_flow
            OR the condition of the mfmsink connected to the
                  mfmtransport is high_pressure
            OR the condition of the mfmsink connected to the
                  mfmtransport is high_flow_high_pressure
            OR the condition of the mfmsink connected to the
                  mfmtransport is  high_flow_low_pressure))
  THEN inform the operator for the next 2 seconds that
     "transport [the name of the  mfmtransport] has high
     flow"
  AND invoke source rules
  AND invoke storage rules
  AND conclude that the state of the mfmtransport is
     secondary


IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
                  mfmtransport is high_flow
```

```
                OR the condition of the mfmsink connected to
                    the mfmtransport is high_pressure
                OR the condition of the mfmsink connected to
                    the mfmtransport is low_flow_high_pressure
                OR the condition of the mfmsink connected to
                    the mfmtransport is
                    high_flow_high_pressure))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has high
        pressure "
    AND invoke storage rules
    AND conclude that the state of the mfmtransport is
        secondary

IF the condition of any mfmtransport low_flow_high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
        THEN (the in_condition of the mfmconnection connected
                    at the input of the mfmtransport is
                    low_pressure
            OR the in_condition of the mfmconnection connected
                at the input of the mfmtransport is
                low_flow_low_pressure))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has high
        level and high pressure"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        secondary


IF the condition of any mfmtransport is low_pressure
AND (IF there exists a mfmconnection connected at the input
            of the mfmtransport
        THEN (the in_condition of the mfmconnection connected
                    at the input of the mfmtransport is
                    low_pressure
```

```
        OR the in_condition of the mfmconnection connected
        at the input of the mfmtranspprt is
        low_flow_low_pressure))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has high
      level and high pressure"
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      secondary

IF the condition of any mfmtransport is low_flow_low_pressure
AND (IF there exists a mfmconnection connected at the input
      of the mfmtransport
      THEN (the in_condition of the mfmconnection connected
                  at the input of the mfmtransport is
                  low_pressure
         OR the in_condition of the mfmconnection connected
            at the input of the mfmtransport is
            low_flow_low_pressure))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has high
      level and high pressure"
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      secondary

IF the condition of any mfmtransport is high flow
AND (IF there exists a mfmconnection connected at the output
      of the mfmtransport
      THEN (the out_condition of the mfmconnection connected
                  at the output of the mfmtransport is
                  high_pressure
         OR the out_condition of the mfmconnection connected
```

```
                at the  tput of the mfmtransport is high_flow
            OR the out_condition of the mfmconnection connected
                at the output of the mfmtransport is
                high_flow_high_pressure))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has
        high_level and high pressure"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        secondary


IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmconnection connected at the output
        of the mfmtransport
        THEN (the out_condition of the mfmconnection connected
                        at the output of the mfmtransport is
                        high_pressure
            OR the out_condition of the mfmconnection connected
                at the output of the mfmtransport is high_flow
            OR the out_condition of the mfmconnection connected
                at the output of the mfmtransport is
                high_flow_high_pressure))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has high
        level and high pressure"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        secondary


IF the condition of any mfmtransport is
    high_flow_high_pressure
AND (IF there exists a mfmconnection connected at the output
        of the mfmtransport
```

```
      THEN (the out_condition of the mfmconnection connected
                  at the output of the mfmtransport is
                  high_pressure
          OR the out_condition of the mfmconnection connected
              at the output of the mfmtransport is high_flow
          OR the out_condition of the mfmconnection connected
              at the output of the mfmtransport is
              high_flow_low_pressure))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has high
      level and high pressure "
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      secondary


IF the condition of the mfmtransport is
   high_flow_low_pressure
AND (IF there is a mfmconnection connected at the output of
      the mfmtransport
      THEN (the out_condition of the mfmconnection connected
                  at the output of the mfmtransport is
                  high_flow
          OR the out_condition of the mfmconnection connected
              at the output of the mfmtransport is
              high_flow_low_pressure))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has high
      level and high pressure"
   AND invoke storage
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      secondary


IF the condition of any mfmtransport is
low_flow_high_pressure
```

```
AND (IF there exists a mfmconnection connected at that output
      of the mfmtransport
    THEN (the out_condition of the mfmconnection connected
                  at the output of the mfmtransport is
                 high_pressure
        OR the out_condition of the mfmconnection connected
             at the output of the mfmtransport is
            low_flow_low_pressure))
  THEN inform the operator for the next 2 seconds that
    "transport [the name of the mfmtransport] has high
     level and high pressure"
  AND invoke storage rules
  AND invoke source rules
  AND conclude that the state of the mfmtransport is
      secondary
```

F.3 Storage Rules

Rules used to determine secondary alarms with mfm storage function.

SECONDARY_STORAGE

IF the condition of any mfmstorage is low_level_low_pressure
AND (IF there exists a mfmtransport connected at the input of the mfmstorage
THEN (the condition of the mfmtransport connected at the input of the mfmstorage is low_pressure
OR the condition of the mfmtransport connected at the input of the mfmstorage is low_flow_low_pressure
OR the condition of the mfmtransport connected at the input of the mfmstorage is low_flow
OR (IF the condition of the mfmtransport connected at the input of the mfmstorage is no_sensory_data
THEN reflection_secondary(the condition of the mfmstorage,the flow_function connected at the input of the mfmstorage ,input)) is faulty))
THEN inform the operator for the next 2 seconds that "storage [the name of the mfmstorage] has low level and low pressure "
AND invoke transport rules
AND conclude that the state of the mfmstorage is secondary

IF the condition of any mfmstorage is low_level
AND (IF there exists a mfmtransport connected at the input of the mfmstorage
THEN (the condition of the mfmtransport connected at the input of the mfmstorage is low_flow
OR the condition of the mfmtransport connected at the input of the mfmstorage is low_pressure
OR the condition of the mfmtransport connected at the

```
        input of the mfmstorage is low_flow_low_pressure
    OR  (IF the condition of the mfmtransport connected at
        the input of the mfmstorage is no_sensory_data
        THEN reflection_secondary(the condition of the
            mfmstorage,the flow_function connected at the
            input of the mfmstorage ,input)) is faulty))
  THEN inform the operator for the next 2 seconds that
    "storage [the name of the mfmstorage] has low level"
  AND invoke transport rules
  AND conclude that the state of the mfmstorage is secondary


IF the condition of any mfmstorage is high_level
AND (IF there exists a mfmtransport connected at the output
        of the mfmstorage
    THEN (the condition of the mfmtransport connected at
        the output of the mfmstorage is high_flow
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is high_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is high_flow_high_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is high_flow_low_pressure
    OR  (IF the condition of the mfmtransport connected at
        the output of the mfmstorage is no_sensory_data
        THEN reflection_secondary(the condition of the
            mfmstorage,the flow_function connected at the
            output of the mfmstorage ,output)) is faulty))
  THEN inform the operator for the next 2 seconds that
    "storage [the name of the mfmstorage]  has high flow "
  AND invoke source rules
  AND invoke transport rules
  AND conclude that the state of the mfmstorage is secondary


IF the condition of any mfmstorage is
  high_level_high_pressure
```

```
AND (IF there exists a mfmtransport connected at the output
        of the mfmstorage
    THEN (the condition of the mfmtransport connected at
        the output of the mfmstorage is high_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is high_flow
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is high_flow_high_pressure
    OR  (IF the condition of the mfmtransport connected at
        the output of the mfmstorage is no_sensory_data
        THEN reflection_secondary(the condition of the
            mfmstorage,the flow_function connected at the
            output of the mfmstorage ,output)) is faulty))
  THEN inform the operator for the next 2 seconds that
    "storage [the name of the mfmstorage] has high flow and
    high pressure."
  AND invoke transport rules
  AND invoke source rules
  AND conclude that the state of the mfmstorage is secondary


IF the condition of any mfmstorage is high_pressure
AND (IF there exists a mfmtransport connected at the output
        of the mfmstorage
    THEN (the condition of the mfmtransport connected at
        the output of the mfmstorage is high_flow
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is high_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is high_flow_high_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is high_flow_low_pressure
    OR  (IF the condition of the mfmtransport connected at
        the output of the mfmstorage is no_sensory_data
        THEN reflection_secondary(the condition of the
            mfmstorage,the flow_function connected at the
            output of the mfmstorage ,output)) is faulty))
```

```
    THEN inform the operator for the next 2 seconds that
        "storage [the name of the mfmstorage] has high
        pressure"
    AND invoke transport rules
    AND conclude that the state of the mfmstorage is secondary


IF the condition of any mfmstorage is low_pressure
AND (IF there exists a mfmtransport connected at the input of
        the mfmstorage
    THEN (the condition of the mfmtransport connected at
        the input of the mfmstorage is low_pressure
    OR the condition of the mfmtransport connected at the
        input of the mfmstorage is low_flow_low_pressure
    OR the condition of the mfmtransport connected at the
        input of the mfmstorage is low_flow
    OR (IF the condition of the mfmtransport connected at
        the input of the mfmstorage is no_sensory_data
        THEN reflection_secondary(the condition of the
            mfmstorage,the flow_function connected at the
            input of the mfmstorage ,input)) is faulty))
    THEN inform the operator for the next 2 seconds that"
        storage [the name of the mfmstorage] has low pressure
        and pressure"
    AND invoke transport rules
    AND conclude that the state of the mfmstorage is secondary
```

F.4 Calculation Rules

Rules used to determine secondary alarms with mfm calculation function.

SECONDARY_CALCULATION

IF the condition of any mfmcalc is low_flow_low_pressure AND
(IF there exists a mfmtransport connected at the input of
the mfmcalc
THEN (the condition of the mfmtransport connected at
the input of the mfmcalc is low_pressure
OR the condition of the mfmtransport connected at the
input of the mfmcalc is low_flow_low_pressure
OR the condition of the mfmtransport connected at the
input of the mfmcalc is low_flow
OR (IF the condition of the mfmtransport connected at
the input of the mfmcalc is no_sensory_data
THEN reflection_secondary(the condition of the
mfmcalc,the flow_function connected at the
input of the mfmcalc ,input)) is faulty))
THEN inform the operator for the next 2 seconds that
"storage [the name of the mfmcalc] has low flow and low
pressure "
AND invoke transport rules
AND conclude that the state of the mfmcalc is secondary

IF the condition of any mfmcalc is low_flow
AND (IF there exists a mfmtransport connected at the input of
the mfmcalc
THEN (the condition of the mfmtransport connected at
the input of the mfmcalc is low_flow
OR the condition of the mfmtransport connected at the
input of the mfmcalc is low_pressure
OR the condition of the mfmtransport connected at the

```
            input of the mfmcalc is low_flow_low_pressure
        OR  (IF the condition of the mfmtransport connected at
            the input of the mfmcalc is no_sensory_data
            THEN reflection_secondary(the condition of the
                mfmcalc,the flow_function connected at the
                input of the mfmcalc ,input) is faulty))
    THEN inform the operator for the next 2 seconds that
        "storage [the name of the mfmcalc] has low flow"
    AND invoke transport rules
    AND conclude that the state of the mfmcalc is secondary


IF the condition of any mfmcalc is high_flow
AND (IF there exists a mfmtransport connected at the output
            of the mfmcalc
        THEN (the condition of the mfmtransport connected at
            the output of the mfmcalc is high_flow
        OR the condition of the mfmtransport connected at the
            output of the mfmcalc is high_pressure
        OR the condition of the mfmtransport connected at the
            output of the mfmcalc is high_flow_high_pressure
        OR the condition of the mfmtransport connected at the
            output of the mfmcalc is high_flow_low_pressure
        OR  (IF the condition of the mfmtransport connected at
            the output of the mfmcalc is no_sensory_data
            THEN reflection_secondary(the condition of the
                mfmcalc,the flow_function connected at the
                output of the mfmcalc ,output)) is faulty))
    THEN inform the operator for the next 2 seconds that
        "storage [the name of the mfmcalc]  has high flow "
    AND invoke source rules
    AND invoke transport rules
    AND conclude that the state of the mfmcalc is secondary


