CHALMERS | UNIVERSITY OF GOTHENBURG

MASTER'S THESIS

Architecture for Fault-tolerant Control and Construction of Bayesian Network for Diagnosis

ERIK LANDSTRÖM

Department of Mathematical Statistics
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2010

Thesis for the Degree of Master of Science (30 credits)

Architecture for Fault-tolerant Control and Construction of Bayesian Network for Diagnosis

Erik P.J. Landström

CHALMERS | UNIVERSITY OF GOTHENBURG

Department of Mathematical Statistics
Chalmers University of Technology and University of Gothenburg
SE – 412 96 Gothenburg, Sweden

Gothenburg, September 2010

Abstract

This thesis evaluates an architecture for diagnostic modeling and fault-tolerant control. The first part of the thesis introduces the architecture and a mechatronic system inspired by automotive applications is constructed in Simulink. Common and critical faults of the system are identified and made available for activation. Single and double faults are implemented and the system is found to have promising fault-tolerant properties.

The second part of the thesis deals with model based diagnosis of the mechatronic system. The architecture enables the construction of a Bayesian network which is built with the use of expert knowledge as the sole source of information. The network is validated and is found to isolate the better part of logic faults as well as faults related to uncertainties. Questions are raised about the network used in large-scale applications regarding intractability of inference and the difficulties involved using expert knowledge.

Acknowledgements

First of all, I would like to thank my excellent supervisor at Scania CV AB, Mattias Nyberg, for his great support during my work. I would also like to thank my supervisor Patrik Albin at Chalmers for valuable advices and the input towards the end of the work. Finally, I would also like to give my gratitude to all persons attending the reference meetings at Scania during the thesis.

Contents

Chapter 1

Introduction

Our modern society has become strongly dependent upon the availability of complex technological services. One example is the transportation service which a modern truck offers. The trucks of today use embedded systems called *electronic control units* (ECU) for control of electrical systems or subsystems. The ECUs communicate with each other through a specifically designed bus standard called *controller-area network* (CAN). Managing the increasing complexity and number of ECUs in a vehicle has become a key challenge for motor vehicle manufacturers. A single fault in an ECU can have major effects on the systems' (ECUs and CAN) availability and performance. Also strict environmental regulations exist which claim that the engine has to be supervised and shut off in the case of a fault.

The increasing demand for availability and safety of complex technological services has led to the development of the branch of control theory which is referred to as *fault-tolerant control* (FTC). Traditional methods for FTC has been used in safety-critical systems where in case of a fault the system switches to a redundant component. This method is too expensive for the industry and a cheaper solution could be to use *reconfiguration* leading to a possible degraded but acceptable performance.

The FTC issues are integrated at the very early stages of control systems design. Typically the FTC problem is controlling, to the maximum extent possible, the operation of the systems in the presence of fault(s). What is meant by control depends on the demands of the user but for this thesis it is defined as in [6] under three major statements.

1. Continue system operation without (unbearable) loss of performance.

2. Continue system operation with reduced specifications.

3. Abandon mission while avoiding disaster.

The second and third statement above implies that a faulty component may need to be replaced. If that is the case then it is highly desirable to have a model of the system in order to perform troubleshooting.

The objective of this master thesis is to apply and evaluate a new type of architecture [8] for fault-tolerant control. The architecture enables diagnosis and reconfiguration to be fully decentralised. It is also well suited for modeling the system as a *bayesian network* which in recent years have been preferred by reserachers that study fault diagnosis in engineering areas such as airplane diagnosis [10]. The architecture is applied to a Simulink model of a simplified ECU. The evaluation is based on how fault-tolerant the system is against some common faults and also on how well the Bayesian network isolates the faulty components and performs in a more realistic large-scale model in terms of inference speed.

A restriction has been made not to consider how the ECU safely should be shut down in case of the third statement above.

This Masters thesis has been performed for Scania CV AB at the group of RESD during the period march - august of 2010.

1.1 Notation

Table 1.1 below shows the abbreviations with their meaning used in this thesis.

Abbreviation	*Meaning*
ECU	Electrical control unit
SW	Software
HW	Hardware
AFCS	Air Flow Control system
FTC	Fault-tolerant control
ADC	Analog to Digital Converter
PWM	Pulse Width Modulation
VSSE	Variant service status estimation

Table 1.1: List of abbreviations and their meaning.

Part I

Fault-tolerant control of mechatronic system

Chapter 2

Basic concepts and example design of a service provider

2.1 Introduction

The common architecture of fault-tolerant control, as described in [1] and depicted below in Figure 2.1, uses a centralised diagnoser to detect and identify a fault and a centralised re-designer to adjust the controller to the faulty situation.

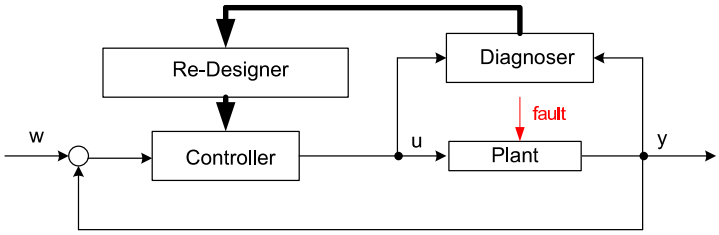

Figure 2.1: Common architecture of fault-tolerant control. The plant is subjected to a fault. The diagnoser identifies the fault and the re-Designer adjusts the controller to the faulty situation.

In this thesis a new type of architecture is used [8]. It is believed to be better suited for solving the FTC problem of large-scale mechatronic systems distributed over several ECUs by using a decentralized and modular architecture. The theory of [8] is described in this chapter with some exceptions mentioned in the text. Definitions are highlighted with italic letters.

2.2 Basic concepts of a service provider

The system regarded in this thesis is a simplified ECU connected in a communication network controlling a mechatronic system consisting of physical components. The software (SW) and hardware (HW) in the ECU are divided into modules. Each SW-module, HW-module and physical component is viewed upon as a *service provider* since its purpose is to deliver one or more services. In this thesis the service providers only provide one single service but the extension to more services is trivial.

The service provider delivers its service to a *customer*. The customer can be human or another service provider in the system. To deliver its service the service provider uses one or more *suppliers* which in turn are service providers. The customer - service provider - supplier relationship is depicted in Figure 2.2.

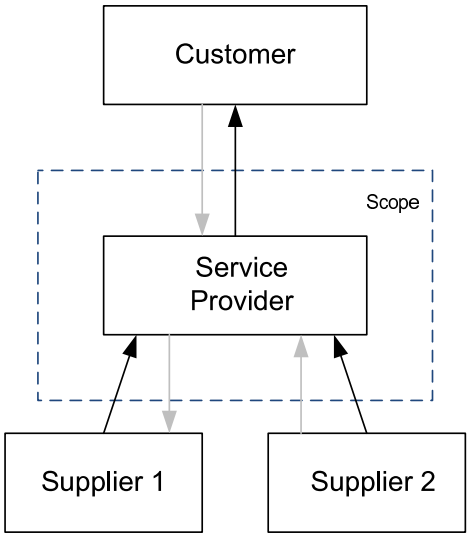

Figure 2.2: Relationship between customer, service provider and suppliers. Black arrows represent service dependencies and grey arrows represent signal flow.

2.2.1 Service status

The service provider may deliver its service with a different quality. The different qualities are described by the *service status* of a service provider. The service status is divided into three *service classes* which is denoted by *nominal* (NOM), *disturbed* (DIST) and *unavailable* (UNA). The service status of a service provider m is denoted by S_m.

The status NOM is always pertinent. UNA and DIST may or may not be needed depending on the specific relation between the service provider and its customer. If DIST is used it has a clear meaning defined in agreement with the customer of the service.

If needed the service class of DIST can further be divided into different levels DIST_1, DIST_2, DIST_3, etc.

2.2.2 Service monitoring and the scope of a service provider

The service provider has, in addition to deliver its service, the task to monitor its service status and communicate this to its customers upwards in the system hierarchy. This is called *service monitoring*. The purpose of service monitoring is to give the customer of the service the chance to adapt itself using reconfiguration to a possible better alternative.

In order to reduce the amount of dependencies in the system the service provider is only allowed to use information within its *scope*. The scope of a service provider shown in Figure 2.2 is defined to be:

1. Signals from customers and suppliers.
2. Signals from itself.
3. Service statuses from suppliers.
4. Often some knowledge of the hardware or electronics.

It is evident that the service status estimated from information within the scope could differ and be less accurate in comparison with using information from outside the scope. This is the prize paid for trying to keep the dependencies low and using a decentralised architecture. The estimate by a service provider m of its service status S_m is denoted by $\hat{S}_m$.

Since it might not be possible to obtain $\hat{S}_m$ with high precision it is sometimes enough to use a lower resolution of $\hat{S}_m$ compared to S_m. This means that the domain of $\hat{S}_m$ and S_m may differ.

2.2.3 Variants

To achieve fault-tolerance a service provider should avoid failure when its suppliers are delivering their services with degraded performance. Fault-tolerance is achieved with reconfiguration which is accomplished with the concepts of *variants*.

A service provider m may exist in several variants denoted *m:1*, *m:2* etc. Each variant deliver the same service but differ by its accuracy and naturally there exists a preference relation among them that is defined by the designer. Each variant uses different and possibly overlapping subsets of the suppliers. Thus different variants will have different sensitivities to different degraded suppliers. The strategy for obtaining fault-tolerance is then to choose the best variant to be run.

If there is no reconfiguration available, it is the special case of a service provider with one single possible variant.

2.2.4 General principles of service status estimation

The service status of a service provider equals the service status of the variant that is selected to be run. The service status of a variant m:i is denoted by $S_{m:i}$. The estimated service status of a variant m:i is denoted by $\hat{S}_{m:i}$.

To estimate its status each variant m:i uses the service statuses from the suppliers which it is dependent on and possibly other signals within the scope of the service provider. The essential signals can be extracted and used in a *diagnostic test*. The diagnostic tests in this thesis uses standard techniques such as in-range checks, size of control-error and analytic residuals.

The general principles of service status estimation will now be given. Starting with the case where no diagnostic tests are involved.

No diagnostic test

All the variants of a service provider has a *default service status* when all of its suppliers are delivering their services with the status NOM. The most prefered variant has a default status equal to NOM. When its suppliers' service statuses becomes equal to UNA the most preferred variant cannot deliver its service. It is then useful to design variants with less requirements on the suppliers' service statuses with a default service status of DIST. Based on the idea of a default service and the natural principle that a variant cannot deliver its service when one of its suppliers becomes UNA the following general principles are obtained:

1. If all suppliers communicates NOM, the service status of the variant is equal to its default service status.

2. If any supplier communicate UNA, the service status of the variant becomes UNA.

If any supplier communicates DIST then the service status of the variant can be NOM, DIST or UNA i.e. no general principles exist for this particular case. The choice is based on a systematic analysis and it is up to the system designer to make the decision.

Diagnostic tests involved

The service status communicated by suppliers is based on the information available in their respective scopes. Since the scope is a limited source of information the communicated service statuses are not necessarily the true ones. Therefore when estimating the service statuses of the variants the use of diagnostic tests are a good potential. The diagnostic tests are contained within the service provider and utilize signals within the scope. The use of a diagnostic test means that even if the communicated service statuses of the suppliers are NOM the test result may imply that the service status of the variant is DIST or UNA. Again the choice is based on a systematic analysis of the consequences and it is up to the system designer to make the decision.

The tests in this thesis are executed if the suppliers are not communicating UNA. If the test is not executed the test result is failed[1].

2.2.5 Selection of variant

In a service provider with the capability of reconfiguration there must exist a mechanism for selecting the variant to be used. The selection is based on the mapping from service statuses of variants to the selection of variant which is realized with the preference relation among the variants defined by the designer of the system.

2.2.6 Physical components

The view of a service provider can also be applied to physical components. Everything said above is valid except that a physical component do not have the ability to estimate and communicate its service.

2.3 Example design of service monitoring

This section explains the example design of service monitoring used in this thesis. The process of service monitoring is divided into two steps - variant service status estimation and selection of variant.

2.3.1 Variant service status estimation (VSSE)

In the first step each variant estimates its service status using a mechanism called *VSSE*. The inputs to the VSSE are the estimated service statuses of suppliers and relevant diagnostic test results. This solution is different from the one used in [8] where the service statuses of suppliers are reassessed using a minimization process with the diagnostic test results and the estimated service status of the suppliers, and then used as inputs to the VSSE. It is believed by the author of this thesis that the different solutions essentially give the same results and the solution used in this thesis just reflects the authors preference of using less components. The output of the VSSE for both solutions is the service status of the variant: NOM, DIST or UNA.

The VSSE can be seen as a mapping from the possible combinations of supplier service statuses and diagnostic test results to the service status of the variant. It is required that this is done in an efficient way. The design used in this thesis uses if-else-cases which may become unattractive if the number of combinations grow. If the number of combinations are large a table could be used or possibly if-else-cases combined with a table for treatment of special cases.

[1] Usually the standard is to have a value which represents the case of a non-executed test in order to avoid setting off other alarms higher in the system hierarchy. This is not used in this thesis since it would imply more signal values to handle.

2.3.2 Selector

To select the variant to be run the service provider uses a mechanism called *selector*. The inputs to the selector are the estimated variant service statuses and the output is the selected variant and estimated service status of the service provider. The selector in this thesis uses if-else-cases based upon the predefined preference relation among the variants to select the variant with the best service status.

Chapter 3

Air Flow Control system - In a service view

In order to test the architecture specified in Chapter 2 a simplified ECU inspired by automotive applications is created in Simulink. The function of the ECU is described in the first section of this chapter. In order to evaluate the system in terms of its fault-tolerant property a set of faults are made available for implementation. The second section of this chapter depicts and motivates the set of faults considered. The third section describes how the service provider architecture is applied to the system design.

