Improving the availability of a DVB cable modem head-end system

Kwisthout, C.W.

Award date:
2002

Eindhoven University of Technology
Telecommunication Technology and Electromagnetics (TTE) Group
Electro-Optical Communication (ECO) Division

Improving the Availability of a DVB Cable Modem Head-end system

by C.W. Kwisthout

Report of the graduation project
carried out between February 2001 and December 2001,
commissioned by Prof. Ir. A.M.J. Koonen (TU/e)
and supervised by G.J.K.M. Koolen (BarcoNet Eindhoven B.V.) and Ir. H.P.A. van den Boom (TU/e).

Report of the graduation project carried out between February 2001 and December 2001, for the study Electrical Engineering of the Eindhoven University of Technology, Telecommunication Technology and Electromagnetics (TTE) Group, Electro-Optical Communication (ECO) Division, commissioned by Prof. Ir. A.M.J. Koonen (TU/e) and supervised by G.J.K.M. Koolen (BarcoNet Eindhoven B.V.) and Ir. H.P.A. van den Boom (TU/e).

SUMMARY

BarcoNet Eindhoven B.V. is a company that develops cable modem head-end systems that use the Community Antenna TeleVision (CATV) network to provide broadband access to users at home. Their head-end, the CableDock 200 is based upon the Digital Video Broadcasting (DVB) standard. This report describes a method to improve the availability of this CableDock 200. Availability can be defined as the time during which a user is able to use a system and its services with a certain service level. This method uses the user point of view for improving this availability. The main solution to achieve the improved availability is that multiple head-ends are clustered in a network. Every head-end is equipped with a so called 'redundancy system' that enables it to communicate with other head-ends. This communication is used to detect failing head-ends, to share information with each other and to take over cable modems from each other. This last feature requires the head-ends and the CATV network to be interconnected through one or more switches, capable of switching Hybrid Fiber Coax (HFC) signals.

The redundancy system consists of four functions:
- A function that detects if other devices are failing.
- A function that shares and collects information.
- A function that decides which device is the best candidate to take over a failing device.
- A function that carries out the actual take over action of a failing device.

These functions have very simple tasks making their implementation and thus the implementation of the redundancy system very simple. Together they form a system having the following features:
- The ability to detect a failing head-end within a certain (configurable) time range.
- A mechanism to decide which head-end is the best candidate to take over a failing head-end. This decision is based upon information of the network and its participating devices and upon the calculation of a quality factor per device using (configurable) weights.
- A mechanism to automatically take over a failing head-end.

Furthermore, a method is presented to use this redundancy system for failures within a single head-end as well. This added functionality provides the ability to automatically detect and take over failing boards within a single head-end.

The redundancy system can completely be programmed in software.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Scope of the graduation project

The final phase of the study Electrical Engineering at the Eindhoven University of Technology (TU/e) consists of a graduation project that combines the knowledge gained during the study with a practical or theoretical problem. After a successful trainee project at a company that develops broadband access systems for cable networks, I preferred to carry out my graduation project at a company as well. Since telecommunication is the area of research that interests me, I decided to carry out my graduation project for the Electro-Optical Communication (ECO) division of the Telecommunication Technology and Electromagnetics (TTE) group. The graduation project is commissioned by Prof. Ir. A.M.J. Koonen and supervised by Ir. H.P.A. van den Boom. At the same company as the company were I carried out the trainee project, BarcoNet Eindhoven B.V., they had some interesting research topics. Together with my supervisor there, G.J.K.M. Koolen, we defined an interesting graduation project: improving the availability of a DVB Cable Modem Head-end system.

One additional note has to be made; due to circumstances the research and development of the DVB cable modem head-end system was discontinued during this graduation project at BarcoNet Eindhoven B.V. However, the project is continued and finished as it originally was planned.

1.2 A DVB Cable Modem Head-end system

Nowadays, there is an ever-increasing demand for broadband (tele-) communication services. Especially broadband Internet communication is a hot item. There are several ways available to provide such a broadband Internet connection to a user. One of these ways is by using the Community Antenna TV network[1], or CATV network. These networks cover large (urban) areas in the world. Since CATV networks have enough bandwidth available to provide users with broadband access, they are very suitable for the job. Figure 1.1 shows a general configuration to use a CATV network for broadband access. Such a configuration is called a cable modem head-end system.

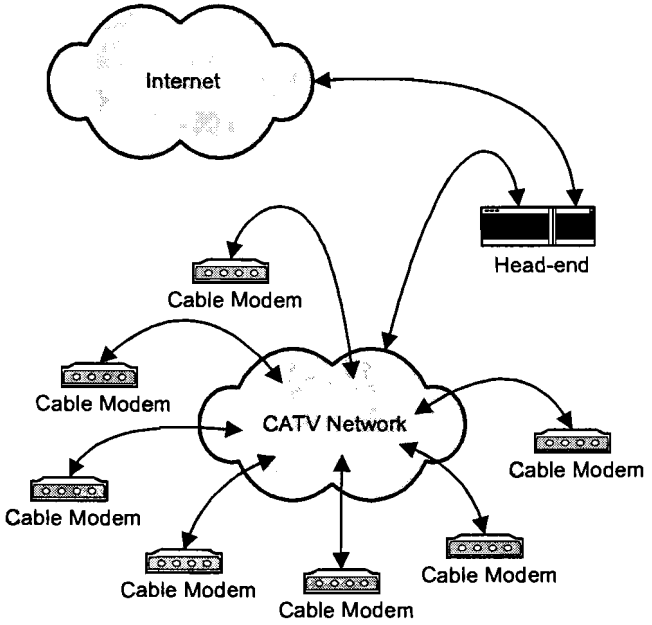

Figure 1.1. A cable modem head-end system.

[1] Also called Cable TV network.

We see that there are several users connected to the CATV network of the operator with the help of a cable modem. Furthermore, we that the CATV network of the operator is connected to the Internet with the help of a head-end. So the head-end is the bridge between the two networks: an Ethernet network (the Internet) and a Hybrid Fiber Coax (HFC) network (the CATV network).

Whereas on the Internet IP is a common standard to communicate, the CATV network has its own standards. Currently, two standards are defined: DOCSIS and DVB [1]. DOCSIS is the American standard and DVB is the European standard[1]. This graduation project deals with cable modem head-end systems based upon the DVB standard. However, most of the findings presented in this report, can also be applied to systems based upon the DOCSIS standard. A head-end based upon the DVB standard is called an Interactive Network Adapter (INA).

One of the most important requirements of a cable modem head-end system is its reliability. Reliability mainly defines the quality of such a system. One reason for this is that the operator wants satisfied users. Another reason for this is that the operator has to compete with other broadband access providers. For example, in case of broadband Internet the competition between CATV network operators and telephone-network operators. The operator may even want to compete with providers of services that do not demand broadband access. For example, in case of normal telephony services the competition between CATV network operators and the traditional telephony companies.

A part of reliability is availability. Availability can be defined as the percentage of the time during which a user is able to use a system and its services with a certain service level[2]. As can be seen in Figure 1.1, the head-end plays a crucial role in the availability of the total cable modem head-end system. The reason for this is that it is the interface between the Internet and the CATV network. Therefore, the availability of the Internet service of the CATV operator to the users, depends on the availability of the head-end. For BarcoNet Eindhoven B.V. the availability of their DVB head-end, the CableDock 200 [2], and ways to improve this availability, was an important research topic. This graduation project should give insight in the availability and ways to improve this availability of the CableDock 200.

1.3 Problem statement

After explaining the background of the graduation project in the previous section, a problem statement can be defined. In short the problem statement is defined as follows: how can the availability of the CableDock 200 be improved?

1.4 Project objectives

Based upon the problem statement of section 1.3, the project objectives can be defined. In section 1.2 the statement has been made that reliability and thus availability is a very important factor in the quality of a head-end. Availability itself is a very broad definition. Therefore, the first project objective is to refine the definition of the availability of a head-end. This also includes the refinement of the term 'improved availability' of a head-end. In other words, what should be the target availability of a head-end?

In order to get a clear insight in the availability of the CableDock 200, the second project objective is to create a (functional) model of the CableDock 200. This model should indicate the possible bottlenecks concerning the availability of a CableDock 200.

The third project objective is an investigation of possible ways to improve the availability of the CableDock 200. This part of the project should answer questions like 'Is it possible to work around

[1] The DOCSIS standard also has a version that is adjusted for the European market. This version is called EuroDOCSIS. Furthermore, the DOCSIS standard is in its 2nd phase: DOCSIS 1.1. The DVB standard is only in its 1st phase, but generally speaking this standard is equal to the DOCSIS 1.1 standard.
[2] See also Appendix A.

bottlenecks?' and 'Is it possible to decrease the impact of a bottle-neck?'. This part of the project should result in a list of possible solutions to improve the availability.