IF the condition of any mfmcalc is ~ high_flow_high_pressure
AND (IF there exists a mfmtransport connected at the output
```

```
        of the mfmcalc
      THEN (the condition of the mfmtransport connected at
          the output of the mfmcalc is high_pressure
        OR the condition of the mfmtransport connected at the
          output of the mfmcalc is high_flow
        OR the condition of the mfmtransport connected at the
          output of the mfmcalc is high_flow_high_pressure
        OR  (IF the condition of the mfmtransport connected at
          the qutput of the mfmcalc is no_sensory_data
          THEN reflection_secondary(the condition of the
            mfmcalc,the flow_function connected at the
            output of the mfmcalc ,output)) is faulty))
    THEN inform the operator for the next 2 seconds that
      "storage [the name of the mfmcalc] has high flow and
      high pressure "
    AND invoke transport rules
    AND invoke source rules
    AND conclude that the state of the mfmcalc is secondary


IF the condition of any mfmcalc is high_pressure
AND (IF there exists a mfmtransport connected at the output
        of the mfmcalc
      THEN (the condition of the mfmtransport connected at
          the output of the mfmcalc is high_flow
        OR the condition of the mfmtransport connected at the
          output of the mfmcalc is high_pressure
        OR the condition of the mfmtransport connected at the
          output of the mfmcalc is high_flow_high_pressure
        OR the condition of the mfmtransport connected at the
          output of the mfmcalc is high_flow_low_pressure
        OR  (IF the condition of the mfmtransport connected at
          the output of the mfmcalc is no_sensory_data
          THEN reflection_secondary(the condition of the
            mfmcalc,the flow_function connected at the
            output of the mfmcalc ,output)) is faulty))
    THEN inform the operator for the next 2 seconds that
```

"storage [the name of the mfmcalc] has high pressure"
AND invoke transport rules
AND conclude that the state of the mfmcalc is secondary

IF the condition of any mfmcalc is low_pressure
AND (IF there exists a mfmtransport connected at the input of the mfmcalc
THEN (the condition of the mfmtransport connected at the input of the mfmcalc is low_pressure
OR the condition of the mfmtransport connected at the input of the mfmcalc is low_flow_low_pressure
OR the condition of the mfmtransport connected at the input of the mfmcalc is low_flow
OR (IF the condition of the mfmtransport connected at the input of the mfmcalc is no_sensory_data
THEN reflection_secondary(the condition of the mfmcalc,the flow_function connected at the input of the mfmcalc ,input)) is faulty))
THEN inform the operator for the next 2 seconds that"
storage [the name of the mfmcalc] has low pressure and pressure"
AND invoke transport rules
AND conclude that the state of the mfmcalc is secondary

F.5 Sink Rules

Rules used to determine secondary alarms with mfm sink function.

SECONDARY_SINK

```
IF the condition of any mfmsink is high_flow
AND (the condition of the mfmtransport connected to the
        mfmsink is high_flow
     OR the condition of the mfmtransport connected to the
        mfmsink is high_flow_high_pressure
     OR the condition of the mfmtransport connected to the
        mfmsink is high_flow_low_pressure
     OR  (IF the condition of the mfmtransport connected to
        the mfmsink is no_sensory_data
        THEN reflection_secondary(the condition of the
           mfmcalc,the flow_function connected to the
           mfmsink ,input)) is faulty))
  THEN inform the operator for the next 2 seconds that "sink
     [the name of the mfmsink] has high flow"
  AND invoke transport rules
  AND concludes that the state of the mfmsink is secondary


IF the condition of any mfmsink is low_flow
AND ((the condition of the mfmtransport connected to the
        mfmsink is low_flow
     OR the condition of the mfmtransport connected to the
        mfmsink is low_pressure
     OR the condition of the mfmtransport connected to the
           mfmsink is low_flow_low_pressure
     OR  (IF the condition of the mfmtransport connected to
        the mfmsink is no_sensory_data
        THEN reflection_secondary(the condition of the
           mfmcalc,the flow_function connected to the
```

mfmsink ,input)) is faulty))
THEN inform the operator for the next 2 seconds that
"sink [the name of the mfmsink] has low flow"
AND invoke transport rules
AND conclude that the state of the mfmsink is secondary

IF the condition of any mfmsink is high_flow_low_pressure
AND ((the condition of the mfmtransport connected to the
mfmsink is low_pressure
OR the condition of the mfmtransport connected to the
mfmsink is high_flow_low_pressure
OR the condition of the mfmtransport connected to the
mfmsink is high_flow
OR (IF the condition of the mfmtransport connected to
the mfmsink is no_sensory_data
THEN reflection_secondary(the condition of the
mfmcalc,the flow_function connected to the
mfmsink ,input)) is faulty))
THEN inform the operator for the next 2 seconds that "sink
[the name of the mfmsink] has high flow and low
pressure "
AND invoke transport rules
AND conclude that the state of the mfmsink is secondary

IF the condition of any mfmsink is high_flow_high_pressure
AND (the condition of the mfmtransport connected to the
mfmsink is high_pressure
OR the condition of the mfmtransport connected to the
mfmsink is high_flow
OR the condition of the mfmtransport connected to the
mfmsink is high_flow_high_pressure
OR (IF the condition of the mfmtransport connected to
the mfmsink is no_sensory_data
THEN reflection_secondary(the condition of the
mfmcalc,the flow_function connected to the

```
                mfmsink ,input)) is faulty)
  THEN inform the operator for the next 2 seconds that "sink
     [the name of the mfmsink] has high flow and high
     pressure"
  AND invoke transport rules
  AND conclude that the state of the mfmsink is secondary


IF the condition of any mfmsink is low_flow_low_pressure
AND ((the condition of the mfmtransport connected to the
        mfmsink is low_flow_low_pressure
     OR the condition of the mfmtransport connected to the
        mfmsink is low_pressure
     OR the condition of the mfmtransport connected to the
        mfmsink is low_flow
     OR  (IF the condition of the mfmtransport connected to
        the mfmsink is no_sensory_data
        THEN reflection_secondary(the condition of the
           mfmcalc,the flow_function connected to the
           mfmsink ,input)) is faulty))
  THEN inform the operator for the next 2 seconds that "sink
     [the name of the mfmsink] has low flow and low
     pressure"
  AND invoke transport rules
  AND conclude that the state of the mfmsink is secondary


IF the condition of any mfmsink is low_pressure
AND ((the condition of the mfmtransport connected to the
        mfmsink is low_pressure
     OR the condition of the mfmtransport connected to the
        mfmsink is low_flow_low_pressure
     OR the condition of the mfmtransport connected to the
     mfmsink is low_flow
     OR  (IF the condition of the mfmtransport connected to
        the mfmsink is no_sensory_data
        THEN reflection_secondary(the condition of the
```

```
        mfmcalc,the flow_function connected to the
        mfmsink ,input)) is faulty))
THEN inform the operator for the next 2 seconds that "
     sink [the name of the mfmsink] has low pressure "
AND invoke transport rules
AND conclude that the state of the mfmsink is secondary
```

Appendix G

Fault Diagnostic Rules

This appendix contains the rules used for fault diagnostic.

G.1 Source Rules

Rules used to determine when a mfm source function has a fault and the type of fault.

FAULT_SOURCE

IF the condition of any mfmsource is low_flow_high_pressure
AND (the condition of the mfmtransport connected to the
mfmsource is low_flow_low_pressure
OR the condition of the mfmtransport connected to the
mfmsource is low_flow
OR the condition of the mfmtransport connected to the
mfmsource is low_pressure
OR (IF the condition of the mfmtransport connected to
the mfmsource is no_sensory_data
THEN reflection(the flow_function connected at the
output of the mfmsource,output,blockage))is faulty)

THEN inform the operator for the next 2 seconds that

"source [the name of the mfmsource] has low flow and high pressure blocked "
AND invoke transport rules
AND conclude that the state of the mfmsource is fault_blockage
AND start set_manager (the state of the mfmsource,"[the name of the mfmsource]","")

IF the condition of any mfmsource is low_flow_low_pressure
THEN inform the operator for the next 2 seconds that "source [the name of the mfmsource] has low flow and low pressure - inlet blocked"
AND conclude that the state of the mfmsource is fault_blockage
AND start set_manager (the state of the mfmsource,"[the name of the mfmsource]","[the name of the mfmsource]")

IF the condition of any mfmsource is low_pressure
THEN inform the operator for the next 2 seconds that "source [the name of the mfmsource] has low flow and low pressure - inlet blocked "
AND conclude that the state of the mfmsource is fault_blockage
AND start set_manager (the state of the mfmsource,"[the name of the mfmsource]","[the name of the mfmsource]")

IF the condition of any mfmsource is high_pressure
AND (the condition of the mfmtransport connected to the mfmsource is low_flow_low_pressure
OR the condition of the mfmtransport connected to the mfmsource is normal
OR the condition of the mfmtransport connected to the mfmsource is low_pressure

```
      OR the condition of the mfmtransport connected to the
          mfmsource is low_flow
      OR (IF the condition of the mfmtransport connected to
          the mfmsource is no_sensory_data
          THEN reflection(the flow_function connected at the
          output of the mfmsource
          connected,output,blockage)}is faulty))
   THEN inform the operator for the next 2 seconds that
      "source [the name of the mfmsource] has high pressure"
   AND invoke transport rules
   AND conclude that the state of the mfmsource is
      fault_blockage
   AND start set_manager (the state of the mfmsource,"[the
      name of the mfmsource]","")

IF the condition of any mfmsource is high_pressure
AND the condition of the mfmtransport connected to the
   mfmsource is normal
   THEN inform the operator for the next 2 seconds that
      "source {the name of the mfmsource} has high pressure -
      check sensors"
   AND invoke transport
   AND concluded that the state of the mfmsource is fault
   AND start set_manager (the state of the mfmsource,"[the
      name of the mfmsource]","[the name of the  mfmsource]")

IF the condition of any mfmsource is high_flow
AND (the condition of the mfmtransport is
          low_flow_high_pressure
      OR the condition of the mfmtransport connected to the
          mfmtransport connected to the mfmsource is
          low_flow_low_pressure
      OR the condition of the mfmtran port connected to the
          mfmsource is low_flow
      OR the condition of the mfmtransport connected to the
```

```
        mfmsource is low_pressure
    OR (IF the condition of the mfmtransport connected to
        the mfmsource is no_sensory_data
        THEN reflection(the flow_function connected at the
        output of the mfmsource ,output,leakage))is
        faulty))
THEN inform the operator for the next 2 seconds that
    "source [the name of the mfmsource] has high flow"
AND invoke transport rules
AND conclude that the state of the mfmsource is
    fault_leakage
AND start set_manager (the state of the mfmsource,"[the
    name of the mfmsource]","[the name of the mfmsource]")


IF the condition of any mfmsource is high_flow_low_pressure
AND (the condition of the mfmtransport connected to the
        mfmsource is low_flow_low_pressure
    OR the condition of the mfmtransport connected to the
        mfmsource is low_flow
    OR the condition of the mfmtransport connected to the
        mfmsource is low_flow_high_pressure
    OR (IF the condition of the mfmtransport connected to
        the mfmsource is no_sensory_data
        THEN reflection(the flow_function connected at the
        output of the mfmsource,output,leakage))is faulty))

THEN inform the operator for the next 2 seconds that
    "source [the name of the mfmsource] has high flow and
    low pressure "
AND invoke transport rules
AND conclude that the state of the mfmsource is
    fault_leakage
AND start set_manager (the state of the mfmsource,"[the
    name of the mfmsource]","")
```

G.2 Transport Rules

Rules used to determine when a mfm transport function has a fault and the type of fault.

FAULT_TRANSPORT_SOURCE