3.1 Air Flow Control system

The Air Flow Control system (AFCS) is a simplified ECU which controls the stoichiometry combustion in an engine by regulating the air flow under the influence of an external fuel reference command. The target is the ideal stoichiometric ratio between air mass and fuel mass which provides a good compromise between power, fuel economy and emissions. The target ratio is set to 14.3. A schematic of the AFCS is shown in Figure 3.1.

The physical part of the system consists of an air tank with high pressure air connected by a throttle valve to ambient air. A fuel injector is connected at the other side. The throttle valve is mechanically linked to a manoeuvre valve which is pneumatically controlled through a tube by a proportional valve. There are two sensors. The first one measures the pressure in the air tank. The other one measures the position of the throttle valve. The proportional valve, the fuel injector and the sensors are connected to the ECU. The ECU is composed of HW and SW. The HW consists of ADCs and PWMs while the SW consists of estimators and controllers.

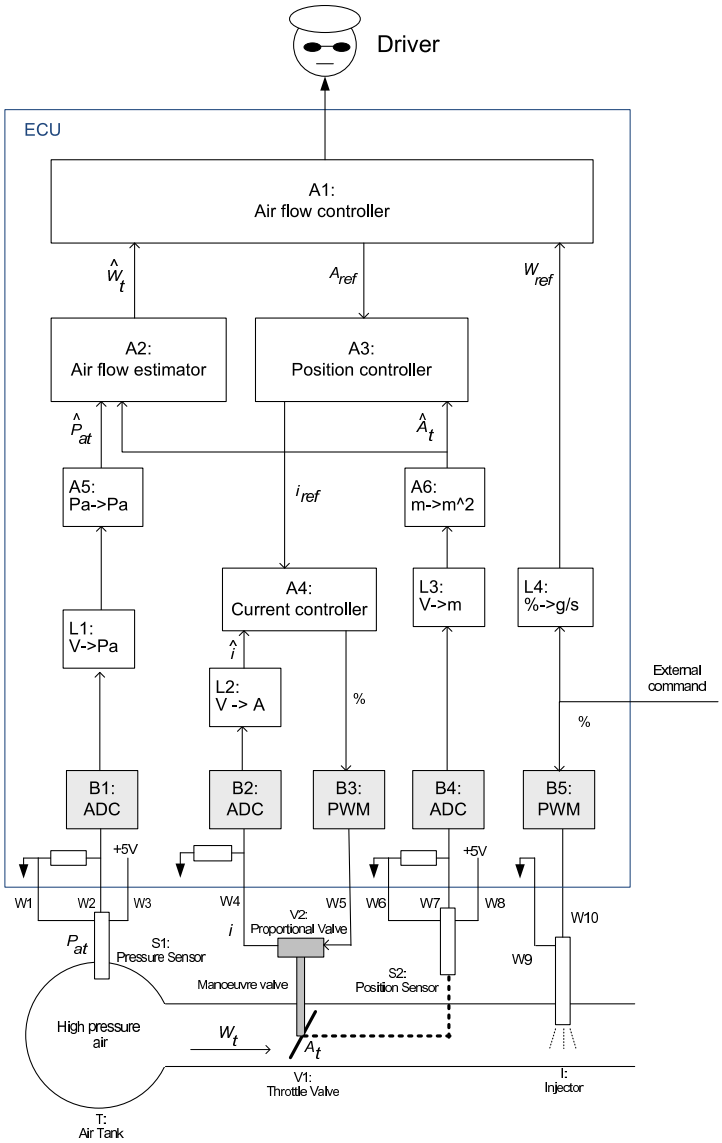

Figure 3.1: Schematic of Air Flow Control system.

3.1.1 Equations and physics

Manoeuvre valve

The manoeuvre valve is modeled as an ideal spring-mass-damper system as shown in Figure 3.2. The system has a mass m_p, a spring constant k_s and viscous damper of damping coefficient c. By Newtons second law

$$m_p\ddot{x} = -k_s x - c\dot{x} + F. \tag{3.1}$$

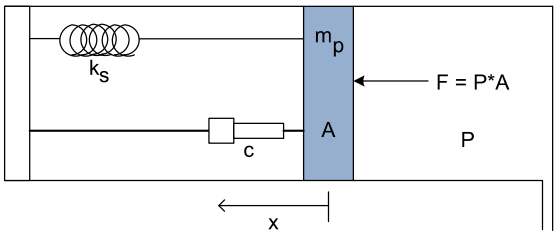

Figure 3.2: Model of manoeuvre valve.

In (3.1) x is the displacement and F is the force acting on the mass. Here F is composed of the pressure, tuned by the proportional valve, times the area of the adjustable wall.

The value of the damping ratio is

$$\zeta = \frac{c}{2\sqrt{m_p k_s}}. \tag{3.2}$$

In (3.2) c, m_p and k_s are chosen in order for the system to become underdamped.

Air flow estimation

The throttle valve acts like a restriction to the air mass. In order to keep the model simple it is assumed that the air is incompressible and isenthalpic. The model is inspired by the model of an air-filter and the air flow, W_t, through the throttle is given by

$$W_t = \begin{cases} CA_t\Delta P\sqrt{P_{at}}, & \Delta p \geq 0 \\ 0, & \Delta p < 0. \end{cases}$$

In the equation above P_{at} is the pressure in the air tank, $\Delta P = (P_{at} - P_{amb})$ where P_{amb} is the ambient pressure, C is a constant and A_t is the open throttle area.

High pressure air tank

The initial pressure of the tank is set to 20 Bar. The pressure, P_{at}, will drop because of the air flow according to the ideal gas law

$$P_{at} = \frac{RT}{V}\int -W_t dt. \tag{3.3}$$

In (3.3) R, T and V are assumed to be constants.

External command

The external command is a random pulse signal which changes every twelth second.

PID controllers

The SW-modules A1, A3 and A4 are PID controllers with feed forward control.

3.1.2 Component classes

The components of the AFCS are classified into five classes according to Figure 3.3. The classes have connections regarding interchangeability. The components in the Drivers class have connections with the components in Generic sensors and actuators (GSA). For example, L1 knows that the voltage it is converting comes from the pressure sensor but not what the sensor is connected to. The components in the Application class have connections with the components in the Physical components related to application class (PCRA). For example, A3 has knowledge about the throttle valve data. The Bios class have no connection with the other classes. Using this principle, GSA together with Drivers are interchangeable and can be used in another application where the use of an pressure sensor is needed. Application together with PCRA are interchangeable and can be used in another system. Bios, being a part of the ECU hardware, is also interchangeable.

3.2 Faults

According to [1] a reasonable application of fault-tolerant control starts with the selection of the most critical faults and continues with the investigation of fault tolerance against these faults. Further a fault is defined to be a severe change whose effect on the system cannot be suppressed by a fixed controller.

Figure 3.4 shows the selection of critical and common faults that are considered for the AFCS. Table 3.1 shows a description of the faults and the notation that will be used in the rest of the thesis. It is made possible to activate the fault(s) at a specific running time.

A limitation is made to not consider the faults F_5, F_{15}, F_{21}, F_{22} and F_{23}. The limitation is made for F_5, F_{21}, F_{22} and F_{23} since there does not exist any way of being fault-tolerant against these faults in the current system architecture. The limitation is made for F_{15} since this fault is not a fault technically speaking according to the definition made earlier in this section. A deviation from the specification of the proportional valve is attenauted by the integral part of A4 after a short time of degraded performance. Transient behaviour and intermittent faults are realistic and interesting occurences but are not considered in this thesis.

3.3 Service view

Following the theory in chapter 2 all components in the AFCS, SW, HW and physical, are considered as service providers.

Fault	*Description*	*Notation*
F_1	ADC has a malfunction	B1 = MALF
F_2	ADC has a malfunction	B2 = MALF
F_3	PWM has a malfunction	B3 = MALF
F_4	ADC has a malfunction	B4 = MALF
F_5	PWM has a malfunction	B5 = MALF
F_6	Open circuit from pressure sensor to data port	W2 = OC
F_7	Open circuit from power supply to pressure sensor	W3 = OC
F_8	Open circuit from pressure sensor to ground	W1 = OC
F_9	Pressure sensor has a bias	S1 = BIAS
F_{10}	Air tank has a leakage	T = LEAK
F_{11}	Short circuit to ground	W4 = SCTG
F_{12}	Open circuit from proportional valve to data port	W4 = OC
F_{13}	High friction in throttle valve	V1 = HIGH FRICTION
F_{14}	Throttle valve is stuck	V1 = STUCK
F_{15}	Proportional valve has a deviation from specification	V2 = DEV. SPEC.
F_{16}	Open circuit from PWM to Proportional Valve	W5 = OC
F_{17}	Open circuit from power supply to position sensor	W8 = OC
F_{18}	Open circuit from position sensor to data port	W7 = OC
F_{19}	Open circuit from position sensor to ground	W6 = OC
F_{20}	Position sensor has a bias	S2 = BIAS
F_{21}	Open circuit from fuel injector to ground	W9 = OC
F_{22}	Open circuit from power supply to fuel injector	W10 = OC
F_{23}	Fuel injector has a bias	I = BIAS

Table 3.1: Description of faults.

As a first step in applying the service view, the service of each component must be defined and its respective suppliers and customers identified. Identifying suppliers and customers requires an investigation of causal dependencies which demands an understanding of the systems functionality and terminology. The investigation begins by adding components one at a time starting in the PCRA class and ending up in the Application class. As components are added the service dependecies form a graph with no directed cycles. The reward of constructing the graph is that it can be used as the basis for the construction of a Bayesian network modeling the system, which will be described in chapter 6.

A description of each service provider with its provided service, suppliers, customers and number of variants is listed in Table 3.2. The service providers I, W9, W10, B5 and L4 are not affected by faults and are therefore not considered since they will not be needed in the Bayesian network.

As an example, consider module A4 which service is to set the current in the proportional

	Service	*Supplier(s)*	*Customer(s)*	*Variant(s)*
T	Deliver air	-	A1	1
V1	Adjust air flow	-	A3	1
V2	Deliver pressure	-	A4	1
W1	Supply power	-	S1	1
W2	Deliver passage to ground	-	S1	1
W3	Deliver voltage to port	-	S1	1
W4	Deliver passage to ground	-	A4	1
W5	Supply power	-	A4	1
W6	Supply power	-	S2	1
W7	Deliver passage to ground	-	S2	1
W8	Deliver voltage to port	-	S2	1
S1	Deliver voltage corresponding to pressure	W1,W2,W3	L1	1
S2	Deliver voltage corresponding to position	W6,W7,W8	L3	1
B1	Convert analog number to digital number	-	L1	1
B2	Convert analog number to digital number	-	L2	1
B3	Deliver power corresponding to duty cycle	-	A4	1
B4	Convert analog number to digital number	-	L3	1
L1	Deliver pressure measured by pressure sensor	B1,S1	A5	1
L2	Deliver current measured at input port	B2	A4	1
L3	Deliver position measured by position sensor	B4,S2	A6	1
A4	Set current in proportional valve	B3,L2,V2, W4,W5	A3	2
A5	Deliver pressure in tank	L1	A2	1
A6	Deliver throttle area	L3	A2,A3	1
A3	Set throttle valve position	A4,A6,V1	A1	2
A2	Deliver air flow through throttle valve	A5,A6	A1	3
A1	Control air flow through throttle valve	A2,A3,T	The Driver	1

Table 3.2: Description of service providers together with their provided service, suppliers, customers and number of variants.

valve. A4 does not know what the proportional valve affects since that is in the scope of application modules. A4 uses B3 to set the current so if B3 fails to deliver its service, A4 will fail too. Hence B3 is a supplier to A4 even though the signal flow is in the opposite direction. Further to utilize closed-loop control, A4 uses feedback of the current in the circuit from the module L2. Hence L2 is also a supplier to A4. To set the current in V2, A4 is also dependent of the valve itself to function properly. Therefore V2 is a supplier to A4. The scope of A4 is thus L2, B3 and physical data regarding V2, since A4 must know how to set the current properly.

Figure 3.5 shows the AFCS modeled in Simulink. A submodel view of the component classes

is employed. The Figure shows the communication of service statuses and signal flow between the classes.

3.3.1 Service monitoring and reconfiguration strategies

This section describes the service monitoring process and reconfiguration strategies utilized by the modules of the AFCS.

L1

The service of L1 is to deliver the pressure measured by the pressure sensor. The possible service classes of L1 are NOM, DIST and UNA i.e. $S_{L1} \in$ {UNA,DIST,NOM}. To perform its service L1 uses two suppliers, S1 and B1. S1 is a physical component and cannot communicate its service status. To improve fault tolerance against faults concerning S1, L1 uses a diagnostic test, T_{L1}. The test simply checks if the pressure is in range with respect to the physical limitations of the pressure sensor. The test result is 0 if the test has passed and 1 if the test fails [2]. The tests at this level of component classes cannot detect bias errors which is what the service class of DIST corrsponds to. Thus L1 is an example where a lower resolution of $\hat{S}_{L1}$ is used, i.e. $\hat{S}_{L1} \in$ {UNA,NOM}.

The VSSE then simply becomes:

If ($\hat{S}_{B1}$ = NOM & $T_{L1} = 0$)
$\hat{S}_{L1}$ = NOM
else
$\hat{S}_{L1}$ = UNA

The architecture of L3 is similar.

B1

The service of B1 is to convert the analog number at the input port to a digital number. The hardware component B1 does not have any suppliers. The possible service classes of B1 are NOM and UNA i.e. $\hat{S}_{B1} \in$ {UNA,NOM}. Inside the scope of B1 is a test, T_{B1}, which detects hardware malfunction. The test is for simplicity assumed to be ideal. The test result is 1 if the the hardware has a malfunction and 0 otherwise. The VSSE then simply becomes:

If ($T_{B1} = 0$)
$\hat{S}_{B1}$ = NOM
else
$\hat{S}_{B1}$ = UNA

[2] Usually a high-low test is standard.

The architecture of B2, B3, and B4 is similar.