The most promising solution will be selected from the list of possible solutions. The fourth project objective is to describe (the basis of) an implementation of the selected solution.

Furthermore, the fifth project objective is to present an example using the implementation of the selection solution, that shows that indeed the availability of the CableDock 200 has improved.

The final project objective is to combine the findings in order to be able to draw conclusions concerning the availability of the CableDock 200 and to present recommendations.

1.5 Structure of this report

The structure of this report is based upon the project objectives as defined in section 1.4. Chapter 2 describes the refinement of the definition of availability and the investigation of the target availability. In chapter 3 three possible solutions to improve the availability of the CableDock 200, are presented. However, during the project it turned out that a combination of these three possible solutions would be far more promising than the individual solutions. The reason for this is presented at the end of chapter 3. The results of chapter 3 are used to define a requirements specification of this combined solution. This requirements specification is presented in chapter 4. Chapter 5 describes (the basis of) an implementation of this combined solution. In chapter 6 some examples of the working of this implementation are described. Finally, in chapter 7 the conclusions and recommendations are presented.

2 INVESTIGATION OF THE TARGET AVAILABILITY OF A HEAD-END

2.1 Introduction

In order to improve the availability of a head-end, one first has to define what the target availability is. Otherwise, one would easily conclude trivial things like 'higher is better'. This target availability also refines the definition of availability itself. This chapter describes an investigation of the target availability.

As stated in section 1.2, an operator can use the cable modem head-end system to compete with various services of various providers. The head-end is the part of the cable modem head-end system that has to provide the competing service in the first place. Therefore, the head-end competes with the systems of the various other providers. So in fact, the head-end competes with various levels of reliability and thus availability. An obvious choice for the target availability of the head-end would be the availability of the competing systems.

Another obvious choice for the target availability of the head-end would be the availability that is required by a (competing) service that is provided by the head-end.

Redundancy is an ever-returning subject when discussing availability. Redundancy can be described as the provision of multiple interchangeable components to perform a single function, in order to cope with failures and errors[1]. Therefore, redundancy is also used in the investigation.

Note that this background research has been carried out in the beginning of the project. Therefore, 'the current situation' refers to the situation as it existed in February and March 2001.

2.2 The government

First, the availability requirements concerning CATV networks, specified by the government were investigated. The question is whether the government has legislation that deals with the quality of CATV networks or with the quality of the services provided using those networks. After a profound investigation the conclusion can be drawn that there exists no such legislation. This also turned out to be the case with other communication networks, as for example the PSTN network[2]. The government only specifies a basis concerning the quality of the network and its services in its legislation. The actual quality is left to be regulated by the market and to be improved in time as a result of competition. In the Netherlands this basis is arranged in the Telecommunications Act [3] together with its directives and decrees. However, to avoid a miserable quality and to protect users, every country has a National Regulatory Authority (NRA) that has to watch over the market.

Before these organizations are discussed, one note has to be made. During this investigation it became clear that governments used to invest a lot of research in the opportunities of CATV networks. One of the most important reasons for this is that in most countries the PSTN communication market is (being) privatized. The monopoly of traditional telephony companies decreased as new telephony companies were founded. Since CATV networks cover large (urban) areas, CATV operators could easily become new competitors. This competition was thought to be good for the market. Therefore, many investigations were started of the possible ways to integrate the CATV networks with the PSTN network. However, the rapid development of the mobile phone market, made the traditional telephony services less interesting. Furthermore, the mobile phone market had an increasing number of competitors and the struggle new competitors had, made many CATV operators afraid that this struggle would await them too. Finally, this is a very complex market. For example, usually there are multiple CATV operators per country, so who is to blame in case something goes wrong. As a result,

[1] See also Appendix A.
[2] Public Switched Telephone Network; the traditional telephone network.

governments do not know how to cope with this situation. Therefore, there is no legislation and the opportunities of the CATV networks are hardly being used.

OPTA
OPTA ('Onafhankelijke Post en Telecommunicatie Autoriteit') is the NRA of the Netherlands. One of the tasks of OPTA is to define requirements concerning the quality of the telecommunication networks. It turned out that OPTA does not specify any requirements that deal with availability for CATV networks. This is partly due to the fact that the use of CATV networks for communication types other than the broadcast of television and radio signals, is a relative new market. Also, it is partly due to the fact that there are many CATV operators. This makes questions like 'Who is responsible in case of failures?' difficult to answer, compared to the PSTN market. Furthermore, OPTA relies on the market to regulate itself. For example, operators having a network with a miserable quality will simply not make it.

The Dutch case is not an individual case. At this moment it seems that none of NRA's in the world is going to define requirements that deal with availability. Most of the PSTN networks are already high quality networks and the CATV networks should follow this tendency in order to be competitive.

BORG
Another opportunity of CATV networks would be to use them in security systems. For example, a CATV operator could offer its users a security system with a central logging system. BORG is the Dutch certifying organization for security systems. BORG states that the quality of the communication networks used for the security systems is the responsibility of security system's owner. Again, one relies on the market to regulate itself. Therefore, there are no requirements concerning availability defined for this type of service.

Other organizations
Besides the OPTA and the BORG, more organizations (like ITU, ETSI and EU[1]) were investigated. Also, standards defined by those organizations were checked. However, none of them specified any requirements concerning availability of CATV networks or head-ends. See also references [3], [4], [5], [6], [7], [8] and [9].

2.3 The CableDock 200

From the CableDock 200 point of view, two requirements specifications exist: a commercial requirements specification [10] and a functional requirements specification [11]. These specifications were used as the basis for the design of the system [12]. Besides design requirements, the current performance has also been checked.

Commercial requirements specification of the CableDock 200
There are commercial requirements specified that deal with availability. At this time, operators do not demand a guaranteed minimum availability. They just compare the various systems available and choose the one that has the best mix of features. However, as stated in section 1.2, an operator wants satisfied users. This means that for an operator the most important view on availability, is the availability of the services as seen from the user.

Functional requirements specification of the CableDock 200
The CableDock 200 should have the following features that deal with availability:

- Hot Swap without losing connections and a maximum of 1 s channel loss. Hot Swap can be described as the possibility to remove, replace or add hardware without turning off the system. Currently, this feature is only implemented in hardware. To use it, the software also needs to support it.
- Hot Download with a maximum of 2 s downtime due to software updates. Hot Download can be described as the possibility to install or update software without resetting the system. This feature

[1] International Telecommunication Union, European Telecommunications Standards Institute and European Union.

is only partly implemented. Currently, the CableDock has to be reset in order to use new software and that means that all connections have to be set up again.
- A startup time of maximum 30 s (target is 15 s). This requirement is met.
- Support of watchdog programs. A watchdog program can be described as a piece of hardware or software that is triggered by a set of inputs. A failing input is recognized by the watchdog and the appropriate action is taken. The CableDock 200 should have various watchdog programs. Currently, most of them are implemented and active. However, none of them is allowed to take any action. For example, there is a watchdog program checks if the temperature of a board is too high. However, this is only reported; it does not switch off boards in order to cool down a board.

Current performance
In the beginning of this project, there was no long-term performance data available of the CableDock 200. Therefore, the long-term availability of the system is unknown and it cannot be used as a basis for an improved availability.

2.4 Standards and services supported by the CableDock 200

As stated in section 2.1, the target availability could be based upon the required availability of standards and services supported by a head-end. The CableDock 200 is a DVB head-end and currently, DVB is the only supported standard. However, this could be extended with support for EuroDOCSIS.

DVB
The DVB standard [2] does not specify any requirements that deal with availability. It only specifies a basis for some form of software redundancy.

EuroDOCSIS
The EuroDOCSIS standard also does not specify any requirements that deals with availability.

PacketCable 1.0
The PacketCable standard [13], [14] has been developed for providing packet based multimedia services over HFC systems using the DOCSIS protocol. This standard is being translated to use it with the DVB standard. Since voice functionality of the CableDock 200 will be based upon this standard, it has also been checked for availability requirements. Amongst other things, this standard specifies the following requirements:
- The probability of blocking a call has to be less than 1% during the High Day Busy Hour.
- Call cut-offs and call defects have to be less than 1 per 10,000 completed calls.

The first requirement has more to do with resources (bandwidth) than with availability. However, the second requirement deals vaguely with long-term system availability. This is an easy to meet requirement. It should be noticed that PacketCable 1.0 is a vague defined standard, probably because it is very new.

VoIP
Voice over IP (VoIP) is a protocol that can be used for telephony on an IP based network. The VoIP protocol does not require any guaranteed availability.