```
IF the condition of any mfmtransport is low_flow_low_pressure
AND (IF there exists a mfmsource connected to the
          mfmtransport
     THEN (the condition of the mfmsource connected to the
          mfmtransport is high_flow_high_pressure
          OR the condition of the mfmsource connected to the
               mfmtransport is high_pressure
          OR the condition of the mfmsource connected to the
               mfmtransport is high_flow
          OR the condition of the mfmsource connected to the
               mfmtransport is high_flow_low_pressure
          OR (IF the condition of the mfmsource connected to
               the mfmtransport is no_sensory_data
               THEN reflection(the flow_function connected at
               the input of the mfmtransport, input, leakage))
               is faulty))
  THEN inform the operator for the next 2 seconds that"
       transport [the name of the mfmtransport] has low flow
       and low pressure - leak"
  AND invoke storage rules
  AND invoke source rules
  AND conclude that the state of the mfmtransport is
       fault_leakage
  AND start set_manager (the state of the mfmtransport,"",
       "[the name of the mfmtransport]")


IF the condition of any mfmtransport is low_flow_low_pressure
AND (IF there exists a mfmsource connected to the
```

```
    mfmtransport
    THEN (the condition of the mfmsource connected to the
        mfmtransport is low_flow_high_pressure
        OR the condition of the mfmsource connected to the
        mfmtransport is high_pressure
        OR (IF the condition of the mfmsource connected to
            the mfmtransport is no_sensory_data
            THEN reflection(the flow_function connected at
            the input of the mfmtransport, input,
            blockage)) is faulty))
  THEN inform the operator [the name of the mfmtransport has
      low flow and low pressure - blocked"
  AND invoke storage rules
  AND invoke source rules
  AND conclude that the state of the mfmtransport is fault
  AND start set_manager (the state of the
      mfmtransport,"","[the name of the mfmtransport]")


IF the condition of any mfmtransport is low_flow
AND (IF there exists a mfmsource connected to the
      mfmtransport
      THEN (the condition of the mfmsource connected to the
            mfmtransport is high_flow_high_pressure
          OR the mfmtransport is high_flow
          OR the condition of the mfmsource connected to the
            mfmtransport is high_flow_low_pressure
          OR the condition of the mfmsource connected to the
            mfmtransport is high_pressure
          OR (IF the condition of the mfmsource connected to
            the mfmtransport is no_sensory_data
            THEN reflection(the flow_function connected at
            the input of the mfmtransport, input, leakage))
            is faulty))
  THEN inform the operator for the next 2 seconds that
      "transport] has low flow - leak"
  AND invoke source rules
```

```
    AND conclude that the state of the mfmtransport is
        fault_leakage
    AND start set_manager (the state of the
        mfmtransport,"","[the name of the mfmtransport]")

IF the condition of any mfmtransport is
low_flow_high_pressure
AND (IF there exists a mfmsource connected to the
        mfmtransport
        THEN (the condition of the mfmsource connected to the
                        mfmtransport is high_flow_high_pressure
            OR the condition of the mfmsource connected to the
                mfmtransport is high_flow
            OR the condition of the mfmsource connected to the
                mfmtransport is high_flow_low_pressure
            OR (IF the condition of the mfmsource connected to
                the mfmtransport is no_sensory_data
                THEN reflection(the flow_function connected at
                the input of the mfmtransport, input, leakage))
                is faulty))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has low flow
        - leak"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        fault_leakage
    AND start set_manager (the state of the mfmtransport
        ,"","[the name of the mfmtransport]")


IF the condition of any mfmtransport is low_pressure
AND (IF there exists a mfmsource connected to the
        mfmtransport
        THEN (the condition of the mfmsource connected to the
                        mfmtransport is high_flow_high_pressure
            OR the condition of the mfmsource connected to the
```

```
            mfmtransport is high_flow
          OR the condition of the mfmsource connected to the
            mfmtransport is high_flow_low_pressure
          OR (IF the condition of the mfmsource connected to
            the mfmtransport is no_sensory_data
            THEN reflection(the flow_function connected at
            the input of the mfmtransport, input, leakage))
            is faulty))
  THEN inform the operator for the next 2 seconds that
    "transport [the name of the mfmtransport] has low flow
    - leak"
  AND invoke storage rules
  AND invoke source rules
  AND conclude that the state of the mfmtransport is
    fault_leakage
  AND start set_manager (the state of the mfmtransport
    ,"","[the name of the mfmtransport]")
```

FAULT_TRANSPORT_CONNECTION

```
IF the condition of any mfmtransport is high_flow
AND (IF there exists a mfmconnection connected to the
      mfmtransport
      THEN (the in_condition of the mfmconnection connected
                  to the mfmtransport is low_flow
          OR the in_condition of the mfmconnection connected
            to the mfmtransport is low_pressure
          OR the in_condition of the mfmconnection connected
            to the mfmtransport is low_flow_low_pressure
          OR the in_condition of the mfmconnection connected
            to the mfmtransport is low_flow_high_pressure
          OR the in_condition of the mfmconnection connected
            to the mfmtransport is normal))
  THEN inform the operator for the next 2 seconds that
    "transport [the name of the mfmtransport] has high flow
    - leak or check sensors"
```

```
   AND invoke sink rules
   AND conclude that the state of the mfmtransport is
       fault_leakage
   AND start set_manager (the state of the mfmtransport,"[the
       name of the mfmtransport]","")


IF the condition of any mfmtransport is
high_flow_low_pressure
AND (IF there exists a mfmconnection connected to the
       mfmtransport
       THEN (the in_condition of the mfmconnection connected
                        to the mfmtransport is
                        low_flow_low_pressure
             OR the in_condition of the mfmconnection connected
                  to the mfmtransport is low_flow))
   THEN inform the operator for the next 2 seconds that
       "transport [the name of the mfmtransport] has high flow
       and low pressure - leak"
   AND state of the mfmtransport is fault_leakage
   AND start set_manager (the state of the mfmtransport,"[the
       name of the mfmtransport","")


IF the condition of any mfmtransport is
high_flow_low_pressure
AND (IF there exists a mfmconnection connected to the
       mfmtransport
       THEN (the in_condition of the mfmtransport is
                        high_pressu
             OR the in_condition of t...  ...connection connected
                  to the mfmtransport is normal))
   THEN inform the operator for the next 2 seconds that "
       transport [the name of the mfmtransport] has high flow
       and low pressure "
   AND invoke storage rules
   AND invoke source rules
```

```
   AND conclude that the start set_manager (the state of the
      mfmtransport,"[the name of the mfmtransport]")

IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmconnection connected to the
      mfmtransport
      THEN (the in_condition of the mfmconnection connected
                        to the mfmtransport is low_pressure
            OR the in_condition of the mfmconnection connected
                  to the mfmtransport is low_flow_low_pressure))
   THEN inform the operator for the next 2 seconds that "
      transport [the name of the mfmtransport] has high
      pressure - blocked "
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      fault_blockage
   AND conclude that the state of the mfmtransport is
      fault_blockage
   AND start set_manager (the state of the mfmtransport,"[the
      name of the mfmtransport]","")

IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmconnection connected to the
      mfmtransport
      THEN (the in_condition of the mfmconnection connected
                        to the mfmtransport is low_flow
            OR the in_condition of the mfmconnection connected
                  to the mfmtransport is low_flow_high_pressure
            OR the in_condition of the mfmtransport is normal))
   THEN inform the operator for the next 2 seconds that "
      transport [the name of the mfmtransport] has high
      pressure - leak "
   AND invoke storage rules
   AND invoke source rules
```

```
    AND conclude that the state of the mfmtransport is
        fault_leakage
    AND start set_manager (the state of the mfmtransport,"
        "[the name of mfmtransport]","")


IF the condition of any mfmtransport is
high_flow_high_pressure
AND (IF there exists a mfmconnection connected to the
        mfmtransport
        THEN (the in_condition of the mfmconnection connected
                        to the mfmtransport is low_pressure
            OR the in_condition of the mfmconnection connected
                to the mfmtransport is low_flow
            OR the in_condition of the mfmconnection connected
                to the mfmtransport is low_flow_low_pressure
            OR the in_condition of the mfmconnection connected
                to the mfmtransport is low_flow_high_pressure
            OR the in_condition of the mfmconnection connected
                to the mfmtransport is normal))
    THEN inform the operator for the next 2 seconds that"
        transport [the name of the mfmtransport] has high flow
        and high pressure - leak"
    AND invoke sink rules
    AND conclude that the state of the mfmtransport is fault
        leakage
    AND start set manager (the state of the mfmtransport,"[the
        name of the mfmtransport]","")


IF the condition of any mfmtransport is
low_flow_high_pressure
AND (IF there exists a mfmconnection connected to the
        mfmtransport
        THEN (in_condition of the mfmconnection connected to
                        the mfmtransport is low_pressure
```

```
        OR the in_condition of the mfmconnection connected
            to the mfmtransport is low_flow_low_pressure
        OR the in_condition of the mfmconnection connected
            to the mfmtransport is low_flow))
THEN inform the operator for the next 2 seconds that"
    transport [the name of the mfmtransport] has low flow
    and high pressure - blocked"
AND invoke sink rules
AND conclude that the state of the mfmtransport is
    fault_blockage
AND start set_manager (the state of the mfmtransport,"[the
    name of the mfmtransport]","")


IF the condition of any mfmtransport is
low_flow_high_pressure
AND (IF there exists a mfmconnection connected to the
    mfmtransport
    THEN (the in_condition of the mfmconnection connected
                to the mfmtransport is high_flow
        OR the in_condition of the mfmconnection connected
            to the mfmtransport is high_flow_low_pressure
        OR the in_condition of the mfmconnection connected
            to the mfmtransport is high_flow_high_pressure
        OR the in_condition of the mfmconnection connected
            to the mfmtransport is high_flow
        OR the in_condition of the  mfmconnection connected
            to the mfmtransport is normal))
THEN inform the operator for the next 2 seconds that
    "transport [the name of the mfmtransport] has low flow
    and high pressure "
AND invoke source rules
AND invoke storage rules
AND conclude that the state of the mfmtransport is fault
AND start set_manager (the state of the
    mfmtransport]","[the name of the mfmtransport]")
```

FAULT_TRANSPORT_SINK

```
IF the condition of any mfmtransport is high_flow
AND (IF there exists a mfmsink connected to the mfmtransport
     THEN (the condition of the mfmsink connected to the
                    mfmtransport is low_flow
          OR the condition of the mfmsink connected to the
               mfmtransport is low_pressure
          OR the condition of mfmsink connected to the
               mfmtransport is  low_flow_low_pressure
          OR the condition of the mfmsink connected to the
               mfmtransport is low_flow_high_pressure
          OR the condition of the mfmsink connected to the
               mfmtransport is normal))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has high_flow
      - leak or check sensors"
   AND invoke sink rules
   AND conclude that the state of the mfmtransport is
      fault_leakage
   AND start set_manager (the state of the mfmtransport,"[the
      name of the mfmtransport]","")


IF the condition of any mfmtransport is
high_flow_low_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
     THEN (the  condition of the mfmsink connected to the
               mfmtransport is low_flow_low_pressure
          OR the condition of the mfmsink connected to the
               mfmtransport is low_flow))
   THEN inform the operator for the next 2 seconds that"
      transport [the name of the mfmtransport] has high flow
      and low pressure - leak"
   AND invoke sink rules
   AND conclude that the state of the mfmtransport is
```

```
    fault_leakage
  AND start set_manager (the state of the mfmtransport,"[the
    name of the mfmtransport]","")


IF the condition of any mfmtransport is
high_flow_low_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
              mfmtransport is high_pressure
          OR the condition of the mfmsink connected to the
              mfmtransport is normal))
  THEN inform the operator for the next 2 seconds that
    "transport [the name of the mfmtransport] has high flow
    and low pressure"
  AND invoke storage rules
  AND invoke source rules
  AND conclude that the state of the mfmtransport is fault
  AND start set_manager (the state of the mfmtransport,"[the
    name of the mfmtransport] ,""[the name of the
    mfmtransport]")


IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
              mfmtransport is low_pressure
          OR the condition of the mfmsink connected to the
          mfmtransport is low_flow_low_pressure))
  THEN inform the operator for the next 2 section that
    "transport [the name of the mfmtransport] has high
    pressure - blocked"
  AND invoke storage rules
  AND invoke source rules
  AND conclude that the state of the mfmtransport is
    fault_blockage
```

```
    AND start set_manager (the state of the mfmtransport,"[the
        name of the mfmtransport]","")


IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
              mfmtransport is low_flow
            OR the condition of the mfmsink connected to the
              mfmtransport is low_flow_high_pressure
            OR the condition of the mfmsink connected to the
              mfmtransport is normal))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has high
        pressure - leak"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        fault_leakage
    AND start set_manager (the state of the mfmtransport,"[the
        name of the mfmtransport]","")


IF the condition of any mfmtransport is
high_flow_high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
      THEN (the condition of the mfmsink connected to the
              mfmtransport is low_pressure
            OR the condition of the mfmsink connected to the
              mfmtransport is low_flow
            OR the condition of the mfmsink connected to the
              mfmtransport is low_flow_low_pressure
            OR the condition of the mfmsink connected to the
              mfmtransport is low_flow_high_pressure
            OR the condition of the mfmsink connected to the
              mfmtransport is normal))
    THEN inform the operator for the next 2 seconds that"
```

transport [the name of the mfmtransport] has high flow and high pressure - leak"
AND invoke sink rules
AND conclude that the state of the mfmtransport is fault_leakage
AND start set_manager (the state of the mfmtransport,"[the name of the mfmtransport]","")

IF the condition of any mfmtransport is low_flow_high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
THEN (the condition of the mfmsink connected to the mfmtransport is low _pressure
OR the condition of the mfmsink connected to the mfmtransport is low_flow_low_pressure
OR the condition of the mfmsink connected to the mfmtransport is low_flow))
THEN inform the operator for the next 2 that "transport [the name of the mfmtransport] has low_flow and high pressure - blocked"
AND invoke sink rules
AND conclude that the state of the mfmtransport is fault_blockage
AND start set_manager (the state of the mfmtransport,"[the name of the mfmtransport]","")

IF the condition of any mfmtransport is low_flow_high_pressure
AND (IF there exists a mfmsink connected to the mfmtransport
THEN (the condition of the mfmsink connected to the mfmtransport is high_flow
OR the condition of the mfmsink connected to the mfmtransport is high_flow_low_pressure
OR the condition of the mfmsink connected to the mfmtransport is high_flow_high_pressure

```
        OR the condition of the mfmsink connected to the
            mfmtransport is high_flow
        OR the condition of the mfmsink connected to the
            mfmtransport is normal))
THEN inform the operator for the next 2 seconds that
    "transport[ the name of the mfmtransport] has low flow
    and high pressure "
AND invoke source rules
AND invoke storage rules
AND conclude that the state of the mfmtransport is fault
AND start set_manager (the state of the mfmtransport,"[the
    name of the mfmtransport]","[the name of the
    mfmtransport]")
```

FAULT_TRANSPORT_CALCULATION

```
IF the condition of any mfmtransport is low_flow_low_pressure
AND (IF there exists a mfmcalc connected at the output to the
        mfmtransport
    THEN (the condition of the mfmsource connected at the
        input of the mfmtransport is
        high_flow_high_pressure
        OR the condition of the mfmcalc connected at the
            input of the mfmtransport is high_pressure
        OR the condition of the mfmcalc connected at the
            input of the mfmtransport is high_flow
        OR the condition of the mfmcalc connected at the
            input of the mfmtransport is
            high_flow_low_pressure
        OR (IF the condition of the mfmcalc connected at
            the input of the mfmtransport is
            no_sensory_data
            THEN reflection(the flow_function connected at
            the input of the mfmtransport, input, leakage))
            is faulty))
THEN inform the operator for the next 2 seconds that"
```

```
        transport [the name of the mfmtransport] has low flow
        and low pressure - leak"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        fault_leakage
    AND start set_manager (the state of the mfmtransport,"",
        "[the name of the mfmtransport]")


IF the condition of any mfmtransport is low_flow_low_pressure
AND (IF there exists a mfmcalc connected at the input of the
        mfmtransport
    THEN (the condition of the mfmcalc connected at the
            input of the mfmtransport is low_flow_high_pressure
            OR the condition of the mfmcalc connected at the
            input of the mfmtransport is high_pressure
            OR (IF the condition of the mfmcalc connected at
                the input of the mfmtransport is
                no_sensory_data
                THEN reflection(the flow_function connected at
                the input of the mfmtransport, input,
                blockage)) is faulty))
    THEN inform the operator [the name of the mfmtransport has
        low flow and low pressure - blocked"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is fault
    AND start set_manager (the state of the
        mfmtransport,"","[the name of the mfmtransport]")


IF the condition of any mfmtransport is low_flow
AND (IF there exists a mfmcalc connected at the input of the
            mfmtransport
        THEN (the condition of the mfmcalc connected at the
        input the mfmtransport is high_flow_high_pressure
```

```
        OR the condition of the mfmcalc connected at the
            input of the mfmtransport is high_flow
        OR the condition of the mfmcalc connected at the
            input of the mfmtransport is
            high_flow_low_pressure
        OR the condition of the mfmcalc connected at the
            input of the mfmtransport is high_pressure
        OR (IF the condition of the mfmcalc connected at
            the input of the mfmtransport is
            no_sensory_data
            THEN reflection(the flow_function connected at
            the input of the mfmtransport, input, leakage))
            is faulty))
    THEN inform the operator for the next 2 seconds that
        "transport] has low flow - leak"
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        fault_leakage
    AND start set_manager (the state of the
        mfmtransport,"","[the name of the mfmtransport]")

IF the condition of any mfmtransport is
low_flow_high_pressure
AND (IF there exists a mfmcalc connected at the input of the
                mfmtransport
    THEN (the condition of the mfmcalc connected at the
    input of the mfmtransport is high_flow_high_pressure
        OR the condition of the mfmcalc connected at the
            input of the mfmtransport is high_flow
        OR the condition of the mfmcalc connected at the
            input of the mfmtransport is
            high_flow_low_pressure
        OR (IF the condition of the mfmcalc connected at
            the input of the mfmtransport is
            no_sensory_data
            THEN reflection(the flow_function connected at
            the input of the mfmtransport, input, leakage))
```

```
                is faulty))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has low flow
        - leak"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        fault_leakage
    AND start set_manager (the state of the mfmtransport
        ,"","[the name of the mfmtransport]")


IF the condition of any mfmtransport is low_pressure
AND (IF there exists a mfmcalc connected at the input of the
      mfmtransport
      THEN (the condition of the mfmcalc connected at the
      input the mfmtransport is high_flow_high_pressure
          OR the condition of the mfmcalc connected at the
              input the mfmtransport is high_flow
          OR the condition of the mfmcalc connected at the
              input of the mfmtransport is
              high_flow_low_pressure
          OR the condition of the mfmcalc connected at the
              input of the mfmtransport is high_pressure
          OR (IF the condition of the mfmcalc connected at
              the input of the mfmtransport is
              no_sensory_data
              THEN reflection(the flow_function connected at
              the input of the mfmtransport, input, leakage))
              is faulty))
    THEN inform the operator for the next 2 seconds that
        "transport [the name of the mfmtransport] has low flow
        - leak"
    AND invoke storage rules
    AND invoke source rules
    AND conclude that the state of the mfmtransport is
        fault_leakage
```

```
    AND start set_manager (the state of the mfmtransport
        ,"","[the name of the mfmtransport]")


IF the condition of any mfmtransport is high_flow
AND (IF there exists a mfmcalc connected at the output the
   mfmtransport
      THEN (the condition of the mfmcalc connected at the
      output of the mfmtransport is low_flow
          OR the condition of the mfmcalc connected at the
              output of the mfmtransport is low_pressure
          OR the condition of the mfmcalc connected at the
              output of the mfmtransport is
              low_flow_low_pressure
          OR the condition of the mfmcalc connected at the
              output the mfmtransport is
              low_flow_high_pressure
          OR the condition of the mfmcalc connected at the
              output of the mfmtransport is normal
          OR (IF the condition of the mfmcalc connected at
              the output of the mfmtransport is
              no_sensory_data
              THEN reflection(the flow_function connected at
              the output of the mfmtransport, output,
              leakage)) is faulty))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has high_flow
      - leak or check sensors"
   AND invoke sink rules
   AND conclude that the state of the mfmtransport is
      fault_leakage
   AND start set_manager (the state of the mfmtransport,"[the
      name of the mfmtransport]","")


IF the condition of any mfmtransport is
```

```
    high_flow_low_pressure
AND (IF there exists a mfmcalc connected at the output of the
    mfmtransport
        THEN (the condition of the mfmcalc connected at the
            output of the mfmtransport is
            low_flow_low_pressure
            OR the condition of the mfmcalc connected at the
                output of the mfmtransport is low_flow
            OR (IF the condition of the mfmcalc connected at
                the output of the mfmtransport is
                no_sensory_data
                THEN reflection(the flow_function connected at
                the output of the mfmtransport, output,
                leakage)) is faulty))
    THEN inform the operator for the next 2 seconds that"
        transport [the name of the mfmtransport] has high flow
        and low pressure - leak"
    AND invoke sink rules
    AND conclude that the state of the mfmtransport is
        fault_leakage
    AND start set_manager (the state of the mfmtransport,"[the
        name of the mfmtransport]","")


IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmcalc connected at the output of the
    mfmtransport
        THEN (the condition of the mfmcalc connected at the
            output of the mfmtransport is low_pressure
            OR the condition of the  mfmcalc connected at the
            output of the mfmtransport is low_flow_low_pressure
            OR (IF the condition of the mfmcalc connected at
                the output of the mfmtransport is
                no_sensory_data
                THEN reflection(the flow_function connected at
                the output of the mfmtransport, output,
```

```
            blockage)) is faulty))
   THEN infor. the operator for the next 2 section that
      "transport [the name of the mfmtransport] has high
      pressure - blocked"
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      fault_blockage
   AND start set_manager (the state of the mfmtransport,"[the
      name of the mfmtransport]","")


IF the condition of any mfmtransport is high_pressure
AND (IF there exists a mfmcalc connected at the output of the
   mfmtransport
      THEN (the condition of the mfmcalc connected at the
         output of the mfmtransport is low_flow
         OR the condition of the mfmcalc connected at the
            output of the mfmtransport is
            low_flow_high_pressure
         OR the condition of the mfmcalc connected at the
            output of the mfmtransport is normal
         OR (IF the condition of the mfmcalc connected at
            the output of the mfmtransport is
            no_sensory_data
            THEN reflection(the flow_function connected at
            the output of the mfmtransport, output,
            leakage)) is faulty))
   THEN inform the operator for the next 2 seconds that
      "transport [the name of the mfmtransport] has high
      pressure - leak"
   AND invoke storage rules
   AND invoke source rules
   AND conclude that the state of the mfmtransport is
      fault_leakage
   AND start set_manager (the state of the mfmtransport,"[the
      name of the mfmtransport]","")
```

```
IF the condition of any mfmtransport is
   high_flow_high_pressure
AND (IF there exists a mfmcalc connected at the output of
   mfmtransport
      THEN (the condition of the mfmcalc connected at the
            output of the mfmtransport is low_pressure
            OR the condition of the mfmcalc connected at the
               output of the mfmtransport is low_flow
            OR the condition of the mfmcalc connected at the
               output of the mfmtransport is
               low_flow_low_pressure
            OR the condition of the mfmcalc connected at the
               output the mfmtransport is
               low_flow_high_pressure
            OR the condition of the mfmcalc connected at the
               output the mfmtransport is normal
            OR (IF the condition of the mfmcalc connected at
               the output of the mfmtransport is
               no_sensory_data
               THEN reflection(the flow_function connected at
               the output of the mfmtransport, output,
               leakage)) is faulty))
   THEN inform the operator for the next 2 seconds that"
      transport [the name of the mfmtransport] has high flow
      and high pressure - leak"
   AND invoke sink rules
   AND conclude that the state of the mfmtransport is
      fault_leakage
   AND start set_manager (the state of the mfmtransport,"[the
      name of the mfmtransport]","")