A4

The service of A4 is to set the current in the proportional valve. A4 utilize three suppliers L2, B3 and V2. To increase fault-tolerance there are two variants A4:1 and A4:2. A4:1 uses closed-loop control while A4:2 uses open-loop control. A4:1 is dependent of L2 as feedback. B3 is a supplier to both variants since it delivers the power to the circuit. The default status of A4:1 is NOM. The default status of A4:2 is DIST since the performance of the open-loop is slightly worse. The default statuses have a clear meaning to the customer of the service, A3. Inside the scope of A4 is a diagnostic test, T_{A4}. The test checks if the feedback current is in range with respect to the physical limitations of the proportional valve. The test is sensitive to faults causing open circuits and short circuits to ground. If the test fails the strategy is to go to open mode i.e. select A4:2. This might seem dangerous in the case of an open circuit but the strategy is that such a situation will be sorted out higher up in the system hierarchy. The test result is 0 if the test passed and 1 if the test failed. Thus the VSSE for A4:1 becomes:

If ($\hat{S}_{B3}$ = NOM & $\hat{S}_{L2}$ = NOM & $T_{A4} = 0$)
$\hat{S}_{A4:1}$ = NOM
else
$\hat{S}_{A4:1}$ = UNA

and for A4:2:

If ($\hat{S}_{B3}$ = NOM)
$\hat{S}_{A4:2}$ = DIST
else
$\hat{S}_{A4:2}$ = UNA

The selector then decides which variant to use by the preference of using A4:1 ahead of A4:2.

If ($\hat{S}_{A4:1} \geq \hat{S}_{A4:2}$)
Select A4:1
else
Select A4:2

Figure 3.6 shows the architecture of A4.

A3

The service of A3 is to set the position of the throttle valve. A3 employs three suppliers A6, A4 and V1. To increase fault-tolerance there are two variants A3:1 and A3:2. A3:1 uses closed-loop control while A3:2 uses open-loop control. A3:1 is dependent of A6 for feedback. A4 is a supplier to both variants since it is a slave controller. The default status of A3:1 is NOM. The default status of A3:2 is DIST since the performance of the open-loop is slightly worse. If $\hat{S}_{A4}$ = DIST both variants of A3 will communicate their respective default statuses provided that A6 is NOM and no test has failed. This is because the use of A4:1 or A4:2 as a slave controller in this application is almost equally good.

A3 has a diagnostic test, T_{A3}. The test checks the integral size of the control error and its purpose is to be sensitive to high friction faults. If the test fails it is still better to be in closed-loop, but since the performance of the closed-loop is degraded, A3:1 is set to DIST. This means that the service class of DIST has two meanings for the customer of A3. An alternative is to use different levels of DIST but this leads to larger combinations of service statuses which is undesired if they are not needed.

The VSSE for A3:1 is:

If ($\hat{S}_{A6}$ = NOM & ($\hat{S}_{A4}$ = NOM | $\hat{S}_{A4}$ = DIST) & $T_{A3} = 0$)
$\hat{S}_{A3:1}$ = NOM
else if ($\hat{S}_{A6}$ = NOM & ($\hat{S}_{A4}$ = NOM | $\hat{S}_{A4}$ = DIST) & $T_{A3} = 1$)
$\hat{S}_{A3:1}$ = DIST
else
$\hat{S}_{A3:1}$ = UNA

and for A4:2:

If ($\hat{S}_{A4}$ = NOM | $\hat{S}_{A4}$ = DIST)
$\hat{S}_{A3:2}$ = DIST
else
$\hat{S}_{A3:2}$ = UNA

The selector of A3 then decides which variant to use by the preference of using A3:1 ahead of A3:2.

If ($\hat{S}_{A3:1} \geq \hat{S}_{A3:2}$)
Select A3:1
else
Select A3:2

A2

The service of A2 is to deliver the air flow through the throttle valve. A1 has two suppliers, A5 and A6 and three service classes, i.e. $S_{A2} \in \{\text{UNA,DIST,NOM}\}$. Further, to increase fault-tolerance A2 has three variants, A2:1, A2:2 and A2:3. A2:1 uses both of the suppliers to estimate the air flow. A2:2 only uses A5 while A2:3 only uses A6. The default status of A2:1 is NOM. The default status of A2:2 and A2:3 is DIST.

Included in the scope of A2 is a diagnostic test, T_{A2} as used in [8]. The test is based on the residual generator

$$\dot{P}_{at} = \frac{RT}{V}(-C\hat{A}_t\sqrt{P_{at}}(P_{amb} - P_{at})) \tag{3.4}$$

$$r_{A2} = P_{at} - \hat{P}_{at} \tag{3.5}$$

using a model of the air flow together with the estimated throttle area $\hat{A}_t$ delivered by A6. The pressure of 20 bar is used as the initial condition of the state P_{at}. A residual (3.5) is then formed as the difference of the calculated pressure and the pressure $\hat{P}_{at}$ delivered by A5. The test result is obtained by comparing the absolute value of the residual with a threshold value TH_{A2}. The test result is 0 if $r_{A2} \leq TH_{A2}$ and 1 otherwise.

If the test fails it is still better to use A2:1 since the test can not tell which one of A5 and A6 that has a bias. But since the performance of A2:1 is degraded, the service status of A2:1 is set to DIST.

The VSSE for A2:1 is:

If ($\hat{S}_{A5}$ = NOM & $\hat{S}_{A6}$ = NOM & $T_{A2} = 0$)
$\hat{S}_{A2:1}$ = NOM
else if ($\hat{S}_{A5}$ = NOM & $\hat{S}_{A6}$ = NOM & $T_{A2} = 1$)
$\hat{S}_{A2:1}$ = DIST
else
$\hat{S}_{A2:1}$ = UNA

and for A2:2:

If ($\hat{S}_{A5}$ = NOM)
$\hat{S}_{A2:2}$ = DIST
else
$\hat{S}_{A2:2}$ = UNA

and finally for A2:3:

If ($\hat{S}_{A6}$ = NOM)
$\hat{S}_{A2:3}$ = DIST
else

$\hat{S}_{A2:3}$ = UNA

The selector of A2 decides which variant to use by the following logic.

If ($\hat{S}_{A2:1} \geq \hat{S}_{A2:2}$ & $\hat{S}_{A2:1} \geq \hat{S}_{A2:3}$)
Select A2:1
else if ($\hat{S}_{A2:2} \geq \hat{S}_{A2:3}$)
Select A2:2
else
Select A2:3

A1

The service of A1 is to control the air flow through the throttle valve and thereby maintain the stoichiometric air-fuel ratio. A1 has three suppliers, A3, A2 and T. A1 has three service classes, i.e. $S_{A1} \in$ {UNA,DIST,NOM}, that in agreement with the final customer corresponds to how well the target ratio is followed over a period of time. Three service classes are used since it has a connection to Scania where UNA could signal a red warning light and DIST a yellow. The red light means that the system has been safely shut down and is no longer available. The yellow light means that the system is available but with a degraded performance.

Deviation from the stoichiometric ratio is described by the average absolute error

$$E = \frac{1}{T}\int_0^T |W_t(t) - W_{ref}(t)|dt \tag{3.6}$$

, which is assumed to be proportional to emissions. In (3.6) $W_t(t)$ is the true air mass flow, measured by a ideal sensor, and $W_{ref}(t)$ is the reference air mass flow at time t. The service classes of A1 corresponds to the size of (3.6) according to three limits

NOM: E $\leq l_{NOM}$
DIST: l_{NOM} < E $\leq l_{DIST}$
UNA: l_{DIST} < E

A1 has a diagnostic test, T_{A1}, which checks the integral size of the control error. The purpose of the test is to detect failures in critical system components such as A4, A3 and the air tank. If the test fails the service status of A1 is set to UNA with the estimation that the system is no longer controllable.

The VSSE of A1 is:

If ($\hat{S}_{A2}$ = NOM & $\hat{S}_{A3}$ = NOM & $T_{A1} = 0$)
$\hat{S}_{A1}$ = NOM
else if ($\hat{S}_{A2} \sim=$ UNA & $\hat{S}_{A3} \sim=$ UNA & $T_{A1} = 0$)

$\hat{S}_{A1} = \text{DIST}$
else
$\hat{S}_{A1} = \text{UNA}$

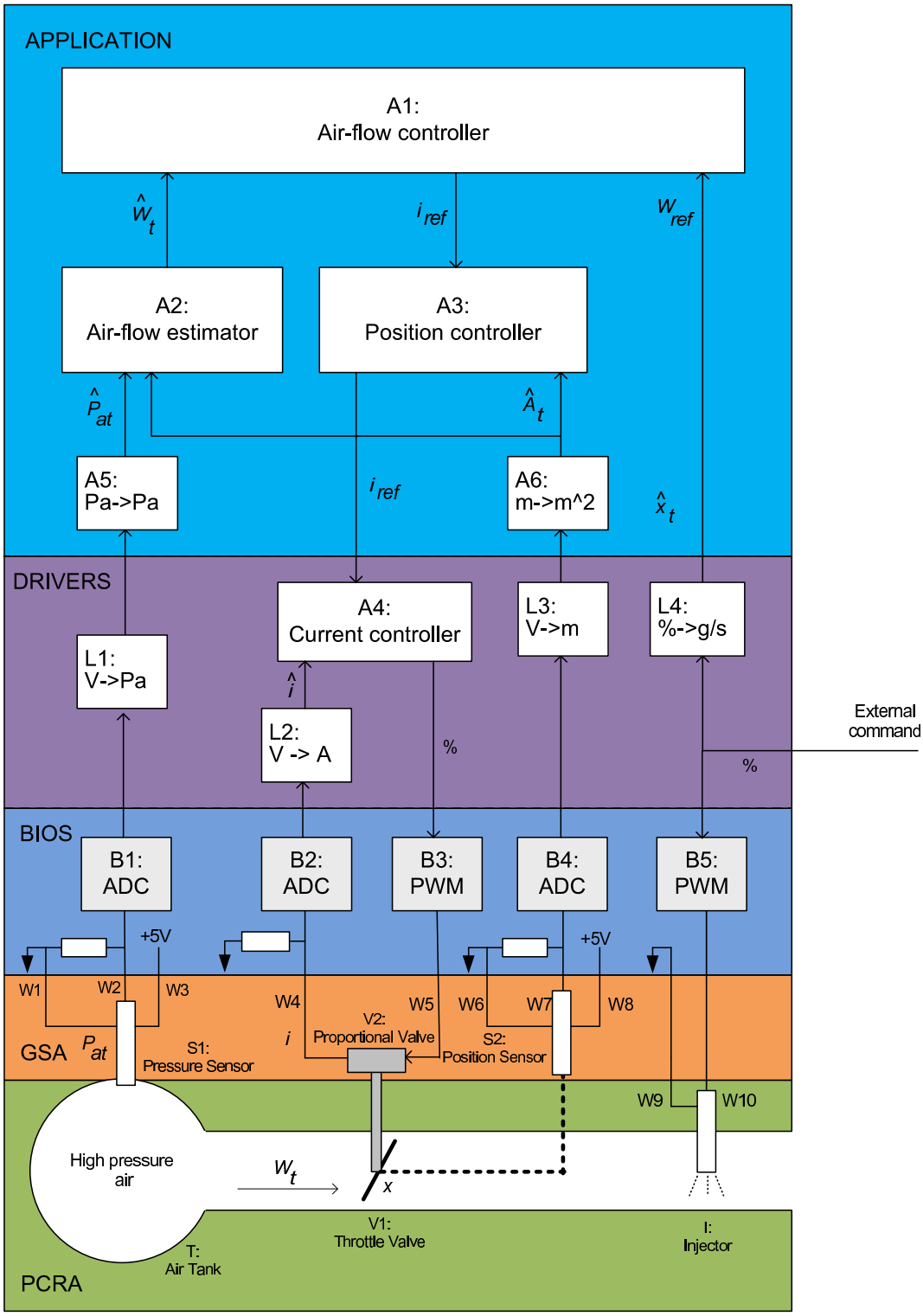

Figure 3.3: Components classified into their respective classes. GSA = Genereic sensors and actuators. PCRA = Physical components related to application.

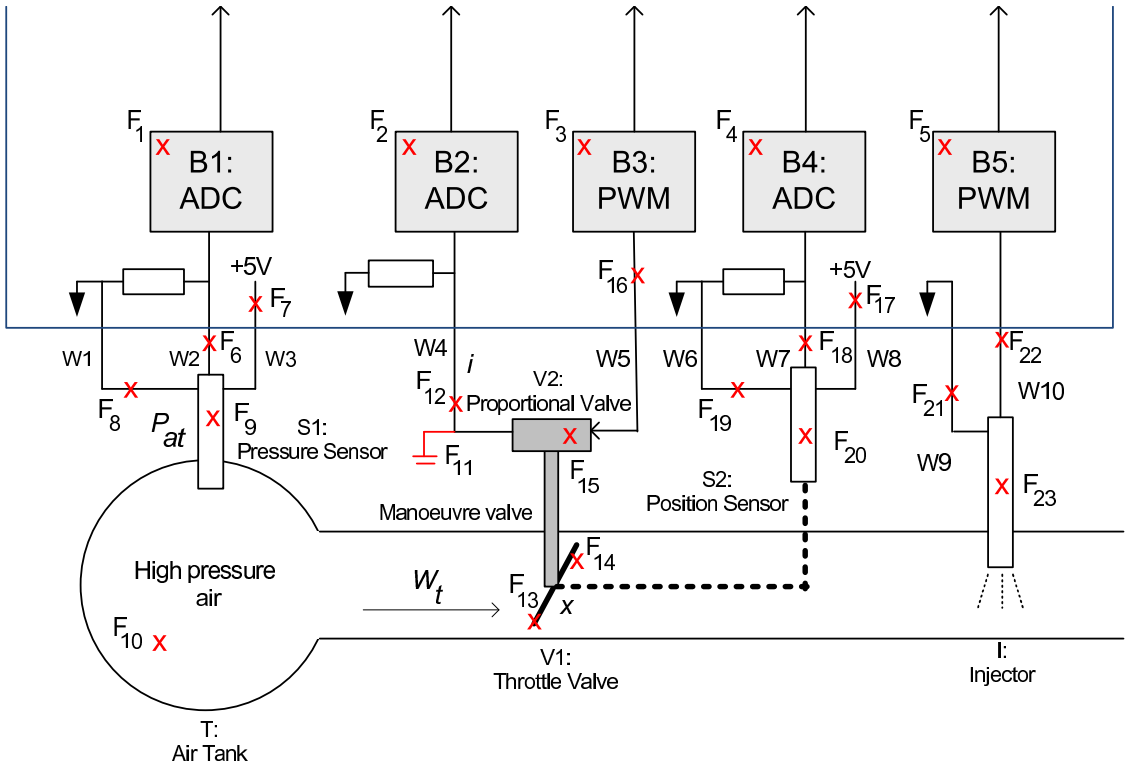

Figure 3.4: Location of faults in AFCS.