2.5 Competitors

Competitors of systems to use the CATV network for broadband access can be split into two groups: the ones that develop DVB based systems and the ones that develop (Euro-) DOCSIS based systems. The former are direct competitors whereas the latter are only competitors when the CATV operator has not decided which standard he is going to use. Availability features of a head-end could affect the operator's decision.

DVB competitors
Currently BarcoNet Eindhoven B.V. is the only company with a serious research and development department for DVB based systems. Therefore, future competing DVB based systems can be skipped for now. Already developed systems do not support any form of redundancy (which could result in an improved availability) except for a spare power supply.

(Euro-) DOCSIS competitors
Most of the (Euro-) DOCSIS based systems currently available, are being developed by American companies and they tend to exaggerate the features of their systems. However, (Euro-) DOCSIS based systems are somewhat further with redundancy than their DVB counterparts. Most of the systems support N+1 solutions[1]. The operator is free to choose between extra capacity (use the spare system as a normal system) and extra redundancy (keep the unit as a spare unit). These N+1 solutions do not have a mechanism to keep the connections alive when switching them from an operational unit to the spare unit. Also, it is unknown if these systems can switch over connections without human interference. Finally, no guarantees are specified concerning the availability of the systems.

2.6 Conclusions concerning the target availability

The purpose of the investigation was to define the target availability of a head-end. Whereas an availability of '5 nines' or 99.999% is a very common feature in the PSTN market, this is not the case when dealing with CATV networks. The high quality of the PSTN market is mainly the result of a self-regulating market. Using CATV networks for other services than the broadcast of television and radio signals is a relative new market. Therefore, this market has not been able to regulate itself. This manifests itself in for example, the lack of legislation and the fact that standards are still being developed. It seems that only short-term system performance is the main research topic. Availability is considered to be a future research topic. However, this does not mean that availability is not important at this moment. Besides an excellent commercial argument, it still remains a topic that needs attention in order to be competitive in the future. Investigation now will lead to results later.

In section 2.1 two ways were presented to define the target availability. The first way was to base it upon the availability of competing systems. The second way was to base it upon the availability required by the provided services. This latter way did not result in any requirements concerning the availability. The former way also did not result in any requirements. It only resulted in one practical thing to base the target availability on; the availability that has been achieved in the PSTN market. However, since telephony services have become less interesting and since this PSTN market is not a new market, this is perhaps not the best thing to base the target availability on.

The only thing that remains to base the target availability on, is that it should be higher than the current availability. Although this is a trivial definition, this results in the following targets:

- The availability of a head-end should in the first place be improved from the user point of view and it should be as high as possible.
- The improved availability has to be realized in a simple way. Since there are no demands for a certain level of availability, it should certainly not cost the operator extra money. The realization also needs to be simple in order not to create complex (and thus difficult to test) adjustments to the current CableDock 200. Therefore, the simplicity of the realization should be as simple as possible.

Note that these two requirements are in fact the opposite of each other. Improving the availability to a very high level will result in a complex realization and vice versa. Therefore, the actual realization of the improved availability will be a compromise between improvement and simplicity.

[1] This is a common form of redundancy. A N+1 solutions means that there is 1 spare (redundant) unit for N operational units.

Besides the target availability, the investigation did result in some side issues. These issues are used for improving the availability and return in the remaining of this report. They are:

- Redundancy is a good starting point. During the investigation it turned out that some systems already have a higher availability due to the support of redundancy. Also, redundancy gives the operator the choice between increased availability and increased capacity.
- Graceful degradation is desirable. Graceful degradation can be described as the degraded level of service that occurs when parts (not all) of a system are not functioning in the normal way[1]. When using this feature, a system can still be available in case of a failure.
- Load balancing can be an additional feature. Redundancy implies that one is able to switch tasks from operational hardware to redundant hardware in case of failures. If this is possible, it may also be possible to switch tasks in case of high loads.

The findings of the investigation of the target availability of a head-end as presented in this chapter, are used to describe some possible solutions for improving the availability in the next chapter.

[1] See also Appendix A.

3 POSSIBLE SOLUTIONS

3.1 Introduction

As stated in section 2.6 the availability of the DVB cable modem head-end system has to be improved from the user point of view. Therefore, it is necessary to present the CATV network from the user point of view to point out the various 'systems' involved. This user point of view is presented in Figure 3.1. This figure is based upon Figure 1.1, but it gives a more detailed view.

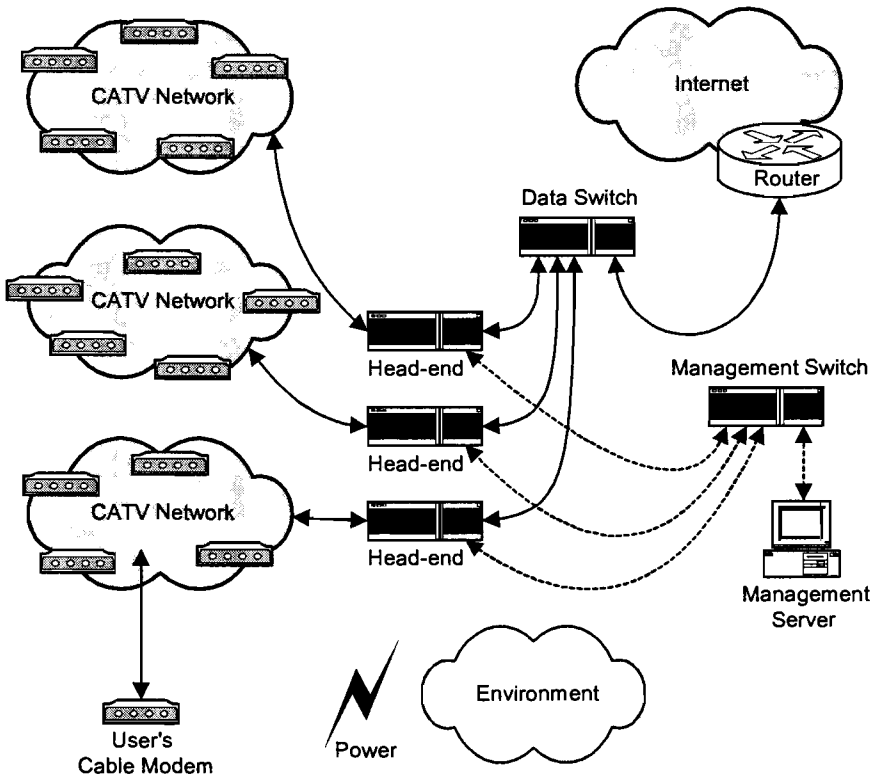

Figure 3.1. CATV network from a user point of view.

Figure 3.1 shows a typical configuration. In this figure normal lines represent payload data and dashed lines represent management data. An operator has several head-ends (in this case 3) to serve several physical areas of the CATV network. Those areas can be seen as separate CATV networks (in this case also 3). The head-ends are connected to the Internet through a switch and a router. The router is necessary since most of the available head-ends do not have Ethernet routing functionality built-in. Also, the head-ends are connected to the management server through a switch.

What 'systems' are involved when going from the user to the Internet? First of all, the user can have several equipment connected to the cable modem[1]. Therefore, the 1st system is: *the user's cable modem*. This modem is connected to the CATV network[2]. Thus, the 2nd system is: *the CATV network*. This CATV network is connected to the 3rd system: *the head-end*. Finally, this head-end is connected to the Internet through the 4th and 5th system: *the data switch* and *the router*. In order to be able to operate the head-end also has to interact with the management server. This leads to the 6th and 7th

[1] Most of the available cable modems only have an Ethernet connector. However, besides a computer also a telephone set or a decoder for digital television could be connected to this cable modem.
[2] Although the CATV network is represented as one 'system', in practice it consists of multiple HFC networks that are interconnected by various equipment, e.g. amplifiers and HFC splitters.

system: *the management switch* and *the management server*. Besides the systems just mentioned, two more 'systems' are involved: *the power* and *the environment*. The former 'system' provides the power for all systems involved. The latter 'system' is the collection of natural influences, e.g. lightning, Electro Magnetic Compatibility (EMC), floods.

Together, these 'systems' and their individual availability define the availability of the DVB cable modem head-end from the user point of view. The most interesting parts of this total system are the cable modem and the head-end. The reason for this is that these systems are developed by BarcoNet Eindhoven. In general, a failure of a cable modem only affects one user whereas a failure of a head-end affects all users connected to that head-end. Therefore, the focus lies on improving the availability of the head-end. In the following sections, three solutions to improve the availability of the CableDock 200 are discussed[1]. The first solution is to improve the availability of a head-end on module level. The second solution is to improve it on board level and the third solution is to improve it on head-end level.