IF the condition of any mfmtransport is
   low_flow_high_pressure
AND (IF there exists a mfmcalc connected at the output of the
   mfmtransport
```

```
    THEN (the condition of the mfmcalc connected at the
        output of the mfmtransport is low_pressure
        OR the condition of the  mfmcalc connected at the
            output of the mfmtransport is
            low_flow_low_pressure
        OR the condition of the mfmcalc connected at the
            output of the mfmtransport is low_flow
        OR (IF the condition of the mfmcalc connected at
            the output of the mfmtransport is
            no_sensory_data
            THEN reflection(the flow_function connected at
            the output of the mfmtransport, output,
            blockage)) is faulty))
THEN inform the operator for the next 2 that "transport
    [the name of the mfmtransport] has low_flow and high
    pressure - blocked"
AND invoke sink rules
AND conclude that the state of the mfmtransport is
    fault_blockage
AND start set_manager (the state of the mfmtransport,"[the
    name of the mfmtransport]","")
```

G.3 Storage Rules

Rules used to determine when a mfm Storage function has a fault and the type of fault.

FAULT_STORAGE

```
IF the condition of any mfmstorage is low_level
AND (IF there exists a mfmtransport connected at the input of
        the mfmstorage
     THEN (the condition of the mfmtransport connected at
        the input of the mfmstorage is high_flow
     OR the condition of the mfmtransport connected at the
        input of the mfmstorage is high_flow_high_pressure
     OR the condition of the mfmtransport connected at the
        input of the mfmstorage is high_flow_low_pressure
     OR (IF the condition of the mfmtransport connected at
        the input of the mfmstorage is no_sensory_data
        THEN reflection(the flow_function connected at the
           input of the mfmstorage, input, leakage)) is
           faulty))
  THEN inform the operator for the next 2 seconds that"
     storage [the name of the mfmstorage] has low level -
     leak "
  AND invoke transport rules
  AND conclude that the state of the mfmstorage is
     fault_leakage
  AND start set_manager (the state of the mfmstorage,"",
     "[the name of the mfmstorag )")


IF the condition of the mfmstorage is low_level
AND (IF there exists a mfmtransport connected at the input of
        the mfmstorage
     THEN (the condition of the mfmtransport connected at
```

```
        the input of the mfmstorage is high_pressure
    OR the condition of the input of the mfmstorage is
        low_flow_high_pressure
    OR (IF the condition of the mfmtransport connected at
        the input of the mfmstorage is no_sensory_data
        THEN reflection(the flow_function connected at the
            input of the mfmstorage, input, blockage)) is
            faulty))
THEN inform the operator for the next 2 seconds that
    "storage [the name of the mfmstorage ] has low level -
    blocked "
AND invoke transport rules
AND conclude that the state of the mfmstorage is
    fault_blockage
AND start set_manager (the state of the mfmstorage,"",
    "[the name of the mfmstorage]")


IF the condition of any mfmstorage is high_level
AND (IF there exists a mfmtransport connected at the output
        of the mfmstorage
    THEN (the condition of the mfmstorage connected at the
        output of the mfmstorage is low_flow
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is low_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage id low_flow_low_pressure
    OR (IF the condition of the mfmtransport connected at
        the output of the mfmstorage is no_sensory_data
        THEN reflection(the flow_function connected at the
            output of the mfmstorage, output, blockage)) is
            faulty))
THEN inform the operator for the next 2 seconds that"
    storage [the name of the mfmstorage] has high level -
    blocked"
AND invoke transport rules
AND conclude that the state of the mfmstorage is
```

```
        fault_blockage
    AND start_set_manager (the state of the mfmstorage,"",
        "[the name of the mfmstorage]")


IF the condition of any mfmstorage is high_pressure
AND (IF there exists a mfmtransport connected at the output
        of the mfmstorage
     THEN (the condition of the mfmtransport connected at
        the output of the mfmstorage is low_pressure
      OR the condition of the mfmtransport connected at the
        output of the mfmstorage is low_flow_low_pressure
      OR the condition of the mfmtransport connected at the
        output of the mfmstorage is low_flow
      OR (IF the condition of the mfmtransport connected at
        the output of the mfmstorage is no_sensory_data
        THEN reflection(the flow_function connected at the
            output of the mfmstorage, output, blockage)) is
            faulty))
    THEN inform the operator for the next 2 seconds that"
        storage [the name of the mfmstorage] has high pressure"
    AND invoke transport rules
    AND invoke source rules
    AND conclude that the state of the mfmstorage is
        fault_blockage
    AND start set_manager (the state of the mfmstorage ,"[the
        name of the mfmstorage]","")


IF the condition of any mfmstorage is low_pressure
AND (IF there exists a mfmtransport connected at the input of
        the mfmstorage
     THEN (the condition of the mfmtransport connected at
        the input of the mfmstorage is
        high_flow_low_pressure
      OR the condition of the mfmtransport connected at the
        input of the mfmstorage is high_flow_high_pressure
```

```
    OR the condition of the mfmtransport connected at the
        input of the mfmstorage is high_flow
    OR (IF the condition of the mfmtransport connected at
        the input of the mfmstorage is no_sensory_data
        THEN reflection(the flow_function connected at the
            input of the mfmstorage, input, leakage)) is
            faulty))
  THEN inform the operator for the next 2 seconds that "
    storage [the name of the mfmstorage] has low pressure -
    leak "
  AND invoke transport rules
  AND conclude that the state of the mfmstorage is
    fault_leakage
  AND start set_manager (the state of the mfmstorage,"[the
    name of the mfmstorage]","")


IF the condition of any mfmstorage is low_pressure
AND (IF there exists a mfmtransport connected at the input of
        the mfmstorage
    THEN (the condition of the mfmtransport connected at
        the input of the mfmstorage is high_pressure
    OR the condition of the mfmtransport connected at the
        input of the mfmstorage is low_flow_high_pressure
    OR (IF the condition of the mfmtransport connected at
        the input of the mfmstorage is no_sensory_data
        THEN reflection(the flow_function connected at the
            input of the mfmstorage, input, blockage)) is
            faulty))
  THEN inform the operator for the next 2 seconds that
    "storage [the name of the mfmstorage] has low pressure
    -blocked "
  AND invoke transport rules
  AND conclude that the state of the mfmstorage is
    fault_blockage
  AND start set_manager (the state of the mfmstorage,"",
    "[the name of the mfmstorage]")
```

IF the condition of any mfmstorage is low_level_low_pressure
AND (IF there exists a mfmtransport connected at an input of
the mfmstorage
THEN (the condition of the mfmtransport connected at
the input of the mfmstorage is
high_flow_high_pressure
OR the condition of the mfmtransport connected at the
input of the mfmstorage is high_flow
OR the condition of the mfmtransport connected at the
input of the mfmstorage is high_flow_low_pressure
OR the condition of the mfmtransport connected at the
input of the mfmstorage is normal
OR (IF the condition of the mfmtransport connected at
the input of the mfmstorage is no_sensory_data
THEN reflection(the flow_function connected at the
input of the mfmstorage, input, leakage)) is
faulty))
THEN inform the operator for the next 2 seconds that
"storage [the normal of the mfmstorage] has low level
and low pressure -leak "
AND invoke transport rules
AND conclude that the state of the mfmstorage is
fault_leakage
AND start set_manager (the state of the mfmstorage ,"",
"[the name of the mfmstorage]")

IF the condition of any mfmstorage is low_level_low_pressure
AND (IF there exists a mfmtransport connected at an input of
the mfmstorage
THEN (the condition of the mfmtransport connected at
the input of the mfmstorage is high_pressure
OR the condition of the mfmtransport connected at the
input of the mfmstorage is low_flow_high_pressure
OR (IF the condition of the mfmtransport connected at
the input of the mfmstorage is no_sensory_data

```
            THEN reflection(the flow_function connected at the
                 input of the mfmstorage, input, blockage)) is
                 faulty))
    THEN inform the operator for the next 2 seconds that
       "storage [the name of the mfmstorage] has low level and
       low pressure - blocked "
    AND invoke transport rules
    AND conclude that the state of the mfmstorage is
       fault_blockage
    AND start set_manager (the state of the mfmstorage,"",
       "[the name of the mfmstorage]")


IF the condition of any mfmsto    e is
       high_level_high_pressure
AND (IF there exists a mfmtransport connected at the output
          of the mfmstorage
       THEN (the condition of the mfmtransport connected at
          the output of the mfmstorage is
          low_flow_low_pressur
       OR the condition of the    ransport connected at the
          output of  he mfmstorage is low_flow
       OR (IF the condition of the mfmtransport connected at
          the output of the mfmstorage is no_sensory_data
          THEN reflection(the flow_function connected at the
             output of the mfmstorage, output, leakage)) is
             faulty))
    THEN inform the operator for the next 2 seconds that
       "storage [the name of the mfmstorage] has high level
       and high pressure -leak "
    AND invoke transport rules
    AND conclude that the state of the mfmstorage is
       fault_leakage
    AND start set_manager (the state of the mfmstorage,"[the
       name of the mfmstorage]","")
```

```
IF the condition of any mfmstorage is
        high_level_high_pressure
AND (IF there exists a mfmtransport connected at the output
        of the mfmstorage
    THEN (the condition of the mfmtransport connected at
        the output of the mfmstorage is low_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmstorage is low_flow_low_pressure
    OR (IF the condition of the mfmtransport connected at
        the output of the mfmstorage is no_sensory_data
        THEN reflection(the flow_function connected at the
            output of the mfmstorage, output, leakage)) is
            faulty))
  THEN inform the operator for the next 2 seconds that"
    storage [the name of the mfmstorage] has high level and
    high pressure -blocked "
  AND invoke transport rules
  AND conclude that the state of the mfmstorage is
    fault_blockage
  AND start set_manager (the state of the mfmstorage,"[the
    name of the mfmstorage]","")
```

G.4 Calculation Rules

Rules used to determine when a mfm calculation function has a fault and the type of fault.

FAULT_CALCULATION

```
IF the condition of any mfmcalc is low_flow
AND (IF there exists a mfmtransport connected at the input of
        the mfmcalc
     THEN (the condition of the mfmtransport connected at
        the input of the mfmcalc is high_flow
     OR the condition of the mfmtransport connected at the
        input of the mfmcalc is high_flow_high_pressure
     OR the condition of the mfmtransport connected at the
        input of the mfmcalc is high_flow_low_pressure
     OR (IF the condition of the mfmtransport connected at
        the input of the mfmcalc is no_sensory_data
        THEN reflection(the flow_function connected at the
           input of the mfmcalc, input, leakage)) is
           faulty))
  THEN inform the operator for the next 2 seconds that"
     storage [the name of the mfmcalc] has low flow - leak "
  AND invoke transport rules
  AND conclude that the state of the mfmcalc is
     fault_leakage
  AND start set_manager (the state of the mfmcalc,"", "[the
     name of the mfmcalc]")


IF the condition of the mfmcalc is low_flow
AND (IF there exists a mfmtransport connected at the input of
        the mfmcalc
     THEN (the condition of the mfmtransport connected at
        the input of the mfmcalc is high_pressure
     OR the condition of the input of the mfmcalc is
```

```
        low_flow_high_pressure
    OR (IF the condition of the mfmtransport connected at
        the input of the mfmcalc is no_sensory_data
        THEN reflection(the flow_function connected at the
            input of the mfmcalc, input, blockage)) is
            faulty))
THEN inform the operator for the next 2 seconds that
    "storage [the name of the mfmcalc ] has low flow -
    blocked "
AND invoke transport rules
AND conclude that the state of the mfmcalc is
    fault_blockage
AND start set_manager (the state of the mfmcalc,"", "[the
    name of the mfmcalc]")