Figure 3.5: AFCS modeled in Simulink.

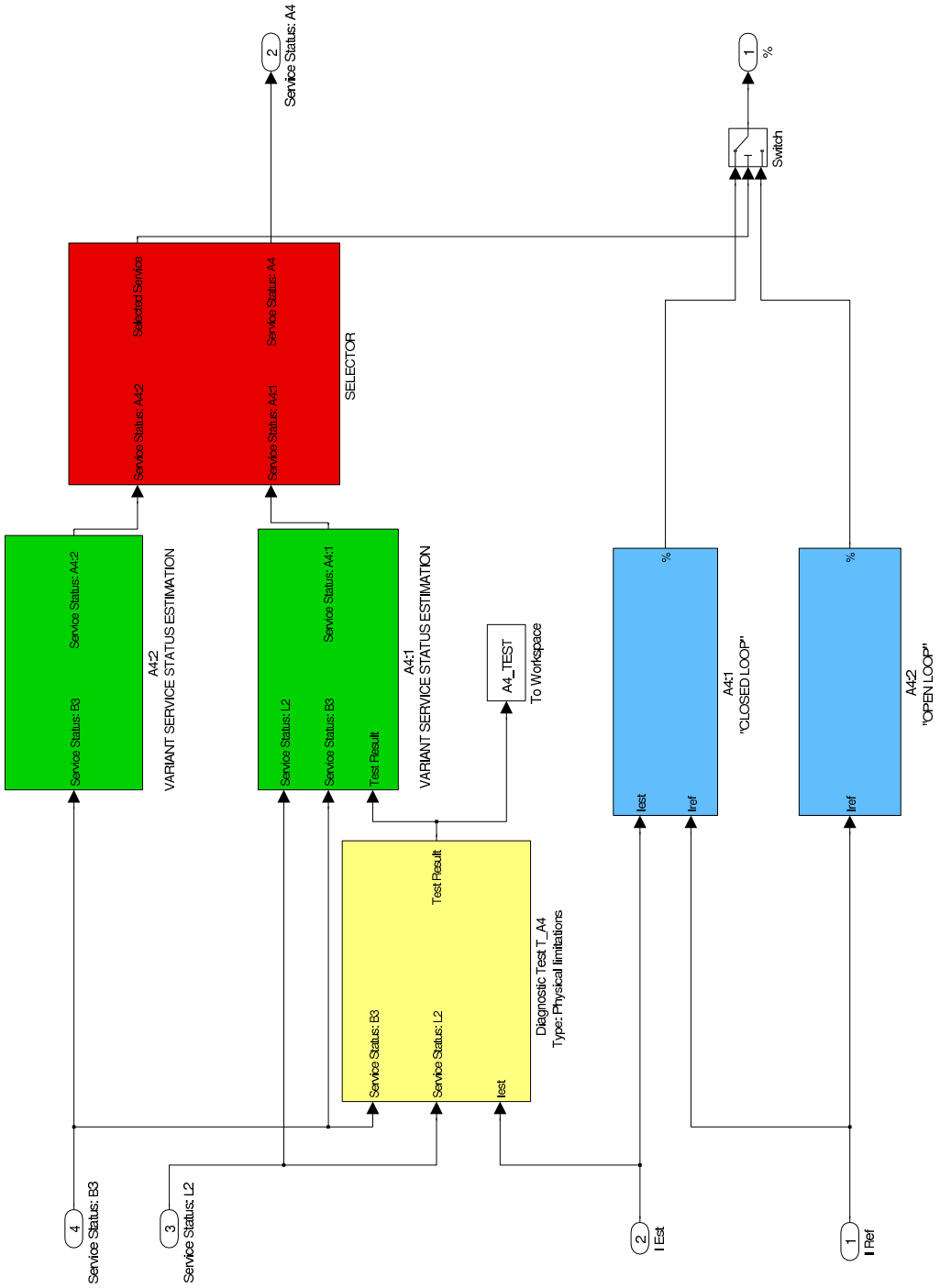

Figure 3.6: The architecture of A4.

Chapter 4

Fault-tolerant control of Air Flow Control system

The evaluation of the AFCS regarding its fault-tolerant properties is based on the three statements given in the introduction which are repeated below.

1. Continue system operation without (unbearable) loss of performance.
2. Continue system operation with reduced specifications.
3. Abandon the mission while avoiding disaster.

Given a fault that causes a failure in a non-critical system component the system operation should continue without or possibly with reduced specifications. If a fault causes a failure in a critical system component the customer of the AFCS must be informed in order to safely shut down the system and avoid a potential disaster. The system operation is defined to be the service of A1.

The faults described in Section 3.2 are implemented one by one and pair wise. For single faults a simulation is made in Simulink. The service status propagation is investigated and the communictaed status of A1, $\hat{S}_{A1}$, is compared with the true status, S_{A1}. For double faults a simulation is made in Simulink and the communicated status of A1, $\hat{S}_{A1}$, is compared with the true status, S_{A1}.

Figure 4.1 shows the nominal behaviour of the AFCS with no faults present. The left figure shows the plot of the stoichiometric ratio between air mass, measured by the ideal sensor, and fuel mass which usually is denoted with λ plotted against simulation time. The right figure shows how the true air flow, measured by the ideal sensor, follows the external command plotted against time.

The first section of this chapter shows the results for single faults and the second section describes the results for double faults.

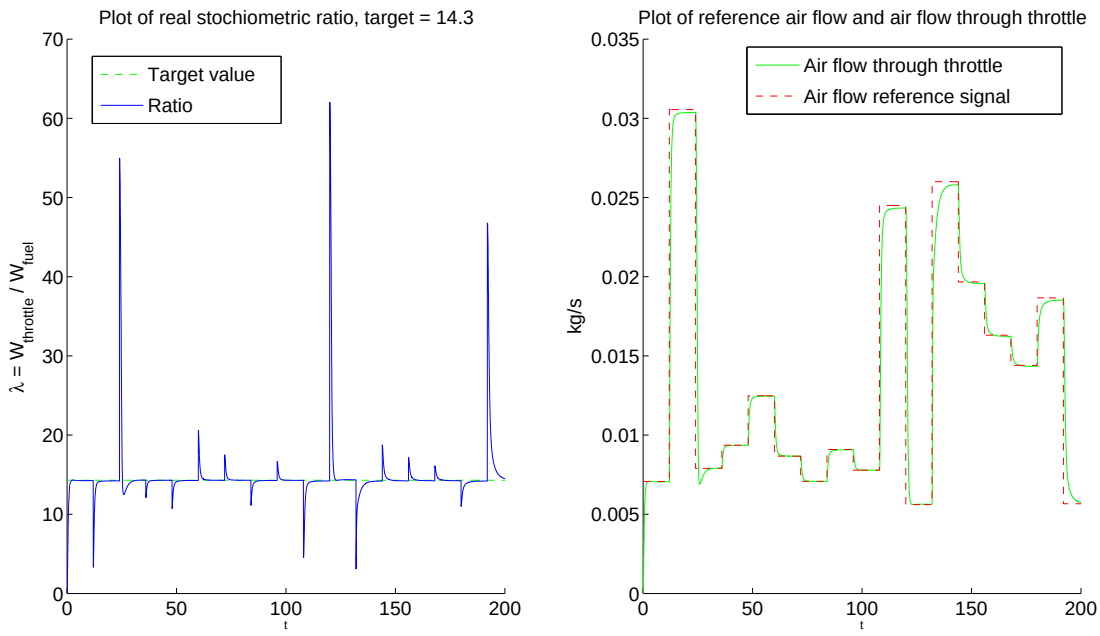

Figure 4.1: Left: Plot of stoichiometric ratio in nominal behaviour of the AFCS. Right: Plot of air flow measured by ideal sensor and reference air flow in nominal behaviour of the AFCS.

4.1 Single faults

Table 4.1 shows $\hat{S}_{A1}$ and S_{A1} for every single fault excluding S1 = BIAS, S2 = BIAS and V1 = HIGH FRICTION whose results will be presented individually.

Below follows a detailed description of the service status propagation and reconfigurations made by the AFCS under the presence of single faults.

B1 = MALF

B1 = MALF is detected in B1 and UNA is communicated to L1 and passed on to A2 via A5. A2 reconfigures to A2:3 and communicates DIST which makes A1 to communicate DIST.

B2 = MALF

B2 = MALF is detected in B2 which communicates UNA to A4 via L2. A4 reconfigures to A4:2 and communicate DIST to A3. A4 in open-mode used as a slave controller by A3 in closed-mode is almost as good as the case when A4 is in closed-mode. Therefore A3 communicates NOM which makes A1 deliver the service status NOM.

Fault	$\hat{S}_{A1}$	S_{A1}
B1 = MALF	DIST	DIST
B2 = MALF	NOM	NOM
B3 = MALF	UNA	UNA
B4 = MALF	DIST	DIST
W2 = OC	DIST	DIST
W3 = OC	DIST	DIST
W1 = OC	DIST	DIST
T = LEAK	UNA	UNA
W4 = SCTG	NOM	NOM
W4 = OC	UNA	UNA
V1 = STUCK	UNA	UNA
W5 = OC	UNA	UNA
W8 = OC	DIST	DIST
W7 = OC	DIST	DIST
W6 = OC	DIST	DIST

Table 4.1: Comparison of communicated and true status of A1 for single faults excluding S1 = BIAS, S2 = BIAS, V1 = HIGH FRICTION.

B3 = MALF

B3 = MALF is detected in B3 which communicates UNA to A4. B3 is a supplier to both variants of A4 which makes A4 to communicate UNA. A3 communicates UNA since A4 is a supplier to both variants of A3. A1 communicates UNA since A3 is a supplier to its service.

B4 = MALF

B4 = MALF is detected in B4 and UNA is communicated to L3 and passed on to A2 and A3 via A6. A2 reconfigures to A2:2 and communicates DIST. A3 reconfigures to A3:2 and communicates DIST. $\hat{S}_{A2} = \hat{S}_{A3}$ = DIST makes A1 communicate DIST.

Figure 4.2 shows the performance of the AFCS in the presence of B4 = MALF activated at simulation time 10. The slower response is due to A3 being in open mode. The offset error is due to A2:2 which estimates the air flow using the ideal gas law with inaccurate values of the physical values of the air tank. The plots illustrates the degraded but acceptable situation of S_{A1} = DIST.

W1 = OC, W2 = OC, W3 = OC

The service status propagation of W1 = OC, W2 = OC, W3 = OC is similar to the case of B1 = MALF described above. W1 = OC, W2 = OC, W3 = OC are detected with T_{L1} in L1 which communicates UNA. This gives the same result as in the case of B1 = MALF.

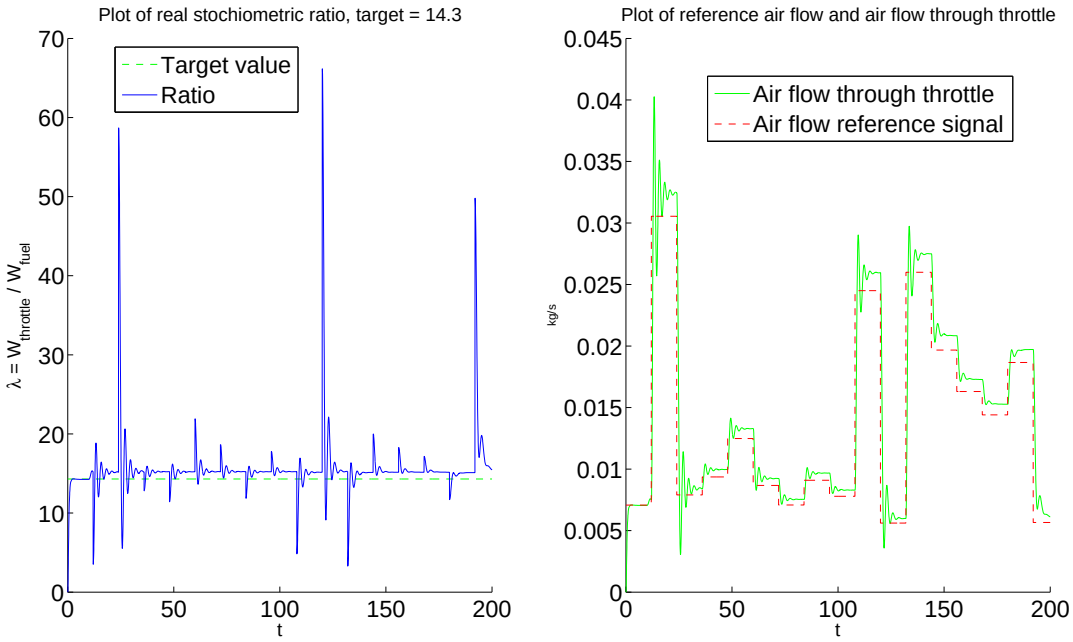

Figure 4.2: Left: Plot of stoichiometric ratio of the AFCS in the presence of B4 = MALF activated at t = 10. Right: Plot of air flow measured by ideal sensor and reference air flow of the AFCS in the presence of B4 = MALF activated at t = 10.