3.2 Improving the availability on module level

3.2.1 Introduction

As stated in section 2.6 the solution for the improved availability should not cost an operator much more. Therefore, we want to avoid the use of additional systems. Therefore, it seems logically to improve the availability of a single head-end because this will not change the configuration as presented in Figure 3.1. In the beginning of the project, this solution was thought to be the most promising one. Therefore, a lot of effort has been put into this solution and it is treated likewise.

First the head-end has been decomposed into main building blocks. These main building blocks were further decomposed into modules. With the help of these modules and the way they are interconnected bottlenecks and Single Points of Failure (SPF)[2], should be pointed out. The solution to improve the availability would include ways to work around these SPF's, for example by adding redundant modules. This results in main building blocks with an improved availability and consequently, a head-end with an improved availability

3.2.2 Module decomposition

The main building blocks of the CableDock 200 are in fact the different boards that are present in the head-end together with the backbone that connects these boards. This decomposition is illustrated in Figure 3.2.

The system backbone is the CompactPCI (CPCI) bus. Several common units are connected to this CPCI bus, like a power supply, a fan unit and a signaling unit[3]. Furthermore, the units that deal with the actual data transport are connected to the CPCI bus: an US board (in this case 2), a DS board and the System Controller.

The US board handles the data from the user (the CATV network) to the System Controller (the Internet). Its input is an RF signal. Every US board has 2 US channels and thus two inputs. The DS board handles the data from the System Controller to the user. The output of the DS board is an IF signal. This signal has to be converted to an RF signal before it can be coupled into the CATV network. Therefore, the operator has to connect an up-converter to the output of the DS board. Note that this up-converter is not depicted in Figure 3.1. The System Controller handles the payload data from the US boards and DS board to the Internet and vice versa. It also handles the management data from the management server and vice versa. The System Controller plays a central role in the complete system.

[1] See also reference [15] for a discussion on designing high availability systems.
[2] See also Appendix A.
[3] For a detailed description of these units, see reference [2].

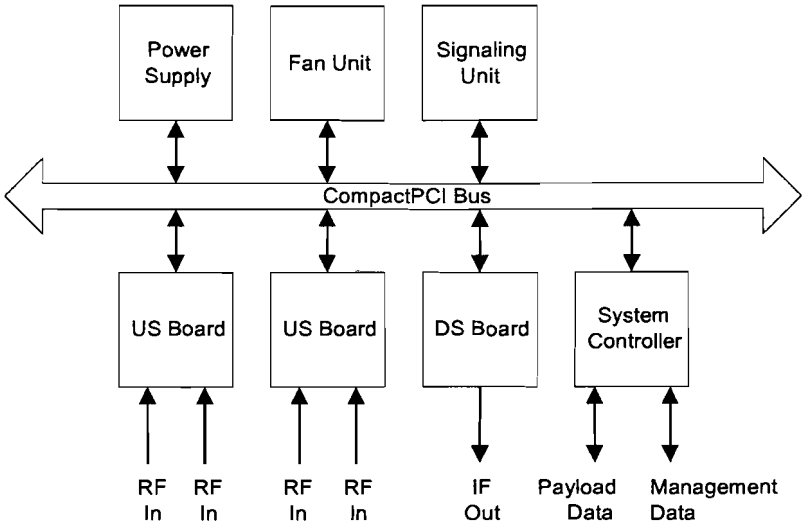

Figure 3.2. Decomposition of the CableDock 200 into its main building blocks.

These main building blocks were further decomposed into modules. The resulting block diagram for an US is depicted in Appendix B, Figure B.1. A detailed discussion of the different modules is beyond the scope of this project. As can be seen in the figure, there is a microcontroller present (Controller XPC8240). This microcontroller executes the software of the US board. Besides the hardware block diagram as presented in Figure B.1, a block diagram of the software can also be made. This block diagram is also present in the design documents of the US Board of a CableDock 200 [16].

3.2.3 Failure analysis

With the help of the block diagrams, the bottlenecks and SPF's can be pointed out. There are several ways to do this. One way is to translate the block diagrams into a computer simulation model. This simulation can be used to calculate reliability parameters like uptime, MTTF, MTBF and MTTR[1]. Recalculating these parameters while changing the characteristics of the different blocks, gives insight in the influence of the individual blocks on the total reliability and thus availability. However, this is a very complicated way because software interacts with hardware and it runs on the same hardware. Therefore, a good analysis would require a very complex model.

Another way to do the analysis is to use a Failure Mode and Effect Analysis (FMEA) [18]. Combined with a definition of the correct working of an US board, this would result in a good analysis of the situation too. The results of this analysis are summarized in Appendix B.

The most important conclusions that can be drawn after the FMEA procedure are:

- Most of the failures described in Table B.1 will almost never occur.
- Most of the failures have a very high impact on the correct working of the system. This means that most of them are SPF's.
- Most of the hardware failures can only be repaired by replacing the failing part. This affects all users connected to the failing piece of hardware.
- Most of the software failures can be repaired by restarting the software. This also affects all users provided with the help of the failing of software.

In order to avoid SPF's and thus to improve the availability on module level, redundant modules should be added. However, there are so many SPF's on a single US board that adding redundant modules would almost mean that a complete redundant board is added. This leads to the solution as presented in section 3.3. Since the DS board is similar to an US board, this would result in the same

[1] See also reference [17] and Appendix A.

thing. The System Controller is even more complex. Furthermore, it is not custom made so adding redundant modules would be at least very difficult. These reasons made that this approach to improve the availability of the system has not been finished.

3.2.4 A partial solution

In spite of the many SPF's, there is a way to partly improve the availability of DS and US boards. This is achieved by decreasing the time to discover a failure. Most of the software failures, the failures of the demodulator and the failures of the FPGA can be detected by the hardware watchdog that is present on the DS and US boards. The procedure to use this watchdog would be:

- The watchdog timer is programmed for a certain time interval. Within that time interval the watchdog should be triggered by a software process. If not, the watchdog automatically restarts the microcontroller of the board.
- The software process that triggers the watchdog has to be a very low priority process; it should have a low interrupt). This process only triggers the watchdog when all its inputs are set.
- The inputs of the software process are set by the software processes and hardware that should be checked. For example, when the FPGA is programmed in such way that it triggers one of the inputs of the software process every certain time interval (with an interrupt), a failing FPGA would result in an input that not has been triggered. Another example is a very small software process that calculates the checksum of the executable memory (the part where the software is present) of the SDRAM. If the checksum differs from the correct checksum this means that the software may be corrupt. Restarting the microprocessor result in a new download of the software.

The big disadvantage of this solution is that a board will be restarted in case of a failure. This implies that every time a failure occurs all connections are reset.

3.3 Improving the availability on board level

One of the findings of the previous section was that adding redundant boards to a single CableDock 200 would be the best way to avoid SPF's. This solution can be seen as improving the availability on board level. In case of the power supply unit this way of redundancy is already implemented [2]. In case of the other units it is somewhat more complex. For example, it is not possible to add a second System Controller to a CableDock 200. This means that the System Controller itself stays an SPF within a single head-end. However, there is a way to avoid this SPF; just add a complete redundant head-end. This solution is presented in the next section.

3.4 Improving the availability on head-end level

Adding complete redundant head-ends is the ultimate solution because every failure on one head-end can be solved by bringing the redundant head-end into duty. As stated in section 2.5, this solution is used by some of the (Euro-) DOCSIS competitors. However the big disadvantage of this solution is that all connections of the failing head-end have to be set up again on the redundant head-end. Also, an operator has to have at least one additional head-end so this increases the costs. This is the opposite of the conclusions drawn in section 2.6.

3.5 Selected solution

In order to improve the availability from the user point of view and do this according to the conclusions as presented in section 2.6, the solutions of section 3.3 and section 3.4 are combined into a new solution: the redundancy system. This redundancy system is described in the following chapters of this report. The partly solution as presented in section 3.2.4 also returns in the following chapters.

4 REQUIREMENTS SPECIFICATION OF THE REDUNDANCY SYSTEM

4.1 Introduction

In this chapter the conclusions of section 2.6 are translated into a requirements specification of the redundancy system. Because this redundancy system is based upon the findings of chapter 3, the requirements specification is also based upon these findings. The requirements specification as presented in this chapter can later on be used to check the final result. These requirements should then have become the features of the redundancy system. They should immediately point out why an operator should choose for this solution.

4.2 The situation

In order to describe the requirements specification of the redundancy system, the situation as presented in section 3.1 has to be further specified. When improving the availability, the use of extra equipment should be avoided. However, in order to be able to use redundant head-ends it should be possible to switch over users at the CATV side of the head-end. The resulting situation is presented in Figure 4.1. Note that the up-converters needed for the DS boards are not depicted.