IF the condition of any mfmcalc is high_flow
AND (IF there exists a mfmtransport connected at the output
        of the mfmcalc
    THEN (the condition of the mfmcalc connected at the
        output of the mfmcalc is low_flow
    OR the condition of the mfmtransport connected at the
        output of the mfmcalc is low_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmcalc id low_flow_low_pressure
    OR (IF the condition of the mfmtransport connected at
        the output of the mfmcalc is no_sensory_data
        THEN reflection(the flow_function connected at the
            output of the mfmcalc, output, blockage)) is
            faulty))
THEN inform the operator for the next 2 seconds that"
    storage [the name of the mfmcalc] has high flow -
    blocked"
AND invoke transport rules
AND conclude that the state of the mfmcalc is
    fault_blockage
AND start set_manager (the state of the mfmcalc,"", "[the
```

```
        name of the mfmcalc]")


IF the condition of any mfmcalc is high_pressure
AND (IF there exists a mfmtransport connected at the output
          of the mfmcalc
       THEN (the condition of the mfmtransport connected at
          the output of the mfmcalc is low_pressure
       OR the condition of the mfmtransport connected at the
          output of the mfmcalc is low_flow_low_pressure
       OR the condition of the mfmtransport connected at the
          output of the mfmcalc is low_flow
       OR (IF the condition of the mfmtransport connected at
          the output of the mfmcalc is no_sensory_data
          THEN reflection(the flow_function connected at the
             output of the mfmcalc, output, leakage)) is
             faulty))
   THEN inform the operator for the next 2 seconds that"
      storage [the name of the mfmcalc] has high pressure"
   AND invoke transport rules
   AND invoke source rules
   AND conclude that the state of the mfmcalc is
      fault_leakage
   AND start set_manager (the state of the mfmcalc ,"[the
      name of the mfmcalc]","")


IF the condition of any mfmcalc is low_pressure
AND (IF there exists a mfmtransport connected at the input of
          the mfmcalc
       THEN (the condition of the mfmtransport connected at
          the input of the mfmcalc is high_flow_low_pressure

       OR the condition of the mfmtransport connected at the
          input of the mfmcalc is high_flow_high_pressure
       OR the condition of the mfmtransport connected at the
          input of the mfmcalc is high_flow
```

```
      OR (IF the condition of the mfmtransport connected at
          the input of the mfmcalc is no_sensory_data
          THEN reflection(the flow_function connected at the
              input of the mfmcalc, input, leakage)) is
              faulty))
   THEN inform the operator for the next 2 seconds that "
      storage [the name of the mfmcalc] has low pressure -
      leak "
   AND invoke transport rules
   AND conclude that the state of the mfmcalc is
      fault_leakage
   AND start set_manager (the state of the mfmcalc,"[the name
      of the mfmcalc]","")


IF the condition of any mfmcalc is low_pressure
AND (IF there exists a mfmtransport connected at the input of
          the mfmcalc
      THEN (the condition of the mfmtransport connected at
          the input of the mfmcalc is high_pressure
      OR the condition of the mfmtransport connected at the
          input of the mfmcalc is low_flow_high_pressure
      OR (IF the condition of the mfmtransport connected at
          the output of the mfmcalc is no_sensory_data
          THEN reflection(the flow_function connected at the
              output of the mfmcalc, output, blockage)) is
              faulty))
   THEN inform the operator for the next 2 seconds that
      "storage [the name of the mfmcalc] has low pressure -
      blocked "
   AND invoke transport rules
   AND conclude that the state of the mfmcalc is
      fault_blockage
   AND start set_manager (the state of the mfmcalc,"", "[the
      name of the mfmcalc]")
```

```
IF the condition of any mfmcalc is low_flow_low_pressure
AND (IF there exists a mfmtransport connected at an input of
        the mfmcalc
    THEN (the condition of the mfmtransport connected at
        the input of the mfmcalc is high_flow_high_pressure

    OR the condition of the mfmtransport connected at the
        input of the mfmcalc is high_flow
    OR the condition of the mfmtransport connected at the
        input of the mfmcalc is high_flow_low_pressure
    OR the condition of the mfmtransport connected at the
        input of the mfmcalc is normal
    OR (IF the condition of the mfmtransport connected at
        the input of the mfmcalc is no_sensory_data
        THEN reflection(the flow_function connected at the
            input of the mfmcalc, input, leakage)) is
            faulty))
  THEN inform the operator for the next 2 seconds that
    "storage [the normal of the mfmcalc] has low flow and
    low pressure -leak "
  AND invoke transport rules
  AND conclude that the state of the mfmcalc is
    fault_leakage
  AND start set_manager (the state of the mfmcalc ,"", "[the
    name of the mfmcalc]")


IF the condition of any mfmcalc is low_flow_low_pressure
AND (IF there exists a mfmtransport connected at an input of
        the mfmcalc
    THEN (the condition of the mfmtransport connected at
        the input of the mfmcalc is high_pressure
    OR the condition of the mfmtransport connected at the
        input of the mfmcalc is low_flow_high_pressure
    OR (IF the condition of the mfmtransport connected at
        the input of the mfmcalc is no_sensory_data
        THEN reflection(the flow_function connected at the
```

```
            input of the mfmcalc, input, blockage)) is
            faulty))
   THEN inform the operator for the next 2 seconds that
      "storage [the name of the mfmcalc] has low flow and low
      pressure - blocked "
   AND invoke transport rules
   AND conclude that the state of the mfmcalc is
      fault_blockage
   AND start set_manager (the state of the mfmcalc,"", "[the
      name of the mfmcalc]")


IF the condition of any mfmcalc is
      high_flow_high_pressure
AND (IF there exists a mfmtransport connected at the output
         of the mfmcalc
      THEN (the condition of the mfmtransport connected at
         the output of the mfmcalc is low_flow_low_pressure
      OR the condition of the mfmtransport connected at the
         output of the mfmcalc is low_flow
      OR (IF the condition of the mfmtransport connected at
         the output of the mfmcalc is no_sensory_data
         THEN reflection(the flow_function connected at the
            output of the mfmcalc, output, leakage)) is
            faulty))
   THEN inform the operator for the next 2 seconds that
      "storage [the name of the mfmcalc] has high flow and
      high pressure -leak "
   AND invoke transport rules
   AND conclude that the state of the mfmcalc is
      fault_leakage
   AND start set_manager (the state of the mfmcalc,"[the name
      of the mfmcalc]","")


IF the condition of any mfmcalc is
         low_flow_high_pressure
```

```
AND (IF there exists a mfmtransport connected at the output
        of the mfmcalc
    THEN (the condition of the mfmtransport connected at
        the output of the mfmcalc is low_pressure
    OR the condition of the mfmtransport connected at the
        output of the mfmcalc is low_flow_low_pressure
    OR (IF the condition of the mfmtransport connected at
        the output of the mfmcalc is no_sensory_data
        THEN reflection(the flow_function connected at the
            output of the mfmcalc, output, blockage)) is
            faulty))
THEN inform the operator for the next 2 seconds that"
    storage [the name of the mfmcalc] has high flow and
    high pressure -blocked "
AND invoke transport rules
AND conclude that the state of the mfmcalc is
    fault_blockage
AND start set_manager (the state of the mfmcalc,"[the name
    of the mfmcalc]","")
```

G.5 Sink Rules

Rules used to determine when a mfm sink function has a fault and the type of fault.

FAULT_SINK

```
IF the condition of any mfmsink is low_flow
AND ((the condition of the mfmtransport connected to the
          mfmsink is high_flow
      OR the condition of the mfmtransport connected to the
          mfmsink is high_flow_high_pressure
      OR the condition of the mfmtransport connected to the
          mfmsink is high_flow_low_pressure
      OR (IF the condition of the mfmtransport connected to
          the mfmsink is no_sensory_data
          THEN reflection(the flow_function connected to the
               mfmcalc, input, leakage)) is faulty))
  THEN inform the operator for the next 2 seconds that "sink
     [the name of the name of the mfmsink] has low flow -
     leak"
  AND invoke transport rules
  AND conclude that the state of the mfmsink is fault_leak
  AND start set_manager (the state of the mfmsink," ",[the
     name of the mfmsink]")


IF the condition of any mfmsink is low_flow
AND ((the condition of the mfmtransport connected to the
          mfmsink is low_flow_high_pressure
      OR the condition of the mfmtransport connected to the
          mfmsink is high_pressure
      OR (IF the condition of the mfmtransport connected to
          the mfmsink is no_sensory_data
          THEN reflection(the flow_function connected to the
               mfmcalc, input, blockage)) is faulty))
  THEN inform the operator for the next 2 seconds that "sink
```

```
    [the name of the mfmsink] has low flow - blocked"
  AND invoke transport rules
  AND conclude that the state of the mfmsink is
      fault_blockage
  AND start set_manager (the state of the mfmsink,"","[the
      name of the mfmsink]")


IF the condition of any mfmsink is high_pressure
  THEN inform the operator for the next 2 seconds that "sink
      [the name of the mfmsink] has high pressure - blocked"
  AND invoke transport rules
  AND conclude that the state of the mfmsink is
      fault_blockage
  AND start set_manager (the state of the mfmsink,"[the name
      of the mfmsink]","[the name of the mfmsink]")


IF the condition of any mfmsink is low_pressure
AND ((the condition of the mfmtransport connected to the
          mfmsink is high_flow_low_pressure
      OR the condition of the mfmtransport connected to the
          mfmsink is high_flow_high_pressure
      OR the condition of the mfmtransport connected to the
          mfmsink is high_flow
      OR (IF the condition of the mfmtransport connected to
          the mfmsink is no_sensory_data
          THEN reflection(the flow_function connected to the
              mfmcalc, input, leakage)) is faulty))
  THEN inform the operator for the next 2 seconds that "sink
      [the name of the mfmsink] has low pressure - leak "
  AND invoke transport rules
  AND conclude that the state of the mfmsink is fault_leak
  AND start set_manager (the state of the mfmsink,"","[the
      name of the mfmsink]")
```

```
IF the condition of any mfmsink is low_pressure
AND ((the condition of the mfmtransport connected to the
        mfmsink is low_flow_high_pressure
    OR the condition of the mfmtransport connected to the
        mfmsink is high_pressure
    OR (IF the condition of the mfmtransport connected to
        the mfmsink is no_sensory_data
        THEN reflection(the flow_function connected to the
            mfmcalc, input, blockage)) is faulty))
THEN inform the operator for the next 2 seconds that "sink
    [the name of the mfmsink] has low pressure - blocked"
AND invoke transport rules
AND conclude that the state of the mfmsink is
    fault_blocked
AND start set_manager (the state of the mfmsink,"","[the
    name of the mfmsink]")

IF the condition of any mfmsink is low_flow_low_pressure
AND ((the condition of the mfmtransport connected to the
        mfmsink is high_flow_high_pressure
    OR the condition of the mfmtransport connected to the
        mfmsink is high_flow
    OR the condition of the mfmtransport connected to the
        mfmsink is high_flow_low_pressure
    OR (IF the condition of the mfmtransport connected to
        the mfmsink is no_sensory_data
        THEN reflection(the flow_function connected to the
            mfmcalc, input, leakage)) is faulty))
THEN inform the operator for the next 2 seconds that "sink
    [the name of the mfmsink] has low flow and low pressure
    - leak"
AND invoke transport rules
AND conclude that the sate of the mfmsink is fault_leak
AND start set_manager (the state of the mfmsink,"","[the
    name of the mfmsink]")
```

```
IF the condition of any mfmsink is low_flow_low_pressure
AND ((the condition of the mfmtransport connected to the
        mfmsink is low_flow_high_pressure
    OR the condition of the mfmtransport connected to the
        mfmsink is high_pressure
    OR (IF the condition of the mfmtransport connected to
        the mfmsink is no_sensory_data
        THEN reflection(the flow_function connected to the
            mfmcalc, input, blockage)) is faulty))
  THEN inform the operator for the next 2 seconds that "sink
    [the name of the mfmsink] has low flow and low pressure
    - blocked"
  AND invoke transport rules
  AND conclude that the state of the mfmsink is
    fault_blockage
  AND start set_manager (the state of the mfmsink,"","[the
    name of the mfmsink]")