S1 = BIAS, S2 = BIAS

The bias error of S1 and S2 is varied from an underestimation of 25% to an overestimation of 25% of the true value with a stepsize of 0.05%. The result for S1 = BIAS is shown in Figure 4.3. The result for S2 = BIAS is shown in Figure 4.4.

For S1 = BIAS there is a deviation between true and communicated service status below -15%. For S2 = BIAS there is a deviation below -20% and for -5% and 5%.

A sound principle is always to let the customer assume that the true service is better than the communicated one which is not the case for the deviations described above. In order to show how this issue can be resolved the orginal threshold for biases is lowered and a second, larger, threshold value is introduced, $TH2_{A2}$. If $r_{A2} > TH_{A2}$ the estimated service status of A2 is set to UNA which implies that A1 is set to UNA.

Simulations are performed with a second threshold. Figure 4.5 shows the result for S1 = BIAS. Figure 4.6 shows the result for S2 = BIAS.

The figures show that the issue is resolved at the cost of communicating UNA for S1 = BIAS between -20% and -10%.

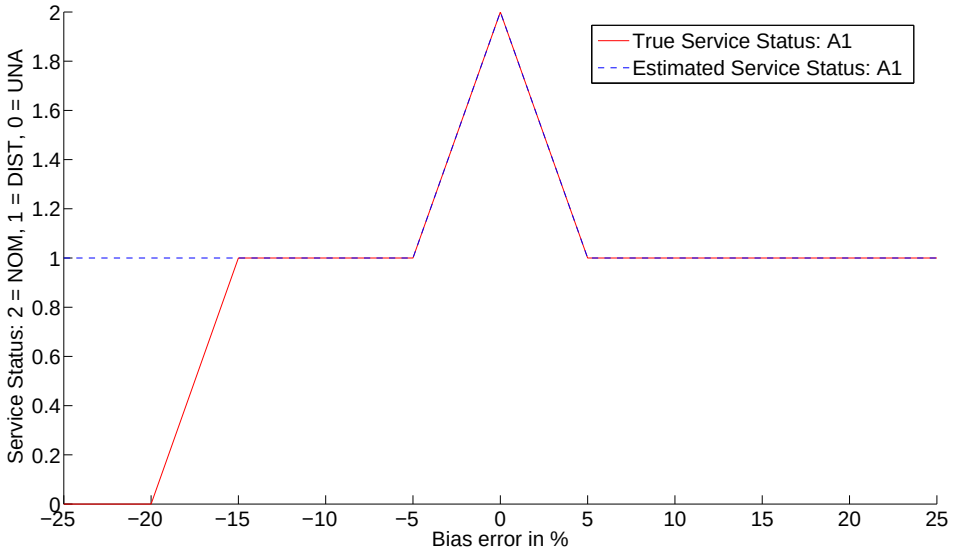

Figure 4.3: True service status of A1 plotted with the estimated service status of A1 as a function of S1 = BIAS in %.

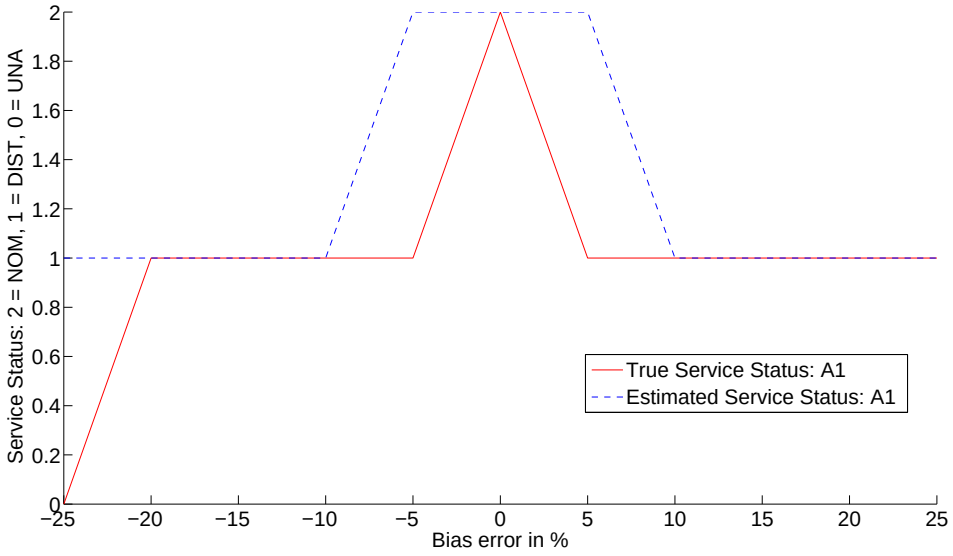

Figure 4.4: True service status of A1 plotted with the estimated service status of A1 as a function of S2 = BIAS in %.

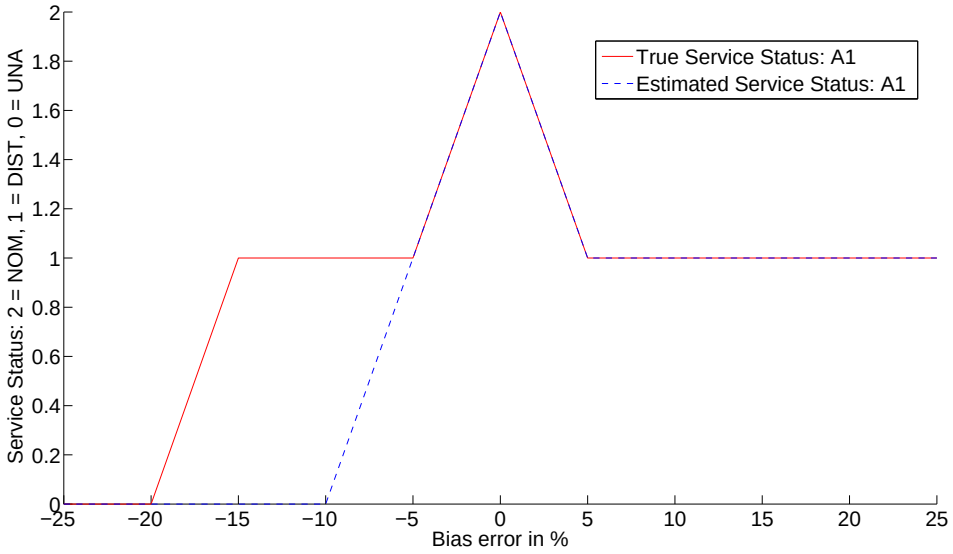

Figure 4.5: True service status of A1 plotted with the estimated service status of A1 as a function of S1 = BIAS in % with a second threshold in T_{A2} added.

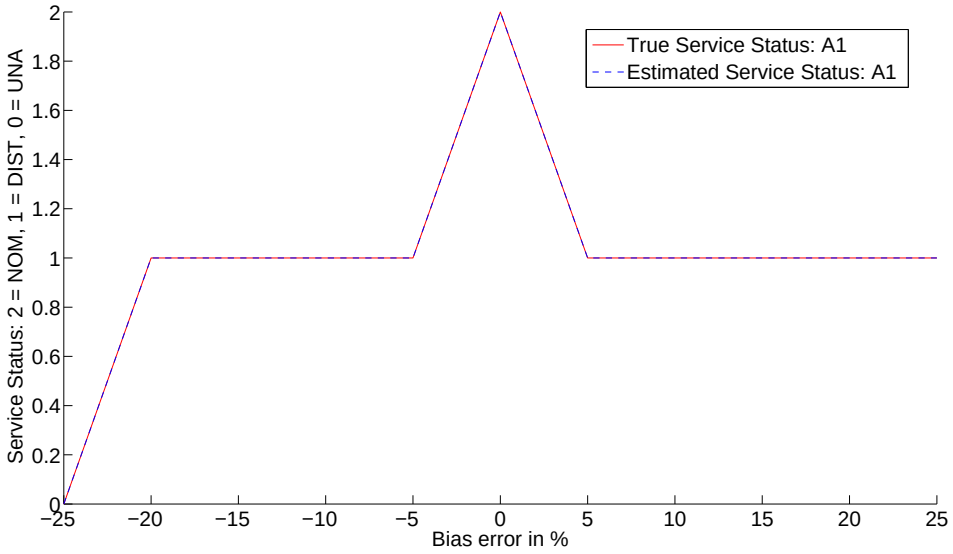

Figure 4.6: True service status of A1 plotted with the estimated service status of A1 as a function of S2 = BIAS in % with a second threshold in T_{A2} added.

T = LEAK

T = LEAK makes T_{A1} to fail which makes A1 communicate UNA.

W4 = SCTG

W4 = SCTG makes T_{A4} fail. A4 reconfigures to A4:2 and communicates DIST to A3. This gives the same result as in the case of B2 = MALF.

W4 = OC, W5=OC

W4 = OC, W5=OC makes T_{A4}, T_{A3} and T_{A1} fail. A4 reconfigures to A4:2. A3 communicates DIST. A1 communicates UNA since T_{A1} has reacted to the loss of control of the throttle valve.

V1 = HIGH FRICTION

The friction coefficient is varied from 1 to 14.5 in steps of 0.5. The nominal value is 0.8. The result is shown in Figure 4.7. There is a deviation between true and communicated service

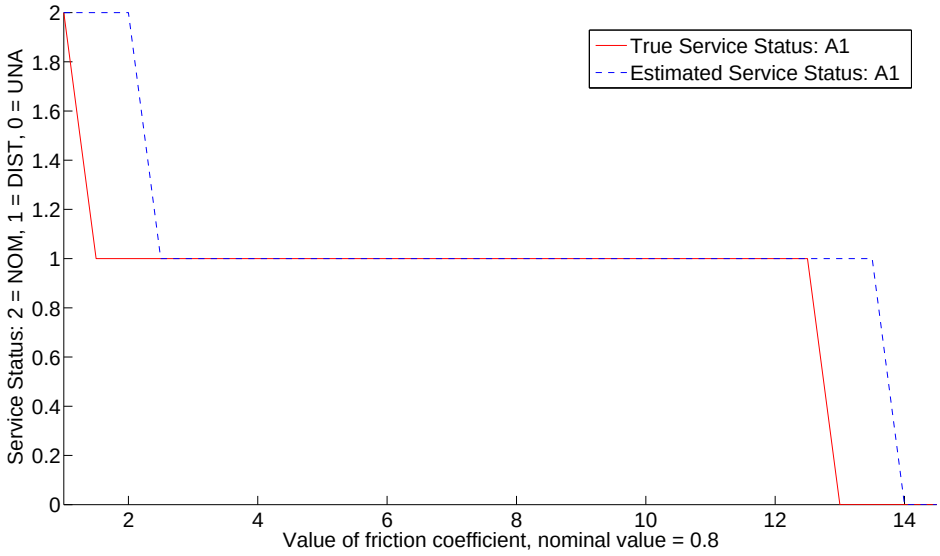

Figure 4.7: True service status of A1 plotted with the estimated service status of A1 as a function of the friction of the throttle valve.

status for friction values of 2-3. This could be resolved by lowering the threshold value of T_{A3} making it more sensitive. There is also a small deviation for friction values of 13.0-13.5. This issue can be resolved by either making T_{A1} more sensitive by lowering the threshold value used in the test or, as in the case of bias faults, add a second higher threshold in T_{A2}.

V1 = STUCK

V1 = STUCK causes T_{A3} and T_{A1} to fail. A3 employs A3:1 and communicates DIST. A1 communicates UNA since T_{A1} has reacted to the loss of control of the throttle valve.

W6 = OC, W7 = OC, W8 = OC

The service status propagation of W6 = OC, W7 = OC, W8 = OC is similar to the case of B4 = MALF described above. W6 = OC, W7 = OC, W8 = OC are detected with T_{L3} in L3 which communicates UNA. This gives the same result as in the case of B4 = MALF.

4.2 Double faults

The analysis of double faults is made for all $\binom{18}{2} = 153$ combinations of faults where S_{A1} is compared with $\hat{S}_{A1}$. S1 = BIAS is set to 10% underestimation. S2 = BIAS is set to 10% underestimation. V1 = HIGH FRICTION is set to 8.

Significant results are S1 = BIAS and V1 = HIGH FRICTION and the pairs of either B1 = MALF, W1 = OC, W2 = OC or W3 = OC with T = LEAK. All cases result in the communicated status of DIST while the true status is UNA.

B1 = MALF, W1 = OC, W2 = OC or W3 = OC makes A2 reconfigure to a variant which uses a substitute value of the pressure in the tank. It then becomes impossible for T_{A1} to detect T = LEAK. This is a serious problem that in the current system architecture cannot be resolved. If this is a common double fault the addition of a redundant pressure sensor might be considered. A reconfiguration to the redundant sensor can be executed in L1.

S1 = BIAS and V1 = HIGH FRICTION lead to a degraded performance of A2 and A3 which sets A1 to UNA. The problem could be solved by using extended service statuses in the service providers to make a more detailed service status propagation available. For an example DIST in A3 could be extended with DIST_1 in the case of T_{A3}=1. The combination of $\hat{S}_{A2}$ = DIST and $\hat{S}_{A3}$ = DIST_1 could then set $\hat{S}_{A1}$ = UNA.

It is clear that extending the service statuses and tests of service providers would lead to increased fault-tolerance. But this may become suboptimal since it enlarges the complexity of the system and the if-else logic which increases the risk of human errors when constructing the variant service status estimation.

Part II

Bayesian network for diagnosis

Chapter 5

Bayesian networks

The joint probability distribution can answer any question about the domain of some random variables but can become intractably large as the number of variables grows. Independence and conditional independence relationships among variables can greatly reduce the number of probabilities that need to be specified in order to define the joint probability distribution.