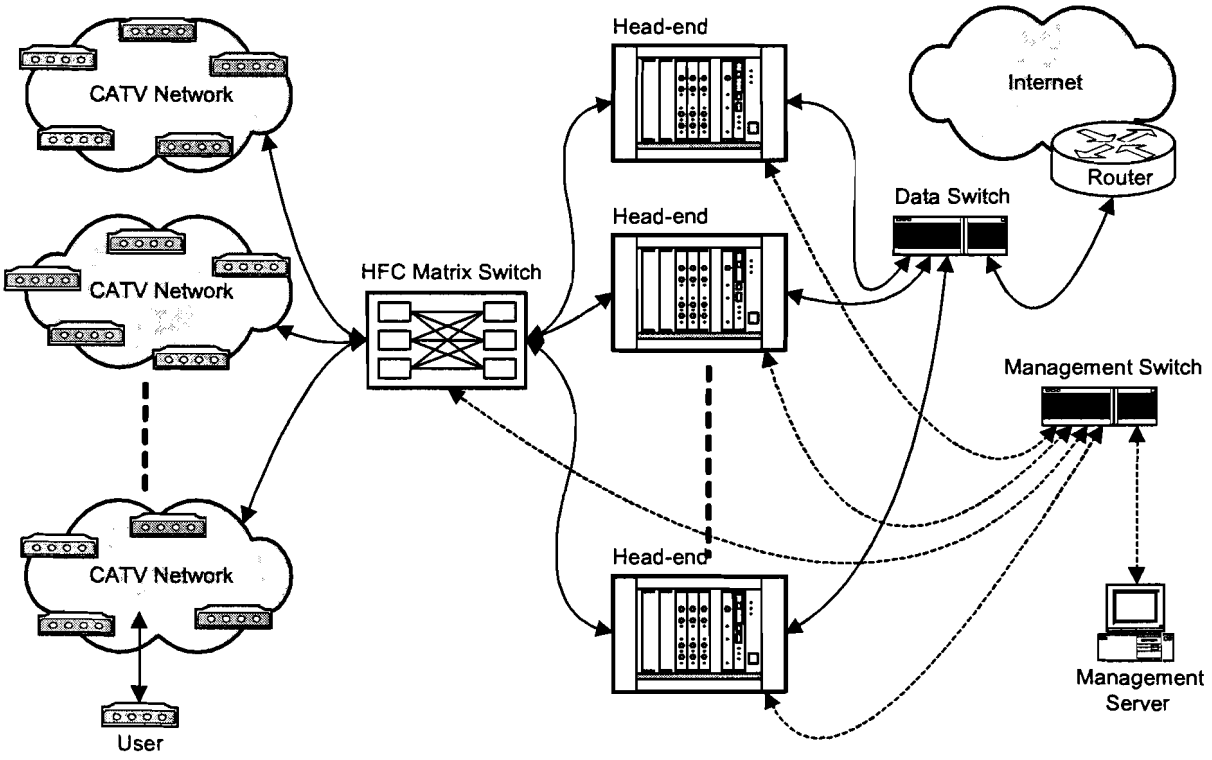

Figure 4.1. General situation.

In Figure 4.1 the normal lines represent the payload data and the dashed lines represent the management data. Compared with the situation as presented in Figure 3.1, a HFC matrix switch has been added to the configuration. This HFC matrix switch connects the CATV networks with the head-ends. It is capable of switching all of its inputs to all of its outputs. Note that although just one switch is depicted, an operator is free to use multiple switches. The switch has to support to be configured through a standard Ethernet interface. It is connected to the management network of the head-ends through this Ethernet interface. HFC matrix switches are commercially available in various

configurations. There are arrays available from 16x16 to as much as 256x256 inputs and outputs. Furthermore, they are capable of switching over inputs and outputs very fast (in the order of a few ns) so they are excellent for the job. Examples of these switched are the Q-Switch series switches of Quintech [19].

The Ethernet interfaces of the head-ends are interconnected with the help of switches. An operator is free to replace these switches by (cheaper) HUB's. The disadvantage of such a decision is of coarse that much more data collisions occur.

In the next section the requirements specification is presented using the situation as described in this section.

4.3 Main requirements

In section 2.6 the conclusion was drawn that the targets concerning the improved availability of a head-end, could be summarized with two main requirements. The first main requirement is:

- *The availability (thus uptime) from the user point of view should be as high as possible.*

This requirement results from commercial reasons. It is not a really quantified requirement. Therefore, the improvement of the availability with the redundancy system implemented compared with the availability without the redundancy system is used as check. Besides the user point of view, the improvement from the operator point of view is checked.

The second main requirement is:

- *The implementation of the redundancy system should be as simple as possible.*

This requirement results from development reasons. If the redundancy system would require very complex adjustments and additions to the current CableDock 200 and the total configuration, the chance that additional errors will be introduced is very well present. Furthermore, the development costs will be too high. The degree of simplicity is not a really quantified requirement, but this requirement is easily translated into more practical requirements as will be clear in the next section.

4.4 Requirements specification

With the first main requirement in mind, graceful degradation[1] would be an excellent feature. If users are still able to use the system in case of a failure (only with a degraded service level), this failure will not or only partly contribute to the downtime of the system from the user point of view. Only if the degradation were very serious, it would probably irritate the user. The first requirement is:

- *The redundancy system should provide a form of graceful degradation.*

From the user point of view, the complete system between the user's cable modem and the Internet should always be available. This is the ideal situation. The redundancy system is mainly based upon improving the availability on head-end level. This means that in case a head-end fails, it should be taken over by another (redundant) head-end. From now on, this form of redundancy will be referred to as inter head-end redundancy[2]. In chapter 3 it also turned out to be possible to improve the availability on board level. From now on this form of redundancy will be referred to as intra head-end redundancy[3]. What form of redundancy is best, depends on the situation. If a System Controller of a head-end fails, inter head-end redundancy is the only option. If a DS board of a head-end fails and there is another DS board present, intra head-end redundancy will be the best option. However, if no other DS board is present, inter head-end redundancy again is the best option.

[1] See also section 2.6 and Appendix A.
[2] See also Appendix A.
[3] See also Appendix A.

Both intra and inter head-end redundancy should be supported and the implementation of these two forms of redundancy should be as simple as possible. This implies that the redundancy system should work for both forms in the same way. Therefore, the second requirement is:

- *The redundancy system should support both intra and inter head-end redundancy using the same methods. This means that a head-end should be able to entirely take over another head-end. It also means that a part of a head-end should be able to entirely take over another (identical) part of that same head-end.*

As stated before, no new SPF's should be introduced when improving the availability with the help of the redundancy system. Therefore, a central master system should be avoided. This results in the following requirements:

- *The redundancy system should be a decentralized system.*
- *In case of inter head-end redundancy each head-end involved should be equal.*
- *In case of intra head-end redundancy each part involved should be equal.*

In this way we avoid a master-slave system. Instead, this results in a symmetrical distributed system.

From now on we will use the term device for a participating head-end in case of inter head-end redundancy and a participating board of a head-end in case of intra head-end redundancy.

The equality requirement leads to another requirement:

- *Since each device should be equal, each device should run the same software.*

The term software is used because the complete redundancy system will be implemented in software.

Another very important requirement is:

- *Since the complete system (black box) should never be down from the user point of view, a take over action should be performed very fast. Then, in worst case a user only notices a very short 'gap' in the connection.*

This 'gap' (downtime) of a user's connection consists mainly of the following time intervals:

1. Detection time: the time that has passed between the detection of a failure and the actual failure itself.
2. Decision time: the time that is needed to decide which device should take over the failing one.
3. Take over time: the time that is needed to actually take over the failing device.

This total time should be as short as possible and it should be predictable. This results in the following requirements:

- *The detection time should be fast and fixed. In this way it is guaranteed that a failing device is detected within a certain (predictable) time interval.*
- *The decision algorithm should be fast. This results in a short decision time.*
- *The take over time should be as short as possible.*

The last two (time) requirements both deal with parts of the redundancy system on a device that depend on information of the other devices: information needed to make the decision and information needed to actually take over connections of the failing device. This information should be provided (be available) as fast as possible. This leads to the following requirement:

- *All devices should have the same, most recent information of the others. This information consists both of information needed for the decision action and the information needed for the take over action.*

The previous requirements also imply that:

- *There should be a device that detects.*
- *There should be a device that decides.*
- *There should be a device that takes over.*

Of coarse it should be possible (and easy) to add and remove devices to and from the total configuration. This leads to the following requirements:
- *The redundancy system should be scalable.*
- *New devices should automatically be included for redundancy.*
- *It should be possible to excluded devices for redundancy in a proper way.*

Finally, an operator has always to be able to control the total network of redundancy systems. This leads to the final requirement:
- *The redundancy system of every device should be user configurable; an operator should be able to include and exclude devices from the automatic redundancy system.*

Now some more practical requirements are presented. In fact, these requirements are more possible ways to implement the previous defined requirements.