IF the condition of any mfmsink is low_flow_high_pressure
  THEN inform the operator for the next 2 seconds that "sink
    [the name of the mfmsink] has low flow and high
    pressure - blocked"
  AND conclude that the state of the mfmsink is
    fault_blockage
  AND start set_manager (the state of the mfmsink,"[the name
    of the mfmsink]","[the name of the mfmsink]")
```

Appendix H

Process Management Rules.

This appendix contains the rules used for process managements.

MANAGEMENT_RULES

IF the condition of any flow_function is fault_flow_sensor
AND state of the flow_function is not fault_flow)sensor
THEN conclude that the state of the flow function is fault_flow_sensor
AND inform the operator for the next 5 seconds that "[the name of the flow_function]has a faulty flow sensor. Check the sensor and once the sensor is operational again, then reset the state of [the name of the flow_function]."

IF the condition of any flow_function is faulty_level_sensor
AND the state of the flow_function is not faulty_level_sensor
THEN conclude that the state of the flow_function is faulty_level_sensor
AND inform the operator for the next 5 seconds that "[the name of the flow_function] has a faulty level sensor. Check the sensor and once the sensor is operational

again, then reset the state of [the name of the flow_function]."

IF the condition of any flow_function is faulty_pressure_sensor
AND the state of the flow_function is not faulty_pressure_sensor
THEN conclude that the state of the flow_function is faulty_pressure_sensor
AND inform the operator for the next 5 seconds that" [the name of the flow_function] has a faulty pressure sensor. Check the sensor and once the sensor is operational again, THEN reset the state of [the name of the flow_function]."

IF the state of any mfmmanager is ok
AND the name of the mfmgoal connected to the mfmmanager is not 1-goal
THEN conclude that the system of the mfmgoal connected to the mfmmanager = "OK"

IF the system of any mfmgoal/="OK"
AND the name of the mfmgoal is not 1-goal
THEN conclude that the system of 1-goal = the system of the mfmgoal

IF the state of any mfmmanager is not ok
THEN conclude that the system of the mfmgoal connected to the mfmmanager = "the goal of the system is not been meet since, there is a [the state of the mfmmanager] between [the fault1 of the mfmmanager] and [the fault2 of the mfmmanager]"

Appendix I

Reset Rules

Rules used to automatically reset individual flow functions when a fault condition has been cleared.

KNOWLEDGE-RULES-GENERAL

```
IF the clear_state_counter of any mfmsource > clear_time
   THEN conclude that the state of the mfmsource is ok

IF the clear_state_counter of any mfmstorage > clear_time
   THEN conclude that the state of the mfmstorage is ok

IF the clear_state_counter of any mfmtransport > clear_time
   THEN conclude that the state of the mfmtransport is ok

IF the clear_state_counter of any mfmsink > clear_time
   THEN conclude that the state of the mfmsink is ok

IF the state of the mfmsource is ok
AND fault1 = "[the name of the mfmsource]"
   THEN conclude that fault1 = "none"

IF the state of any mfmstorage is ok
AND fault1 = "[the name of the mfmstorage]"
   THEN conclude that fault1 = "none"

IF the state of any mfmtransport is ok
```

```
AND fault1 = "[the name of the mfmtransport]"
   THEN conclude that fault1 = "none"

IF the state of the mfmsink is ok
AND fault1 = "[the name of the mfmsink]"
   THEN conclude that fault1 = "none"

IF the state of the mfmsource is ok
AND fault2 = "[the name of the mfmsource]"
THEN conclude that fault2 = "none"

IF the state of the mfmstorage is ok
AND fault2 = "[the name of the mfmstorage]"
   THEN conclude that fault2 = "none"

IF the state of the mfmtransport is ok
AND fault2 = "[the name of the mfmtransport ]"
   THEN conclude that fault2 = "none"

IF the state of the mfmsink is ok
AND fault2 = "[the name of the mfmsink]"
   THEN conclude that fault2 = "none"
```

Appendix J

Functions Used

Functions used in the knowledge base.

J.1 Function: CON

This function determines the condition of the individual flow functions. Two functions are used one for flow and the other for level.

FUNCTION

```
con (who) =

(IF the low_flow of who = the high_flow of who
   AND the low_pressure of who = the high_pressure of who
THEN the symbol no_sensory_data
ELSE
   (IF the flow of who > (max(0,50.5 * extrapolat(who) +
                  extrapolat (who))
      OR the flow of who < (extrapolat (who) - max
            (0.5,0.5*extrapolat(who))
   THEN the symbol faulty_flow_sensor
   ELSE (IF the pressure of who > (max(0.5,0.5*
```

```
    extrapolat_pres (who)) +
    extrapolat_pres(who))
    OR the pressure of who < (extrapolat_pres (who) -
        max (0.5,0.5*extrapolat_pres who)))
THEN the symbol faulty_pressure_sensor
ELSE (IF the flow of who < the low_flow of who
    THEN (IF the pressure of who < the low_pressure of
        who
        THEN the symbol low_flow_low_pressure
        ELSE (IF the pressure of who > the
            high_pressure of who
            THEN the symbol low_flow_high_pressure
            ELSE the symbol low_flow))
    ELSE (IF the flow of who > the high_flow of who
        THEN (IF the pressure of who < the low_pressure
            of who
            THEN the symbol high_flow_low_pressure
            ELSE (IF the pressure of who > the
                high_pressure of who
                THEN the symbol low_flow_high_pressure
                ELSE the symbol low_flow))
        ELSE IF the flow of who > the high_flow of who
            THEN (IF the pressure of who < the
                low_pressure of who
                THEN the symbol high_flow_low_pressure
                ELSE (IF the pressure of who > the
                    high_pressure of who
                    THEN the symbol
                        high_flow_high_pressure
                    ELSE the symbol high_flow))
            ELSE IF the pressure of who < the
                low_pressure of the
                THEN the symbol low_pressure
                ELSE (IF the pressure of who > the
                    high_pressure of who
                    THEN the symbol high_pressure
                    ELSE the symbol normal))))))
```

```
cond_level (who) =

(IF the low_level of who = the high_level of who
  AND the low_pressure of who = the high_pressure of who
THEN the symbol no_sensory_data
ELSE
  (IF the level of who > (max(0.5,0.5*extrapolat_level
    (who)) + extrapolat_level (who)
    OR the level of who < (extrapolat_level (who) -
      max(0.5,0.5*extrapolat_level(who)))
  THEN the symbol faulty_level_sensor
  ELSE (IF the pressure of who > (max(0.5,0.5*
      extrapolat_pres(who)) + extrapolat_pres(who))
    OR the pressure of who < (extrapolat_pres (who) -
      max(0.5,0.5*extrapolat_pres(who)))
    THEN the symbol faulty_pressure_sensor
    ELSE (IF the level of who ≤ the low_level of who
      THEN (IF the pressure of who ≤ the low_pressure
        of who
        THEN the symbol low_level_low_pressure
        ELSE (IF the pressure of who > the
          high_pressure of who
          THEN the symbol low_level_high_pressure
          ELSE the symbol low_level))
      ELSE (IF the level of who > the high_level of
        who
        THEN (IF the pressure of who < the
          low_pressure of who
          THEN the symbol high_level_low_pressure
          ELSE (IF the pressure of who > the
            high_pressure of who
            THEN the symbol
              high_level_high_pressure
            ELSE the symbol high_level))
        ELSE IF the pressure of who < the
        low_pressure of who
```

```
    THEN the symbol low_pressure
ELSE (IF the pressure of who > the
    high_pressure of who
    THEN the symbol high_pressure
    ELSE the symbol normal))))).
```

J.2 Function: EXTRAPOLAT

This function predicts the next measurement by extrapolation. Three functions are used for flow, pressure and the other for level.

```
extrapolat (who) =
   the value of the flow of who as of 5 ago
   - 5 * the value of the flow of who as of 4 ago
   + 10* the value of the flow of who as of 3 ago
   - 10* the value of the flow of who as of 2 ago
   + 5* the value of the flow of who as 1 ago


extrapolat_level (who) =
   the value of the level of who as of 5 ag
   - 5* the value of the level of who as of 4 ago
   + 10* the value of the level of who as of 3 ago
   - 10* the value of the level of who as of 2 ago
   + 5* the value of the level of who as of 1 ago


extrapolat_pres (who) =
   the value of the pressure of who as of 5 ago
   - 5*the value of the pressure of who as of 4 ago
   + 10* the value of the pressure of who as of 3 ago
   - 10* the value of the pressure of who as of 2 ago
   + 5* the value of the pressure of who as of 1 ago
```

J.3 Function: REFLECTION

This function reflects the condition of neighbouring flow functions.

```
reflection(who,direction,type) =

(IF the who is a connection
THEN
   (IF type = blockage
   THEN
      (IF direction = output
      THEN
         (IF the out_condition of who is
             low_flow_low_pressure
         OR the out_condition of who is low_flow
         OR the out_condition of who is low_pressure
         OR the out_condition of who is normal
         OR the out_condition of who is
             low_level_low_pressure
         OR the out_condition of who is low_level
         THEN
             the symbol faulty
         ELSE
             the symbol normal)
      ELSE
         (IF the in_condition of who is
             low_flow_high_pressure
         OR the in_condition of who is high_pressure
         OR the in_condition of who is
             high_level_high_pressure
         OR the in_condition of who is high_level
         THEN
             the symbol faulty
         ELSE
             the symbol normal))
   ELSE
```

```
(IF type = leakage
THEN
    (IF direction = output
    THEN
        (IF the out_condition of who is
            low_flow_low_pressure
        OR the out_condition of who is low_flow
        OR the out_condition of who is normal
        OR the out_condition of who is
            low_level_low_pressure
        OR the out_condition of who is low_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal)
    ELSE
        (IF the in_condition of who is
            high_flow_low_pressure
        OR the in_condition of who is high_flow
        OR the in_condition of who is
            low_level_low_pressure
        OR the in_condition of who is low_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal)))
ELSE
    (IF the condition of who is no_sensory_data
    THEN
        (IF who is a mfmsink OR who is a mfmsource
        THEN
            the symbol faulty
        ELSE
            reflection(the flow function connected at the
                direction of who,direction,type))
    ELSE
    (IF type = blockage
```

```
THEN
    (IF direction = output
    THEN
        (IF the condition of who is
            low_flow_low_pressure
        OR the condition of who is low_flow
        OR the condition of who is low_pressure
        OR the condition of who is normal
        OR the condition of who is
            low_level_low_pressure
        OR the condition of who is low_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal)
    ELSE
        (IF the condition of who is
            low_flow_high_pressure
        OR the condition of who is high_pressure
        OR the condition of who is
            high_level_high_pressure
        OR the condition of who is high_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal))
ELSE
    (IF type = leakage
    THEN
        (IF direction = output
        THEN
            (IF the condition of who is
                low_flow_low_pressure
            OR the condition of who is low_flow
            OR the condition of who is normal
            OR the condition of who is
            low_level_low_pressure
```

```
                OR the condition of who is low_level
                THEN
                    the symbol faulty
                ELSE
                    the symbol normal)
            ELSE
                (IF the condition of who is
                high_flow_low_pressure
                OR the condition of who is high_flow
                OR the condition of who is
                low_level_low_pressure
                OR the condition of who is low_level
                THEN
                    the symbol faulty
                ELSE
                    the symbol normal))))
```



```
reflection_secondary(condition,who,direction) =

(IF the condition of who is no_sensory_data
AND who is not a mfmconnection
AND who is not a mfmsink
AND who is not a mfmsource
THEN
   reflection_secondary(condition,the flow function connected
      at the direction of who,direction)
ELSE