This chapter introduces a local data structure called a Bayesian network to represent dependencies among variables and to give a concise specification of any joint probability distribution. The Bayesian networks was discovered in the middle 1980s to solve the intractability problem of acquiring the joint probability distribution. Definitions are highlighted with italic letters. The interested reader is recommended to view [9] for an introduction or [7] for a more thorough treatment of the theory of Bayesian networks.

5.1 Basic concepts

The full specification of a Bayesian network is as follows:

1. A set of random variables makes up the nodes of the network. Variables may be discrete or continuous.

2. A set of directed links or arrows connects pairs of nodes. If there is an arrow from node X to Y, X is said to be a *parent* of Y and Y is a *child* to X.

3. Each node X_i has a conditional probability distribution $P(X_i|Parents(X_i))$ that quantifies the effect of the parents on the node.

4. The graph is a DAG, i.e. has no directed cycles.

The *topology*, the set of nodes and links, specifies the conditional independence relationships that hold in the domain in a way that is explained by the following example taken from [9]:

Consider the simple example consisting of the boolean variables Toothache, Catch(the dentist's steel probe catches in the tooth), Cavity and the discrete variable Weather which can have four values.

One can argue that one's dental problems do not influence the weather and therefore Weather is independent of Cavity, Catch and Toothache. Further it would be nice if Toothache and Catch were independent but they are not: if the probe catches in the tooth it probably has a cavity and that probably causes a toothache. However these variables are conditionally independent given the presence or absence of a cavity. Each is directly caused by the cavity but neither has a direct effect on the other.

These relationships are represented by the topology shown in Figure 5.1. The independence

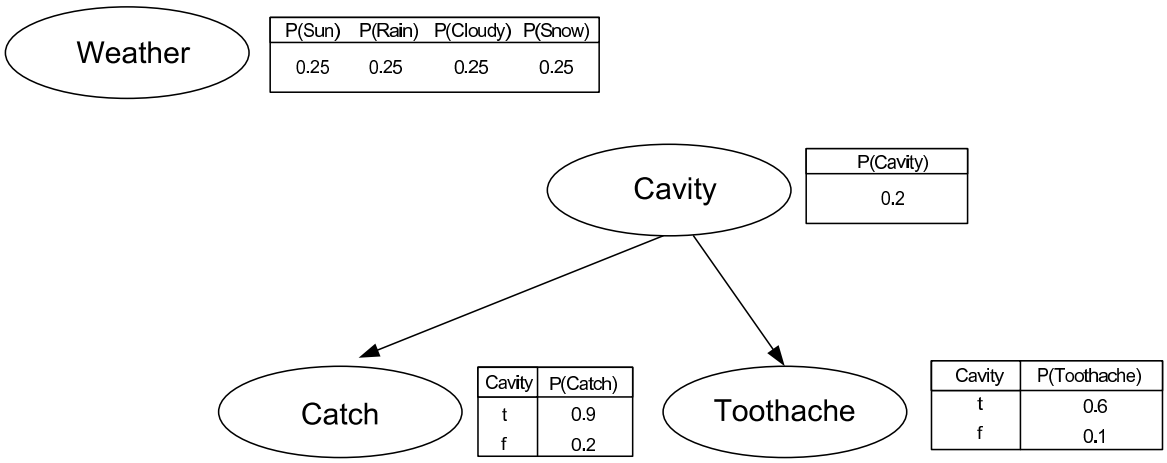

Figure 5.1: A Bayesian network in which Weather is independent of the other three variables and Toothache and Catch are conditionally independent given Cavity.

of Weather of the other three variabels is represented by the absence of links between the node and the other nodes. Formally the conditional independence of Toothache and Catch given Cavity is indicated by the absence of a link between Toothache and Catch. Intuitively the topology represents the fact that Cavity is a direct cause of Toothache and Catch whereas no direct causal relationship exists between Toothache and Catch.

Figure 5.1 also shows some annotated conditional distributions. Each distribution is shown as a *conditional probability table* or CPT which is suitable for discrete variables. Each row in a CPT contains the conditional probability of each node value for a *conditioning case*. A conditioning case is a possible combination of values for the parent nodes. Each row must sum to 1 because the entries represent an exhaustive set of cases for the variable.

The theorem below will show how a combination of the topology and the conditional distributions suffices to (implicitly) specify the full joint distribution for all the variables. From now on let $P(x_1, ..., x_n)$ denote the joint probability distribution of the random variables $X_1, ..., X_n$.

The chain rule for Bayesian networks 5.1.1. *Let BN be a Bayesian network over the random variables $X_1, ..., X_n$. Then BN specifies a unique joint probability distribution $P(x_1, ..., x_n)$ given by the product of all conditional probability tables specified in the BN:*

$$P(x_1, ..., x_n) = \prod_{i=1}^{n} P(X_i | Parents(X_i)), \tag{5.1}$$

where Parents(X_i) are the parents of X_i in the BN and $P(x_1, ..., x_n)$ reflects the properties of the BN.

The interested reader can find the proof in [7].

A Bayesian network is a locally structured network. Each node interacts directly with a bounded number of other nodes. In the general case each random variable is directly influenced by at most k others, for some constant k. If it is assumed for simplicity that there are n boolean nodes the amount of information needed to specify a CPT is at most 2^k numbers and the total network can be specified by at most $n2^k$ numbers. This last amount of numbers should be put in contrast with the joint probability distribution which contains 2^n numbers.

5.2 Where do the numbers come from?

This section aims to give a brief introduction on how to populate the CPTs with probabilities. The most common sources of information are statistical data, literature and expert knowledge. In this thesis expert knowledge is used as the sole source of information since no statistical data is available. Thus this introduction will be focusing on expert knowledge but the interested reader can see [4] for an introduction to data-rich applications.

5.2.1 Prior probabilities

For the root nodes with no parents, the CPT reduces into an unconditional probability table with prior probabilities. In diagnosis these are hypothesis nodes that represent uncertain events which are unobservable but of interest to get estimates of certainty for.

If there is no hunch that a physical component is more likely to fail than any other the prior probabilties are set to the same value in order to not introduce any bias.

If reliability data like *mean time between unscheduled removals (MTBUR)* or *mean time between failure (MTBF)* are available, better prior probabilities can be estimated [10].

5.2.2 Conditional probabilities

When there is little or no statistical data available, expert knowledge constitute the remaining source of probabilistic information. According to [4] the role of domain experts should not be

underestimated. An expert can help with assessing the required probabilities and also refine crude estimates obtained from other sources. In a company like Scania, expert knowledge can be obtained from system engineers responsible for designing and building the ECUs.

There are some difficulties of using expert knowledge. Foremost the amount of probabilities that need to be assigned can be very large. Given that the expert's time is an expensive commodity it may become impossible to obtain all numbers.

One idea in order to become more efficient is given in [10] where the elicitation procedure starts with a crude estimation of probability parameters. In the next step sensitivity analysis is performed to identify the most influential parameters. Finally more accurate values of those identified parameters can be obtained by a more careful assessment made by the responsible domain experts. This procedure is performed iteratively until satisfactory behaviour is achieved.

Another difficulty is the possibility that the domain experts do not think in probabilistic reasoning. Techniques for resolving this issue are developed [5].

5.3 noisy-OR

Even if the number of boolean parents k to a boolean node is small, filling in the CPT would still take up to 2^k numbers. If the distributions are taken from a database, the number of cases for each configuration may become to small. Also the configurations may be to specific for any expert. The expert can be in a situation where he can give reasonable estimates of $P(A|B)$ and $P(A|\,C)$ but not the required $P(A|B,C)$.

But this is in fact a worst case scenario in which the relation between parents and children is completely arbitrary. When the relation is not arbitrary the relation can be described by a parametric distribution that fits some standard pattern. In such cases the complete CPT can be specified by naming the pattern and a few parameters.

One important uncertainty relationship is the *noisy-OR* relation. This relationship has been used extensively in artificial intelligence both in purely probabilistic models and in models that combine probability and logic, and particulary for models that have a causal basis [3]. The noisy-OR relationship and its natural existence in a causal model is now explained by a simple example taken from [9]:

In propositional logic it is said that Fever(X) is true if and only if Flu(Y_1), Cold(Y_2) or Malaria(Y_3) is true. The noisy-OR relation is a generalization of the logical OR and allows for uncertainty about the ability of each parent causing the child to be true. The causal relation between parent and child may be *inhibited*, that is a person can have a cold but not exhibit fever. The noisy-OR relation makes two assumptions:

1. It assumes all causes are listed.

2. The inhibition of each parent is independent of the inhibition of any other parent.

The first assumption is not so strict as it seems beacuse a *leak* node that covers "miscellaneous" causes can be added. Given the assumptions above *Fever* is false iff all its true parents are inhibited and the probability of this is the product of the inhibition probabilities for each parent. Let us suppose that these individual probabilities given by the domain expert, in this case probably a doctor, are as follows:

$$P(\neg\, fever|\ cold, \neg\, flu, \neg\, malaria) = p_1 \tag{5.2}$$
$$P(\neg\, fever|\ \neg\, cold, flu, \neg\, malaria) = p_2 \tag{5.3}$$
$$P(\neg\, fever|\ \neg\, cold, \neg\, flu, malaria) = p_3 \tag{5.4}$$

Using the probabilities above the whole CPT can be constructed using the noisy-OR distribution and the inhibition probabilities

$$P(\neg\, fever|\ Cold, Flu, Malaria) = \prod_{i:Y_i=True} p_i \tag{5.5}$$

In general for noisy-OR relationships in which a variable depends on k parents the CPT can be described using O(k) instead of O(2^k) numbers. The noisy-OR model can be generalized to variables having more than two states, in this form it is called *noisy-MAX*.

5.4 Software tools for Bayesian networks

Even though Bayesian networks are efficient languages for building models with inherent uncertainty, it is a tedious job to perform probabilistic calculations and to fill in the CPTs even for very simple networks. Fortunately software tools that can perform these tasks are available. For this thesis the following properties of a software tool was identified to be desirable:

1. A GUI since it will provide an overview greatly reducing risk of node numbering errors and CPT elicitation errors.

2. The possibility to script since it is desirable to make inference about the major part of the network.

3. Nodes which support Noisy OR/MAX (and Noisy AND distribution).

4. Free, or at least do not cost very much money.

A small investigation was made and the software tools SMILE and GeNIe developed by the Decision Systems Laboratory of the University of Pittsburgh was found to fulfill all of the desiderates given above.

GeNIe is a graphical editor which allows the creation and modification of network models. It is platform dependent and works in Windows. SMILE is the engine of GeNIe consisting of C++ classes and thus platform independent. Both GeNIe and SMILE are free of cost.

5.4.1 Probabilisitc inference in GeNIe

The two major problems of using probability for managing uncertainty are the intractability of acquiring a joint probability distribution with a large number of variables and the intractability of computing posterior probabilities for probabilistic inference. While the Bayesian network seems to be a quite successful solution to the first problem the second problem remains. The task of computing posterior probabilities for probabilistic inference in a Bayesian network is NP-hard [2].

The default algorithm for exact inference in GeNIe is the clustering algorithm which should be sufficient for most applications. Only when networks become very large and complex, the clustering algorithm may not be fast enough. In that case, it is suggested by the developers of GeNIe that the user choose an approximate algorithm, preferably the EPIS-sampling algorithm.

Chapter 6

Bayesian network of Air Flow Control system

As stated in the introduction of this thesis it is highly desirable to have a model of the fault-tolerant system in order to perform troubleshooting. The troubleshooting problem asks the model: Given this set of observations, what is the most probable faulty component? A Bayesian network is a wellsuited tool for this kind of reasoning under uncertainty.

This chapter explains how to construct a Bayesian network of the AFCS using the service dependencies of the service providers as a foundation. As described in chapter 5 a Bayesian network consists of its topology together with the nodes' corresponding CPTs.

Building the network consists of three tasks. The first of these is to identify the variables of importance along with their possible states. When these are identified the next task is to identify the relationships between the variables and to express it in a graphical structure. The third and last task is to obtain the probabilities for the quantative part of the network.

The first section of this chapter describes how the topology is created as described in [8]. The second section explains how the CPTs are populated. The third section shows the network implemented in the software tool GeNIe.

6.1 Constructing the topology

The desire to use the Bayesian network for troubleshooting implies that the network should include all of the service providers of the AFCS - the possible faulty components, which includes the physical components and hardware, along with the software within the ECU.

The creation of the topology starts with the physical components and hardware that are parentless root nodes of the network. Then it continues with the software within the ECU.

6.1.1 Physical components and hardware

For every physical component and hardware component the possible states are identified. All the components have the value OK which correponds to nominal behaviour. The other possible values correspond to failure modes. The failure modes should have names with a clear meaning for the mechanic performing the troubleshooting. For an example W4 has three possible states OK, OC and SCTG. OC means that the wire is causing an open circuit. SCTG means that the wire is causing a short circuit to ground. All physcial components and hardware have links to the software within the ECU according to the service dependencies defined in Table 3.2.

6.1.2 Software

The topology of service providers that are software components consists of two sets of nodes. One stochastic and one deterministic. Figure 6.1 shows the topology of A3 which will be used as an illustrative example.

Deterministic part

The left part of Figure 6.1 shows the deterministic part which has rectangular shaped nodes. It consists of nodes representing the estimated service statuses of the variants, $\hat{S}_{A3:1}$ and $\hat{S}_{A3:2}$, the selector, Se_{A3}, and $\hat{S}_{A3}$ which represents the estimated service status of the service provider. The possible values of $\hat{S}_{A3:1}$ and $\hat{S}_{A3:2}$ are exact copies of the service classes used in the AFCS. The possible values of $\hat{S}_{A3}$ are exact copies of the possible values of $\hat{S}_{A3:1}$ and $\hat{S}_{A3:2}$. The possible values of Se_{A3} are the variants. $\hat{S}_{A3:1}$ and $\hat{S}_{A3:2}$ have links to the selector and $\hat{S}_{A3}$. Input links to $\hat{S}_{A3:1}$ and $\hat{S}_{A3:2}$ are the estimated service statuses of suppliers and diagnostic test results. The selector controls the variant to be used and has links to $\hat{S}_{A3}$ and to a stochastic node S_{A3} which represents the service status of the service provider.