When looking at the functionality of the redundancy system, four separate functions can be distinguished:
- A function that should detect.
- A function that should decide in case a failing device has been detected.
- A function that should perform the take over action.
- A function that should deal with the shared information.

The information needed for the decision and the take over functions can be split into two parts:
- *Static information. This is information like the configuration of the HFC matrix switches.*
- *Dynamic information. This is information like the load of a device, or the number of connections a certain device serves.*

The redundancy system should have several states:
- *An initialization state. During this state the redundancy system includes itself for redundancy.*
- *A running state. This is the normal operation mode.*
- *A shutdown state. During this state the redundancy system excludes itself for redundancy.*
- *An alarm state. A device that detects a failing device enters this special state. Also a device that is going to take over a failing device enters this special state.*

Note that these states have nothing to do with the normal states of the CableDock 200.

Regarding the function (device) that detects, the following statements can be made:
- *Detection should be based upon a virtual ring network. This means that each device checks the following device in the virtual ring and then that device reports back.*
- *The time between two successive checks should be based upon the total number of participating devices. In this way the roundtime for this virtual ring network is fixed.*

As stated before, the information needed for the decision and for the take over functions, needs to be available at every participating device. This implies that a virtual ring network for this type of information is not possible. Therefore:
- *The sharing of the information should be based upon a multicast or broadcast network.*

No more (network) traffic than required should be generated, leading to:
- *The sharing of information should be based upon delta (changed) information.*

In the next chapter the redundancy system based upon the requirements specification as presented in this chapter, is described.

5 THE REDUNDANCY SYSTEM

5.1 Introduction

In the previous chapter the initial project requirements were translated into a requirements specification of the redundancy system. In this chapter this redundancy system is described. It is very important to keep in mind that there are two forms of redundancy: intra head-end redundancy and inter head-end redundancy[1]. Everything described for inter head-end redundancy always has an intra head-end redundancy counterpart. In the first sections of this chapter the redundancy system for inter head-end is discussed. In section 5.8 the then described inter head-end redundancy system is translated into the intra head-end redundancy system. Both systems together form the redundancy system.

First some definitions that deal with the redundancy system. These definitions are used throughout this and the following chapters.

Redundancy network
This is the total network of devices used for redundancy. For intra head-end redundancy these devices are the boards of a single head-end. They are interconnected through the CompactPCI bus[2]. For inter head-end redundancy these devices are the head-ends themselves. They are interconnected through the (Ethernet) management network.

Redundancy system
This is all software on a single device needed to participate in the redundancy network. Since there are two forms of redundancy, there are also two redundancy systems: an intra head-end redundancy system (the software on a single board) and an inter head-end redundancy system (the software on a single head-end). Note that the inter head-end redundancy system is an expansion of the intra head-end redundancy system running on the System Controller.

Redundancy states
These are the different states of the redundancy system. Note that these states have nothing to do with the normal states of a CableDock 200 [2].

Redundancy functions
These are the four different functions that perform the four main tasks of the redundancy system. Together these functions form the redundancy system.

Redundancy messages
These are the messages used by the redundancy systems of different devices to interact. In case of inter head-end redundancy these messages are IP (Ethernet) based. In case of intra head-end redundancy they are CompactPCI based.

5.2 Basis of the redundancy system

5.2.1 Possible views

Each redundancy system can be seen in two ways. One way is to divide it into four different states: initialization state, running state, shutdown state and alarm state. Another way is to divide it into four different (interacting) functions: detection function, information sharing function, take over decision function and take over activating function. Each of these function performs a task of the total redundancy system. First, the state view is summarized in section 5.2.2. The function view is

[1] See also Appendix A.
[2] See also Figure 3.2.

summarized in section 5.2.3. The four different functions are discussed in detail in the sections 5.3, 5.4, 5.5 and 5.6. Finally, in section 5.7 the state view will be discussed in detail.

5.2.2 State view

Figure 5.1 shows the four states of the redundancy system. Note that these states only refer to the state of the redundancy system of a device.

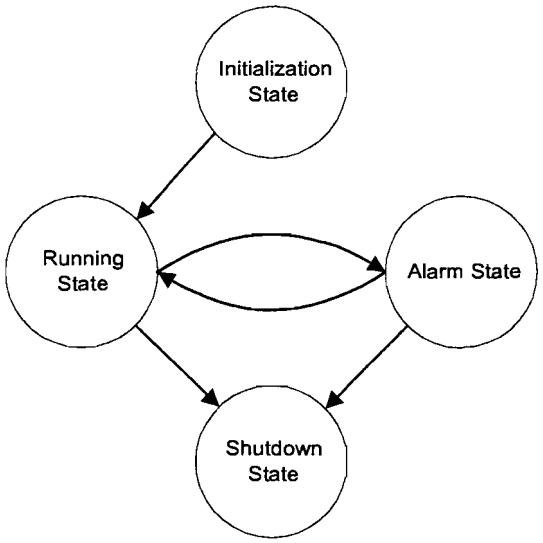

Figure 5.1. Main states of the redundancy system.

Initialization state
The redundancy system of a device starts in its initialization state. When the device itself has finished booting, the redundancy system (automatically) is started. During this state the redundancy system is initialized. This includes actions like:

- Collect information of other devices in the redundancy network.
- Request to be added to the detection ring.
- Share static information concerning the own device with the redundancy systems of other devices.

Running state
After the initialization state, the redundancy system enters the running state. The running state is the normal operation mode of the redundancy system. During this state the system participates in the detection ring. Furthermore, it shares and processes information needed for the take over detection function and the take over activating function.

Shutdown state
In order to remove a device from the redundancy network in a proper way, the shutdown state is present. During this state the device is removed from the detection ring. Furthermore, the other redundancy systems are notified to clear their local information of this device. In this way, a device can be removed properly, for example for maintenance purposes. Note that in order to re-add this device to the redundancy network, the redundancy system has to be initialized again.

Alarm state
In case something unusual occurs, the redundancy system enters the alarm state. Note that the tasks performed in the running state are continued in the alarm state. In fact, the running state and the alarm state can be seen as parallel states. The alarm state is entered for two reasons:

- The redundancy system of one device detects a possible failure of another device. The redundancy system of the device that detects now enters the alarm state and starts the take over

decision function. In case the device itself is the best candidate to take over the failing one, it stays in the alarm state and starts the take over activating function. Otherwise it just notifies the best candidate to take over. When finished, the redundancy system returns to the running state.

- The redundancy system of one device is forced to enter the alarm state by the redundancy system of another device. In this case the other device has detected a failing device and it has decided that this device is the best candidate to take over the failing one. When such a message is received, the redundancy system enters the alarm state and starts the take over activating function. When finished, the redundancy system returns to the running state.

In case of inter head-end redundancy there is one extra reason for entering the alarm state. This occurs when the intra head-end redundancy system cannot solve a failure. For example, it could be possible that a failing DS board is detected, but no other DS boards are present in the same head-end to take over this failing board. Therefore, this extra reason is:

- The intra head-end redundancy system forces the inter head-end redundancy system of the same head-end to enter its alarm state. This system starts the take over decision function. The difference with the first reason to enter the alarm state is that in this case the device to be taken over is (a part of) the own device. When the decision is made, the best candidate to take over is notified.

The just described procedure is actually the same as the procedure that would occur if the inter head-end redundancy system would have detected a possible failure of the own device. The only difference is that take over decision function cannot use the own device as a possible candidate.

After the notification of the best candidate, there are two possibilities: returning to the running state or entering the shutdown state. Note that the failure itself is not solved. Also note that the failing device has not been removed from the detection ring.

Returning to the running state would be the standard way. This implies that the redundancy system needs to share its new situation with the other redundancy systems. In this example this would concern the failing DS board. This is presented to the other devices as if this DS board is no longer present in this device. Entering the shutdown state means that the device will be removed from the redundancy network. In order to participate in the redundancy network again, the redundancy system has to be initialized again.

The choice between returning to the running state or entering the shutdown state depends on the severity of the failure and the configuration of the head-end. For example, if the failing part is a seriously failing DS board and it is only DS board present, the shutdown state would be the correct choice, for this head-end cannot serve a single user anymore. If the failure is not that serious and a simple reboot of just the failing DS board would solve the failure, the running state would probably be a better choice because now the redundancy system remains active.