(CASE(condition) of

low_flow:
   BEGIN
      (IF who is a connection
      THEN
```

```
        (IF direction = output
        THEN
            (IF the out_condition of who is
                low_flow_high_pressure
            OR the out_condition of who is high_pressure
            OR the out_condition of who is
                high_level_high_pressure
            OR the out_condition of who is high_level
            OR the out_condition of who is high_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the in_condition of who is
                low_flow_low_pressure
            OR the in_condition of who is low_pressure
            OR the in_condition of who is
                low_level_low_pressure
            OR the in_condition of who is low_level
            THEN
                the symbol faulty
            ELSE
                the symbol normal))
ELSE
    (IF direction = output
    THEN
        (IF the condition of who is low_flow_high_pressure
        OR the condition of who is high_pressure
        OR the condition of who is high_level_high_pressure
        OR the condition of who is high_level
        OR the condition of who is high_pressure
        THEN
            the symbol faulty
        ELSE
            the symbol normal)
    ELSE
```

```
        (IF the condition of who is low_flow_low_pressure
        OR the condition of who is low_pressure
        OR the condition of who is low_level_low_pressure
        OR the condition of who is low_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal)))
  END;

low_flow_low_pressure:
  BEGIN
    (IF who is a connection
     THEN
        (IF the in_condition of who is
            low_flow_low_pressure
        OR the in_condition of who is low_pressure
        OR the in_condition of who is
            low_level_low_pressure
        OR the in_condition of who is low_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal)
     ELSE
        (IF the condition of who is low_flow_low_pressure
        OR the condition of who is low_pressure
        OR the condition of who is low_level_low_pressure
        OR the condition of who is low_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal))
  END;

low_flow_high_pressure:
  BEGIN
```

```
        (IF who is a connection
        THEN
            (IF the out_condition of who is
                low_flow_high_pressure
            OR the out_condition of who is high_pressure
            OR the out_condition of who is
                high_level_high_pressure
            OR the out_condition of who is high_level
            OR the out_condition of who is high_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the condition of who is low_flow_high_pressure
            OR the condition of who is high_pressure
            OR the condition of who is high_level_high_pressure
            OR the condition of who is high_level
            OR the condition of who is high_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal))
    END;

low_pressure:
    BEGIN
        (IF who is a connection
        THEN
            (IF direction = output
            THEN
                (IF the out_condition of who is
                    high_flow_low_pressure
                OR the out_condition of who is high_flow
                THEN
                    the symbol faulty
                ELSE
```

```
                the symbol normal)
            ELSE
                (IF the in_condition of who is
                    low_flow_low_pressure
                OR the in_condition of who is low_pressure
                OR the in_condition of who is
                    low_level_low_pressure
                OR the in_condition of who is low_level
                OR the in_condition of who is low_flow
                THEN
                    the symbol faulty
                ELSE
                    the symbol normal))
    ELSE
        (IF direction = output
        THEN
            (IF the condition of who is high_flow_low_pressure
            OR the condition of who is high_flow
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the condition of who is low_flow_low_pressure
            OR the condition of who is low_pressure
            OR the condition of who is low_level_low_pressure
            OR the condition of who is low_level
            OR the condition of who is low_flow
            THEN
                the symbol faulty
            ELSE
                the symbol normal)))
    END;

low_level:
    BEGIN
        (IF who is a connection
```

```
    THEN
        (IF direction = output
        THEN
            (IF the out_condition of who is
                high_flow_high_pressure
            OR the out_condition of who is high_flow
            OR the out_condition of who is
                high_flow_low_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the in_condition of who is
                low_flow_low_pressure
            OR the in_condition of who is low_pressure
            OR the in_condition of who is low_flow
            THEN
                the symbol faulty
            ELSE
                the symbol normal))
ELSE
    (IF direction = output
    THEN
        (IF the condition of who is high_flow_high_pressure
        OR the condition of who is high_flow
        OR the condition of who is high_flow_low_pressure
        THEN
            the symbol faulty
        ELSE
            the symbol normal)
    ELSE
        (IF the condition of who is low_flow_low_pressure
        OR the condition of who is low_pressure
        OR the condition of who is low_flow
        THEN
            the symbol faulty
```

```
            ELSE
                the symbol normal)))
    END;

low_level_low_pressure:
    BEGIN
        (IF who is a connection
        THEN
            (IF direction = output
            THEN
                (IF the out_condition of who is
                    high_flow_high_pressure
                OR the out_condition of who is high_flow
                OR the out_condition of who is
                    high_flow_low_pressure
                THEN
                    the symbol faulty
                ELSE
                    the symbol normal)
            ELSE
                (IF the in_condition of who is
                    low_flow_low_pressure
                OR the in_condition of who is low_pressure
                OR the in_condition of who is low_flow
                THEN
                    the symbol faulty
                ELSE
                    the symbol normal))
    ELSE
        (IF direction = output
        THEN
            (IF the condition of who is high_flow_high_pressure
            OR the condition of who is high_flow
            OR the condition of who is high_flow_low_pressure
            THEN
                the symbol faulty
            ELSE
```

```
            the symbol normal)
    ELSE
        (IF the condition of who is low_flow_low_pressure
        OR the condition of who is low_pressure
        OR the condition of who is low_flow
        THEN
            the symbol faulty
        ELSE
            the symbol normal)))
END;

high_flow:
  BEGIN
    (IF who is a connection
    THEN
        (IF direction = output
        THEN
            (IF the out_condition of who is
                high_flow_low_pressure
            OR the out_condition of who is high_flow
            OR the out_condition of who is
                low_level_low_pressure
            OR the out_condition of who is low_level
            OR the out_condition of who is low_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the in_condition of who is
                high_flow_high_pressure
            OR the in_condition of who is high_pressure
            OR the in_condition of who is
                high_level_high_pressure
            OR the in_condition of who is high_level
            THEN
                the symbol faulty
```

```
            ELSE
                the symbol normal))
    ELSE
        (IF direction = output
        THEN
            (IF the condition of who is high_flow_low_pressure
            OR the condition of who is high_flow
            OR the condition of who is low_level_low_pressure
            OR the condition of who is low_level
            OR the condition of who is low_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the condition of who is high_flow_high_pressure
            OR the condition of who is high_pressure
            OR the condition of who is high_level_high_pressure
            OR the condition of who is high_level
            THEN
                the symbol faulty
            ELSE
                the symbol normal)))
    END;

high_flow_low_pressure:
    BEGIN
        (IF who is a connection
        THEN
            (IF the out_condition of who is
                high_flow_low_pressure
            OR the out_condition of who is high_flow
            OR the out_condition of who is
                low_level_low_pressure
            OR the out_condition of who is low_level
            OR the out_condition of who is low_pressure
            THEN
```

```
            the symbol faulty
        ELSE
            the symbol normal)
    ELSE
        (IF direction = output
        THEN
            (IF the condition of who is high_flow_low_pressure
            OR the condition of who is high_flow
            OR the condition of who is low_level_low_pressure
            OR the condition of who is low_level
            OR the condition of who is low_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)))
    END;

high_flow_high_pressure:
    BEGIN
        (IF who is a connection
        THEN
            (IF the in_condition of who is
                high_flow_high_pressure
            OR the in_condition of who is high_pressure
            OR the in_condition of who is
                high_level_high_pressure
            OR the in_condition of who is high_level
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
    ELSE
        (IF direction = output
        THEN
            (IF the condition of who is high_flow_high_pressure
            OR the condition of who is high_pressure
            OR the condition of who is high_level_high_pressure
```



```
        OR the condition of who is high_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal)))
  END;

high_pressure:
  BEGIN
    (IF who is a connection
    THEN
        (IF direction = output
        THEN
            (IF the out_condition of who is
                low_flow_high_pressure
            OR the out_condition of who is
                high_level_high_pressure
            OR the out_condition of who is high_pressure
            OR the out_condition of who is high_level
            OR the out_condition of who is low_flow
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the in_condition of who is
                high_level_high_pressure
            OR the in_condition of who is high_level
            THEN
                the symbol faulty
            ELSE
                the symbol normal))
  ELSE
    (IF direction = output
    THEN
        (IF the condition of who is low_flow_high_pressure
        OR the condition of who is high_level_high_pressure
```

```
        OR the condition of who is high_pressure
        OR the condition of who is high_level
        OR the condition of who is low_flow
        THEN
            the symbol faulty
        ELSE
            the symbol normal)
    ELSE
        (IF the condition of who is
            high_level_high_pressure
        OR the condition of who is high_level
        THEN
            the symbol faulty
        ELSE
            the symbol normal)
END;

high_level:
  BEGIN
    (IF who is a connection
    THEN
        (IF direction = output
        THEN
            (IF the out_condition of who is
                high_flow_high_pressure
            OR the out_condition of who is high_flow
            OR the out_condition of who is
                high_level_high_pressure
            OR the out_condition of who is
                high_level_high_pressure
            OR the out_condition of who is high_high
            OR the out_condition of who is high_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
```

```
            (IF the in_condition of who is
                high_flow_high_pressure
            OR the in_condition of who is high_pressure
            OR the in_condition of who is high_flow
            THEN
                the symbol faulty
            ELSE
                the symbol normal))
    ELSE
        (IF direction = output
        THEN
            (IF the condition of who is high_flow_high_pressure
            OR the condition of who is high_flow
            OR the condition of who is high_level_high_pressure
            OR the condition of who is high_level_high_pressure
            OR the condition of who is high_high
            OR the condition of who is high_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the condition of who is high_flow_high_pressure
            OR the condition of who is high_pressure
            OR the condition of who is high_flow
            THEN
                the symbol faulty
            ELSE
                the symbol normal)))
    END;

high_level_high_pressure:
    BEGIN
        (IF who is a connection
        THEN
            (IF direction = ou put
            THEN
```

```
            (IF the out_condition of who is
                high_flow_high_pressure
            OR the out_condition of who is high_flow
            OR the out_condition of who is
                high_level_high_pressure
            OR the out_condition of who is
                high_level_high_pressure
            OR the out_condition of who is high_high
            OR the out_condition of who is high_pressure
            THEN
                the symbol faulty
            ELSE
                the symbol normal)
        ELSE
            (IF the in_condition of who is
                high_flow_high_pressure
            OR the in_condition of who is high_pressure
            OR the in_condition of who is high_flow
            THEN
                the symbol faulty
            ELSE
                the symbol normal))
ELSE
    (IF direction = output
    THEN
        (IF the condition of who is high_flow_high_pressure
        OR the condition of who is high_flow
        OR the condition of who is high_level_high_pressure
        OR the condition of who is high_level_high_pressure
        OR the condition of who is high_high
        OR the condition of who is high_pressure
        THEN
            the symbol faulty
        ELSE
            the symbol normal)
    ELSE
        (IF the condition of who is high_flow_high_pressure
```

```
        OR the condition of who is high_pressure
        OR the condition of who is high_flow
        THEN
            the symbol faulty
        ELSE
            the symbol normal))
    END;
END
```

J.4 Function: CHECK

This function determines the next value of the clear_state_counter for each flow functions.

```
check (who) =
   the value of the clear_state_counter of who as of 1 ago
   + (IF the condition of who is normal
      THEN 1
      ELSE (-the value of the clear_state_counter of who as
          of 1 ago))
```

Author: BracherStephen.
Name of thesis: Multilevel flow modelling as a generalized modelling technique.

PUBLISHER:
University of the Witwatersrand, Johannesburg
©2015

LEGALNOTICES:

Copyright Notice: All materials on the University of the Witwatersrand, Johannesburg Library website are protected by South African copyright law and may not be distributed, transmitted, displayed or otherwise published in any format, without the prior written permission of the copyright owner.

Disclaimer and Terms of Use: Provided that you maintain all copyright and other notices contained therein, you may download material (one machine readable copy and one print copy per page)for your personal and/or educational non-commercial use only.

The University of the Witwatersrand, Johannesburg, is not responsible for any errors or omissions and excludes any and all liability for any errors in or omissions from the information on the Library website.