Stochastic part

The right part of Figure 6.1 shows the stochastic part which has circular shaped nodes. The service dependencies defined in Table 3.2 are used as the foundation when building the stochastic part. $S_{A3:1}$ and $S_{A3:2}$ represent the service status of the variants. They have input links from their suppliers. The possible values of $S_{A3:1}$ and $S_{A3:2}$ may be extended from the possible values of $\hat{S}_{A3:1}$ and $\hat{S}_{A3:2}$. This is done with an analysis of how the faults in the system influence the service providers. The purpose is that the service provider can be given a more detailed resolution of how it will influence a diagnostic test. $S_{A3:1}$ and $S_{A3:2}$ have links to S_{A3} which possible values are exact copies of the statuses of the variants. Included in the stochastic part is also nodes for possible diagnostic tests, in this case T_{A3}. It has incoming links from suppliers which influence the test result. The possible values of T_{A3} are {PASS,FAIL} representing the binary test results used in the AFCS.

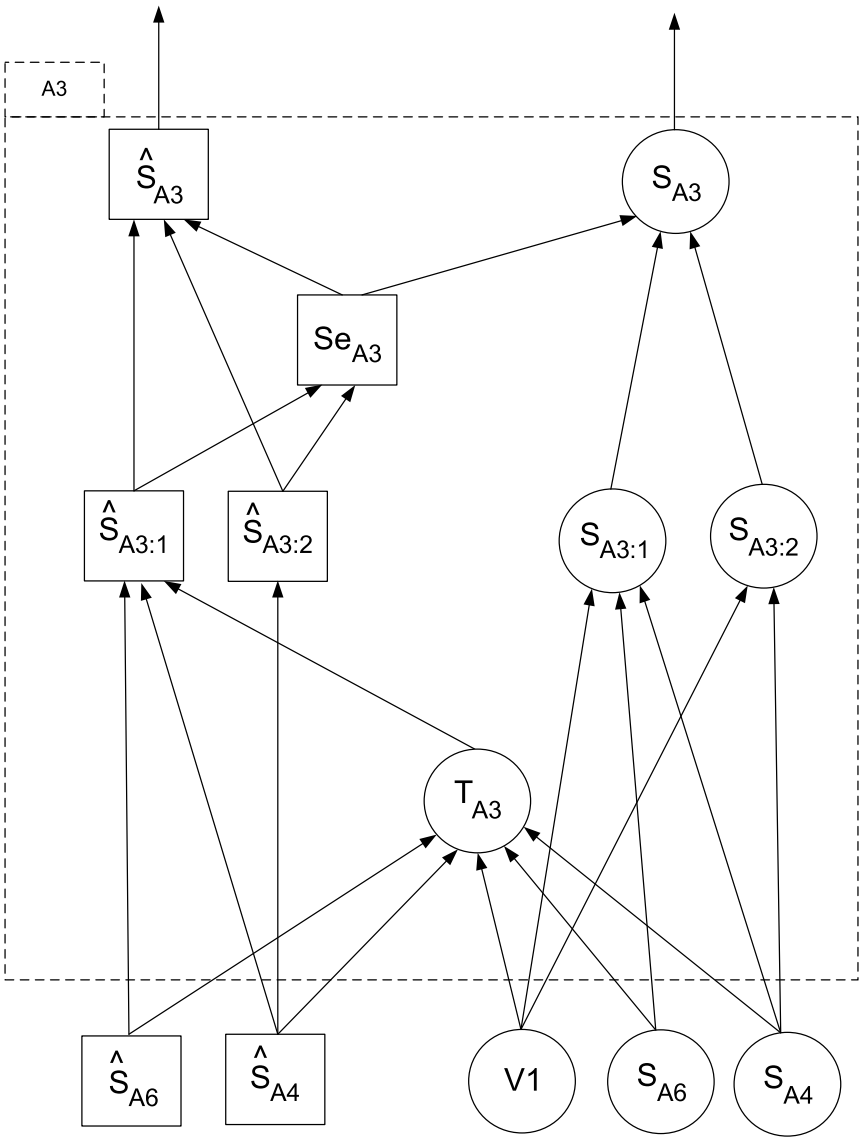

Figure 6.1: Topology of A3.

6.2 CPTs

Deterministic

The CPTs of $\hat{S}_{A3:1}$ and $\hat{S}_{A3:2}$ are deterministic and recompose the if-else logic used in the AFCS for variant service status estimation. The CPT of Se_{A3} is deterministic and represents the if-else logic used in the AFCS for selection of variant. The CPT of $\hat{S}_{A3}$ is deterministic and outputs the estimated service status of the variant the selector selects. The CPT of S_{A3} is deterministic

and outputs the service status of the variant the selector selects.

Stochastic

The Prior probabilities for faults in physical components and hardware are set to the same value 0.01 since no knowledge exists to believe a fault is more probable than any other.

All the conditional probabilities come from the expert of the system which in this case is the author of this thesis. The probabilities are set by direct questions. In order to be consequent the same values of the probabilities are used matching the probability scale with its corresponding verbal estimates shown in Figure 6.2.

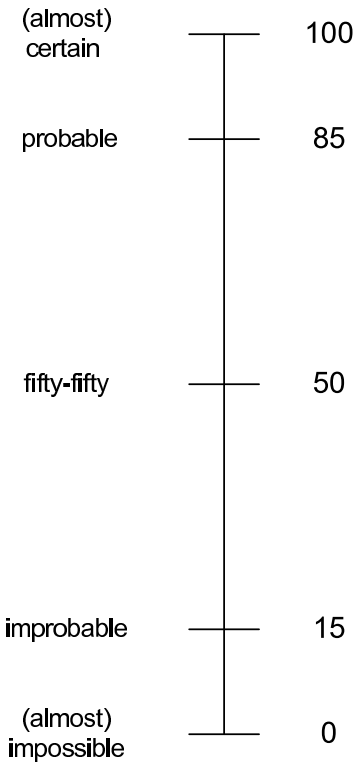

Figure 6.2: Probability scale for the assessment of conditional probabilities.

To reduce the number of probability parameters the noisy-OR model and its generalization noisy-MAX are used.

As an example Table 6.1 shows the nosiy-or model with no leak of T_{A2} with its inhibition probabilities. From the 12 inhibition probabilities, GeNIe automatically constructs the whole CPT of 72 numbers using the noisy-OR distribution. S_{A5} = DIST is an extended service status of A5 with the meaning that A5 has a bias. The conditional probability P(T_{A2} = FAIL | S_{A5} = DIST) is set to 0.95 indicating that it is probable that the test will fail but not certain since

Parent	$\hat{S}_{A5}$		S_{A5}			S_{A6}			$\hat{S}_{A6}$		LEAK
State	UNA	NOM	UNA	DIST	NOM	UNA	DIST	NOM	UNA	NOM	
FAIL	1	0	1	0.95	0	1	0.95	0	1	0	0
PASS	0	1	0	0.05	1	0	0.05	1	0	1	1

Table 6.1: CPT of T_{A2}.

the distribution of a bias error is unknown and the resolution of S2 = BIAS is very low in order to keep down the size of the CPTs. The probability P(T_{A2} = FAIL | $\hat{S}_{A5}$ = UNA) = 1 is the model solution of the principle that a test fails when a supplier is communicating UNA. This example illustrates the Bayesian networks' capability of handling logic events as well as uncertainties.

6.3 GeNIe

The Bayesian network of the AFCS is implemented in the software tool GeNIe. The topology and CPTs are created as described in the previous sections of this chapter. There are two ways of viewing a node in GeNIe, either as an icon or as a bar chart. The icon view shows the node name and a small box which indicates whether the inference is made successful. The icon view gives a nice overlook over the network which is advantageous when performing sensitivity analysis since GeNIe will flash sensitive nodes with a red light.

The bar chart view requires more space since it also shows the posterior probabilities of the nodes possible values. The bar chart view is preferred when performing diagnosis since the probabilities are in direct view. Since the bar charts require a larger space, a submodel view of the Bayesian network is in order for smooth navigation.

Figure 6.3 shows a screenshot of the Bayesian network of the AFCS with the nodes viewed as icons.

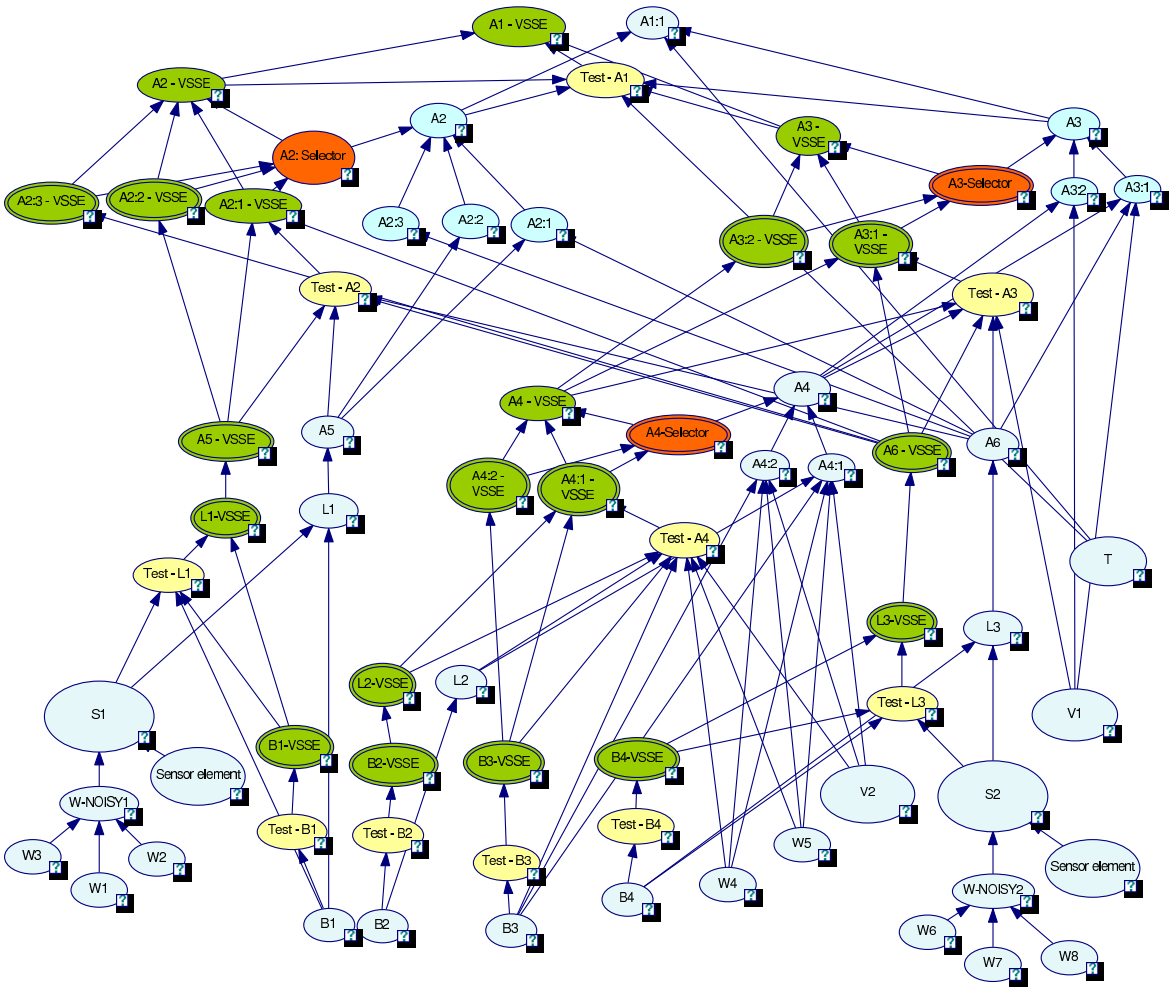

Figure 6.3: Bayesian network of AFCS with nodes in icon view.

Chapter 7

Model validation

For a Bayesian network used in troubleshooting, the diagnosis results constitute of an ordered list of probable causes ranked by posterior probabilities. Ideally the recommended diagnosis agrees with the diagnosis given by domain experts or data record.

In this chapter the Bayesian network model of the AFCS is evaluated. The evaluation is based on whether the top ranked posterior probability of the faulty components match the implemented fault. A small investigation is also made to see how the network performs in a more realistic large-scale model in terms of inference speed.

7.1 Results of troubleshooting

Every single fault described in section 3.2 is activated one by one and a simulation in Simulink is performed. For every simulation the observable test results within the ECU are saved and set as evidence in the Bayesian network. Table 7.1 shows the diagnostic test results for single faults excluding S2 = BIAS and V1 = HIGH FRICTION whose test results are presented individually. S1 = BIAS is varied from an underestimation of 50% to an overestimation of 50%, resulting in the same test results. The test result of 0 means that the test passed while 1 means that the test failed.

Table 7.2 shows the troubleshooting results for single faults excluding S2 = BIAS and V1 = HIGH FRICTION whose results are presented individually. The left column shows the activated fault. The right column shows the top ranked posterior probabilities of possible failure modes of components given by the Bayesian network with a threshold value of 0.05.

For faults like W1 = OC the inhibition probability in the noisy-or model of T_{L1} is set to zero indicating that the test will fail with probability one. Since W1 = OC, W2 = OC and W3 = OC all give the same test results, the posterior probabilities are given by 0.33. This result shows that the Bayesian network has a capability of handling faults of logic nature as well as uncertain events. The inference for these type of faults can be improved by either extending T_{L1}, for example making it a high-low in-range test, or, if reliabililty data is availabe, estimating better prior probabilities indicating if a fault is more probable to cause the test to fail.