Another example is when there are two DS boards present that operate on different DS frequencies and thus are not able to take over each other (in an ideal way). A failure of one of them could start this procedure, for the intra head-end redundancy system cannot solve the problem. Now returning to the running state is the best choice since there are still parts of the head-end functioning properly (the other DS board) so the head-end can still participate in the redundancy network.

5.2.3 Function view

Besides the state view as described in the previous section, another way to describe the redundancy system is the function view. This is perhaps a more convenient view because these functions represent actual pieces of software. Also, they can be modeled, programmed and tested separately. Together these functions form the redundancy system. Since every device has a redundancy system, every device has these four function.

Detection function
This part of the redundancy system takes care of the detection of failing devices. The detection functions use detection messages to interact. This function is mainly used in the running state but it is also used in the initialization state and the shutdown state.

Information sharing function
The task of this part of the redundancy system is twofold. One is that it is used for sharing information of the own device with the redundancy systems of other devices. Two is that it is used for collecting and processing information provided by the redundancy systems of other devices. The information sharing functions use information sharing messages to interact. This function also is mainly used in the running state but it is also used in the initialization state and the shutdown state.

Take over decision function
This function is only used in the alarm state. It takes care of deciding which device is the best candidate to take over a failing device. This decision is based upon the information provided by the information sharing function.

Take over activating function
This function is only used in the alarm state. It takes care of the actual take over of a failing device. This includes for example re-programming the HFC matrix switches and some post processing.

5.3 The detection function

5.3.1 Introduction

In this section the detection function is discussed in detail. First, the basis of this function is defined. Then, the way detection functions of different devices interact is presented together with a general example of how it works. In chapter 6 a more advanced example including the other functions is presented.

5.3.2 Basis of the detection function

The detection function is the part of the redundancy system that takes care of detecting failing devices. Since inter head-end redundancy is described first, these devices are head-ends. The input of the detection function is a detection message and so is the output of this function. As stated before, the interaction of the detection functions is based upon a virtual ring network. From now on this virtual ring network will be referred to as the detection ring. Figure 5.2 shows this detection ring and some participating detection functions.

As illustrated in Figure 5.2, the basis of this function is that the detection function of a device checks if the next device in the detection ring is still working properly by sending it an *alive* message. When a device receives such a message, it replies with an *ack_alive* message. The device that replies with the *ack_alive* message, now becomes the one to check the next device in the detection ring.

This automatic reply with an *ack_alive* message can be further refined. It is possible to only allow a device to reply in case it is absolutely sure it is working correctly. To be sure it works correctly, the device could use the help of (hardware) watchdogs[1].

[1] See also section 3.2.4.

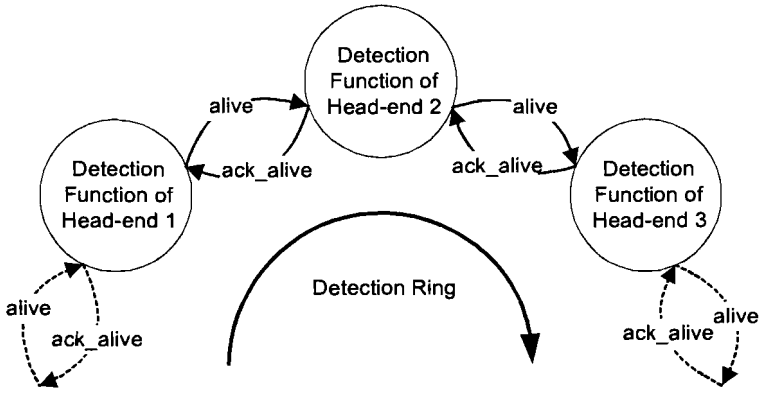

Figure 5.2. Basis of the detection function and its interaction.

The detection function in time

Figure 5.3 presents the basis principle of the detection function in time. It illustrates the interaction in the running state of the detection function of head-end 2 with the detection functions of head-ends 1 and 3.

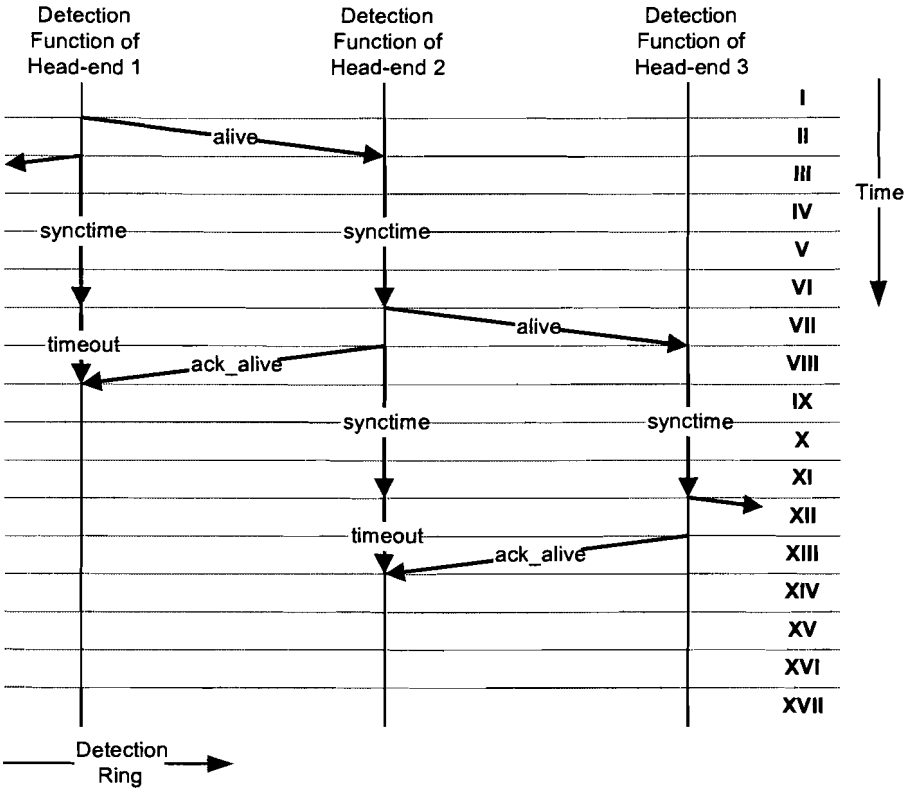

Figure 5.3. Flow chart (in time) of the detection function and its interaction.

Now focus on head-end 2. The detection cycle starts when its detection function receives an *alive* message from the detection function of head-end 1. First, it processes this message[1]. Then, it waits for the time interval *synctime*. After this time interval, it first sends an *alive* message to the next device in the detection ring, head-end 3, and immediately hereafter it replies to head-end 1 with an *ack_alive* message. In practice the time interval VII will be almost nil. Now it waits for the reply of head-end 3. The time it waits consists of the standard time interval *synctime* and the additional time interval

[1] This processing will be discussed in section 5.3.3.

timeout. If everything is alright with head-end 3, head-end 2 should receive the *ack_alive* message during time interval *timeout*. This ends the detection cycle for the detection function of head-end 2 for this round.

By first sending the *alive* message to the next device in the detection ring and then replying on the initial message, we are sure that the detection ring is always continued.

The time interval *synctime* is present for two reasons. The first reason is to keep the roundtime of the complete detection ring fixed. The second reason is to generate no more network traffic than necessary for the detection ring. The value of *synctime* thus depends on the target roundtime and on the number of devices that participate in the detection ring. In order to correct jitter on the roundtime, the value of *synctime* should be adjusted every round. The resulting parameter *synctime* is defined as:

$$synctime = \frac{max\_roundtime}{number\_of\_devices} \qquad (5.1)$$

By varying the value of *max_roundtime*, the value of *synctime* can be controlled (adjusted). This principle is explained in more detail in section 5.3.3.

The value of *timeout* is based upon practical information on the average time it takes to send a message from one device to another. Furthermore, it is based on the average time it takes to process a message. This time interval is present to remove standard network jitter. If this time interval was not present and an *ack_alive* message would arrive somewhat late due to high network traffic, the detection function would have already initiated the take over procedure. In practice this parameter will have a very low value. The exact way it is determined is explained in section 5.3.3.

The detection function in time in case of a failing device
After describing the ideal situation, now the situation in case a device is failing is presented.

The total time a device waits for the *ack_alive* message consists of the time intervals *synctime* and *timeout*. If it does not receive this message within this time interval, there might be a problem with next device in the detection ring. The detection function now forces the redundancy system to enter the alarm state.

As stated before, the detection function should continue its task during the alarm state. The first thing to do is the continuation of the detection ring. This is achieved by removing the failing device from the detection ring. The detection function then continues its cycle with the next device in that detection ring. This principle is illustrated in Figure 5.4.