	T_{A1}	T_{A2}	T_{A3}	T_{A4}	T_{B1}	T_{B2}	T_{B3}	T_{B4}	T_{L1}	T_{L3}
B1 = MALF	0	1	0	0	1	0	0	0	1	0
B2 = MALF	0	0	0	1	0	1	0	0	0	0
B3 = MALF	1	0	1	1	0	0	1	0	0	0
B4 = MALF	0	1	1	0	0	0	0	1	0	1
W1 = OC	0	1	0	0	0	0	0	0	1	0
W2 = OC	0	1	0	0	0	0	0	0	1	0
W3 = OC	0	1	0	0	0	0	0	0	1	0
T = LEAK	1	0	0	0	0	0	0	0	0	0
W4 = SCTG	0	0	0	1	0	0	0	0	0	0
W4 = OC	1	0	1	1	0	0	0	0	0	0
V1 = STUCK	1	0	1	0	0	0	0	0	0	0
W5 = OC	1	0	1	1	0	0	0	0	0	0
W6 = OC	0	1	1	0	0	0	0	0	0	1
W7 = OC	0	1	1	0	0	0	0	0	0	1
W8 = OC	0	1	1	0	0	0	0	0	0	1
S1 = BIAS	0	1	0	0	0	0	0	0	0	0

Table 7.1: Test results for single faults excluding S2 = BIAS and V1 = HIGH FRICTION. 0 = passed, 1 = failed.

The posterior probability of S1 = BIAS is a little bit higher than S2 = BIAS since there is a small probability that S2 = BIAS can set off T_{A3}.

Table 7.3 shows the unique diagnostic test results for S2 = BIAS which is varied from an underestimation of 50% to an overestimation of 50%. Table 7.4 shows the troubleshooting results for S2 = BIAS. The left column shows the size of the bias error. The right column shows the top ranked posterior probabilities of possible failure modes of components given by the Bayesian network with a threshold value of 0.05.

The result for -30% shows the strong influence of T_{A3} on the inference, indicating a much larger posterior probability of S2 = BIAS compared with S1 = BIAS. The same test influence the inference to set a smaller posterior probability of S2 = BIAS compared with S1 = BIAS for the case of -10%. A higher resolution of BIAS, for example BIAS_LOW and BIAS_HIGH, could be used to resolve this issue.

Figure 7.1 shows a screenshot from GeNIe with the inference result obtained from test values of the case S2 = BIAS : -30%. The nodes shown are in the submodel of GSA and are in bar chart view.

However when building a network it is always a trade-off between the desire of a large and rich model to achieve accurate results, on the one hand, and the cost of construction, maintenance and complexity of probabilistic inference on the other hand. This is especially the case when eliciting the probabilities with expert knowledge which can be an expensive and problematic resource. In practice building a Bayesian network is a process that iterates over

Fault	P(Component = Failure mode|T = Test results) $\geq$ 0.05
B1 = MALF	P(B1 = MALF | T)=1.0
B2 = MALF	P(B2 = MALF | T)=1.0
B3 = MALF	P(B3 = MALF | T)=1.0
B4 = MALF	P(B3 = MALF | T)=1.0
W2 = OC	P(W1 = OC | T)=0.33 P(W2 = OC | T)=0.33 P(W3 = OC | T)=0.33
W3 = OC	same as for W2 = OC
W1 = OC	same as for W2 = OC
T = LEAK	P(T = LEAK | T)=0.88 P(V1 = HIGH FRICTION | T)=0.12
W4 = SCTG	P(W4 = SCTG | T)=1.0
W4 = OC	P(W4 = OC | T)=0.5 P(W5 = OC | T)=0.5
V1 = STUCK	P(V1 = STUCK | T)=0.64 P(V1 = HIGH FRICTION | T)=0.36
W5 = OC	same as for W4 = OC
W8 = OC	P(W8 = OC | T)=0.33 P(W7 = OC | T)=0.33 P(W6 = OC | T)=0.33
W7 = OC	same as for W8 = OC
W6 = OC	same as for W8 = OC
S1 = BIAS	P(S1 = BIAS | T)=0.52 P(S2 = BIAS | T)=0.49

Table 7.2: Troubleshooting results for single faults excluding S2 = BIAS and V1 = HIGH FRICTION.

	T_{A1}	T_{A2}	T_{A3}	T_{A4}	T_{B1}	T_{B2}	T_{B3}	T_{B4}	T_{L1}	T_{L3}
-30%	0	1	1	0	0	0	0	0	0	0
-10%	0	1	0	0	0	0	0	0	0	0

Table 7.3: Test results for S2 = BIAS. 0 = passed, 1 = failed.

these tasks until a network that is deemed by the customer of the network to give satisfying results is obtained.

Table 7.5 shows the unique diagnostic test results for V1 = HIGH FRICTION which is varied from a value of 1 to 14 (nominal = 0.8). Table 7.6 shows the troubleshooting results for V1 = HIGH FRICTION. The left column shows the size of the friction of the throttle valve. The right column shows the top ranked posterior probabilities of possible failure modes

	P(Component = Failure mode|T = Test results) ≥ 0.05
-30%	P(S2 = BIAS | T)=0.86 P(V1 = HIGH FRICTION | T)=0.28 P(S1 = BIAS | T)=0.15
-10%	P(S1 = BIAS | T)=0.52 P(S2 = BIAS | T)=0.49

Table 7.4: Troubleshooting results for S2 = BIAS.

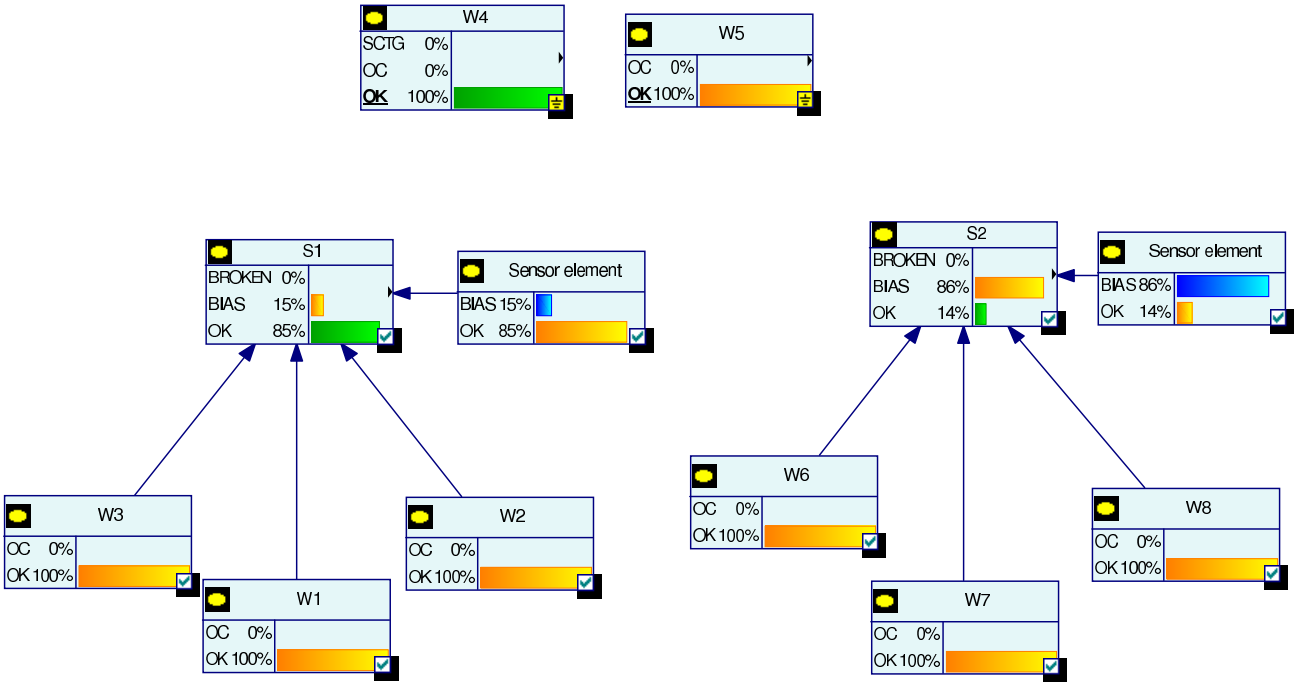

Figure 7.1: Screenshot from GeNIE of posterior probabilites obtained from test results for the case S2 = BIAS : -30%

	T_{A1}	T_{A2}	T_{A3}	T_{A4}	T_{B1}	T_{B2}	T_{B3}	T_{B4}	T_{L1}	T_{L3}
6	0	0	1	0	0	0	0	0	0	0
14	1	0	1	0	0	0	0	0	0	0

Table 7.5: Test results for V1 = HIGH FRICTION. 0 = passed, 1 = failed.

of components given by the Bayesian network with a threshold value of 0.05.

The result for a friction value of 14 shows that the inference prefers V1 = STUCK. However this is expected since V1 = STUCK can be seen as a special case of V1 = HIGH FRICTION for large sizes of the friction coefficient. If there is a wish to distinguish these faults then a

	P(Component = Failure mode|T = Test results) ≥ 0.05
6	P(V1 = HIGH FRICTION | T)=1.0
14	P(V1 = STUCK | T)=0.64 P(V1 = HIGH FRICTION | T)=0.36

Table 7.6: Troubleshooting results for V1 = HIGH FRICTION.

higher resolution of HIGH FRICTION can be employed.

7.2 Intractability of inference

The simplified ECU modeled in this thesis is about ten times smaller in comparison with a real one. Also a Scania truck consists of about 30 ECUs. In order to get a hunch of the intractability problem of inference, two, rather rudimentary, tests are performed. In the first test the Bayesian network of the AFCS is copied ten times and links are drawn between the submodels in GeNIe. Inference performed in this system with the clustering algorithm is experienced to be fast. In the second test the more realistic ECU, in terms of size, is copied ten times and links are drawn between the submodels in GeNIe. Inference perfomed in this system takes several seconds with the clustering algorithm. Remarkably, inference with the EPIS-sampling algorithm is not experienced to go faster.

Chapter 8

Conclusion

The results from Part I and Part II show that the new type of service oriented view together with simple tests makes the modeled AFCS to have promising fault-tolerant properties. Also, the obtained Bayesian network seems wellsuited for troubleshooting.

A higher resolution of service statuses and test results could improve the AFCS but increases the complexity and the risk of human design errors. A higher resolution of the failure modes and service statuses could obtain more accurate inference results for the Bayesian network, but is also suboptimal since it increases the complexity and thereby lowers the speed of probabilistic inference which for large-scale networks, for example the network of several ECUs, might become an issue.

Using Bayesian networks for diagnosis in this application has its advantages and disadvantages. One advantage is the direct correspondence of the network nodes with the real world components of the ECU. This facilitates the maintenance of the model since the addition/removal of a real world component corresponds to the addition/removal of corresponding network node(s). If a noisy-or model is employed for node relationships, a removal will not affect the relevant CPTs in GeNIe and also an addition of a node only requires the graphically aided specification of some inhibition probabilities.

The use of expert knowledge should not be underestimated but it may become disadvantageous when several experts are involved and a large number of probabilities are to be elicited.

The results of this thesis indicate that the service oriented view could be considered as a future architecture of ECUs and other systems with similar properties. Due to the limited time available for this thesis, the modeled ECU was kept simple in comparison with a more realistic model. Thus, the service oriented view should be given a more thorough investigation on a more realistic model of a real world ECU. Such an investigation would clarify questions regarding FTC, intractability of inference and the use of expert knowledge as the sole source of information.

Future work

Three suggestions of future work are presented in order to further evaluate the service oriented architecture.

Fault-tolerant control and diagnosis of more realistic system

The service oriented architecture should be applied to a more realistic system. An investigation could be performed in Scania to find a ready made Simulink model suited for this.

Intractability of inference

An investiagtion on how to deal with the intractability of inference should be done. This is an active area of research and different methods should be compared. For an example relevance-based algorithms, available in GeNIE, where only a small number of variables needs updating, could be of interest.

How to elicit many probabilities

An investiagtion on how to elicit many probabilities from different experts should be performed. This is an active area of research and different methods should be compared and evaluated.

Bibliography

[1] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, Diagnosis and Fault-Tolerant Control, 2nd Edition, Springer, 2006.

[2] G.F. Cooper, "The computational complexity of probabilisitc inference using bayesian belief networks", *Artificial Intelligence*, Volume 42, pp. 393-405, 1990.

[3] F. G. Cozman, "Axiomatizing Noisy-OR", *In Proceedings of the European conference on artificial intelligence*, Valencia, pp. 979-980, 2004.

[4] M. J. Druzdzel and L. C. van der Gaag, "Building Probabilistic Networks: Where Do the Numbers Come From? Guest Editors Introduction", *IEEE Transactions on Knowledge and Data Engineering*, Vol.12, No. 4, 2000.

[5] L.C. van der Gaag, S. Renooij, C.L.M. Witteman, B.M.P. Aleman and B.G. Taal, "How to Elicit Many Probabilities", *Proc. 15th Conf. Uncertainty in Artificial Intelligence*, pp. 647-654, 1999.

[6] A.-L. Gehin, M. Staroswiecki, "Reconfiguration Analysis Using Generic Component Models", *IEEE Transactions on systems, man, and cybernetics - Part A: Systems and humans*, Vol. 38, No 3, 2008.

[7] F. V. Jensen, T. D. Nielsen, Bayesian Networks and Decision Graphs, 2nd Edition, Springer, 2007.

[8] M. Nyberg, C. Svärd, "Diagnostic Modeling and Architecture of Large-Scale Fault-Tolerant Mechatronic Control Systems", *21st International Workshop on the Principles of Diagnosis*, Portland, Oregon, USA, 2010.

[9] S. Russell, P. Norvig, Artificial Intelligence A Modern Approach, 2nd Edition, Prentice Hall, 2003.

[10] H. Wang, "Using Sensitivity Analysis to Validate Bayesian Networks for Airplane Subsystem Diagnosis", *Aerospace Conference IEEE*, pp. 10, 2006.