In Figure 5.4 the dashed lines represent some of the detection messages dealing with head-end 2, as they would normally occur[1]. Normally head-end 1 would receive an *ack_alive* message from head-end 2 during the time interval *timeout*. In this case head-end 2 is down and therefore, head-end 1 does not receive a message. After this time interval the detection function of head-end 1 concludes that head-end 2 must have a problem. It forces the redundancy system to enter the alarm state.

Meanwhile, the detection function continues with removing the failing head-end 2 from the detection ring. This is done by removing head-end 2 from the detection message. Then, this detection message is reprocessed as input of the detection function of head-end 1. Since head-end 2 is removed, head-end 3 is now the next device in the detection ring. Therefore, the detection function continues by sending an *alive* message to head-end 3. Head-end 3 reacts as normal, so it first waits for the time interval *synctime* and then it replies with an *ack_alive* message. While the redundancy system of head-end 1 perhaps is still in the alarm state, this reply ends the detection cycle for head-end 1.

[1] Compare these messages with the situation as presented in Figure 5.3.

Note that in case head-end 3 is down too, the detection function of head-end 1 will queue this failing head-end as input for the take over decision function. It will continue with the next head-end in the ring.

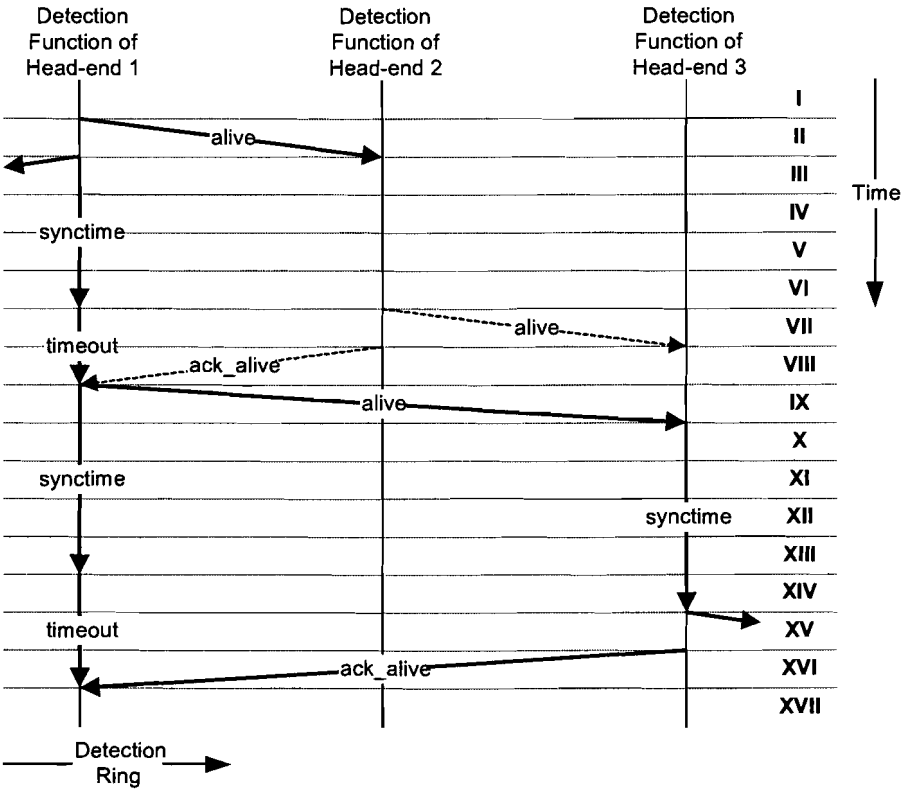

Figure 5.4. Flow chart (in time) of the detection function and its interaction in case head-end 2 is down.

Note that the second time head-end 1 performs the detection procedure, it does not first wait for the time interval *synctime* before sending the *alive* message to head-end 3 as it would normally do when an *alive* message is received. This difference is used to speed up the continuation of the detection ring. Also note that the second time it does not send an *ack_alive* message to the device preceding head-end 1 in the detection ring as it would also normally do prior to sending the *alive* message to the next device in the detection ring. This difference is present because it has already sent such a message the first time.

Furthermore, note that the second time value of *synctime* has changed because the number of participating devices is now one less. Also, note that head-end 3 receives the *alive* message the time intervals VIII + IX later than normal. Together with the changed time interval *synctime*, this means jitter on the roundtime of the detection ring. This is the reason for adjusting the value of *max _roundtime* every round.

5.3.3 The detection message

The detection functions of different devices use detection messages to interact. Since we are dealing with a detection ring (successive checks), we are dealing with unicast messages. The minimum information in a detection message, needed by the detection function is:

- IP addresses of the participating devices.
- Pointer to the next device to check in the detection ring.
- Maximum allowed roundtime (the parameter *max _roundtime*) in ms.

In order to make it easy to process the detection messages, the following fields are also included:
- Frame number. This is present for logging and debugging purposes.
- Total number of participating devices.
- IP address of the device that adjusts the value of *max_roundtime* every round. In case this device is failing, the device that detects this failure replaces this IP address with its own IP address.
- Local time of the device when the value of *max_roundtime* was adjusted the last time.
- Message type: *alive*, *ack_alive* or *alarm*.

Besides this information, the value of the time interval *timeout* also has to be stored. This is done locally at every device because this value only depends on the speed with which two detection functions are able to interact. Therefore, this value may be different for every device.

Every time the cycle for a detection function starts, it starts with the processing of the detection message. First it checks the message type. If it is not an *alive* message, something unusual has occurred. The detection function replies to the sender of the message with that same message, only with the message type set to *alarm*.

If the message type is *alive*, it checks if it is the device that should adjust the roundtime to correct for jitter. If so, it adjusts the value of *max_roundtime* (similar to other adaptive systems) as:

$$max\_roundtime = max\_roundtime + \frac{expected\_time - time\_of\_arrival}{2} \qquad (5.2)$$

In this formula the parameter *expected_time* is defined as:

$$expected\_time = local\_time(from\ message) + max\_roundtime(from\ message) \qquad (5.3)$$

This new value of *max_roundtime* is used to calculate the value of *synctime* according to formula 5.1. In case it was not the device that should adjust the roundtime, the value of *max_roundtime* from the detection message is used to calculate the value of *synctime*.

The new detection message is constructed with an increased frame number and pointer to point to the next device in the detection ring. In case the value of *max_roundtime* was adjusted, this new value is copied into the message, together with the new local time. The message type remains set to *alive*. After the generation of the message it waits for the time interval *synctime*.

Immediately after the new *alive* message has been sent, the detection system replies to the initial *alive* message. The initial message is used as reply, only with the message type set to *ack_alive*. At the same time it clocks the difference in time between the time the new *alive* message was sent to the next device in the detection ring and the time the reply with an *ack_alive* message was received from that device. This value is used to adjust the value of *timeout as*:

$$timeout = timeout + \frac{synctime - time\ between\ send\ and\ reply}{2} \qquad (5.4)$$

This new value of *timeout* is stored locally and it will be used the next round. This ends the cycle for this device.

5.3.4 Remarks regarding the detection function

The size of the detection message depends heavily on the number of devices that participate in the redundancy network. The reason for this is that all their IP addresses must be present in the message. A detection message is probably going to look like this:

- Message type 8 bit
- Frame number 32 bit
- IP address of the device that adjusts the value of *max_roundtime* 32 bit
- *max_roundtime* 16 bit
- *local_time* 32 bit
- *number_of_devices* 8 bit

- Pointer to the next device in the detection ring — 8 bit
- IP addresses of the participating devices — *number_of_devices* x 32 bit

Since the pointer and *number_of_devices* are only 8 bit, 255 different IP addresses are supported in the detection message. This restricts the total number of devices that can participate in a single redundancy network to 255. The minimum number of devices to create a detection ring is of coarse 2. Therefore the total detection message size can defined as:

$$136\,bit + 2 \cdot 32\,bit = 200\,bit \leq detection\,message\,size \leq 136\,bit + 255 \cdot 32\,bit = 8296\,bit \qquad (5.5)$$

Adding an overhead of 14 + 20 bytes (Ethernet and IP header) + 10% = 300 bit to these boundaries, the message to be sent over the Ethernet will vary between 500 bit and 8596 bit. On a simple 10 Mbit/s network, this implies that the time needed to send a detection message varies from 0,05 ms to 0,9 ms.

A normal (practical) value for the time needed for the processing of a detection will be in the order of few tens of ms. Although 255 devices are supported, a more normal configuration will probably include 5 to 10 head-ends that participate in the redundancy network. This means that a roundtime in the order of 1 s could be achieved and this also means that the time needed to send the detection messages can be neglected compared to the roundtime. This is illustrated in Figure 5.5.

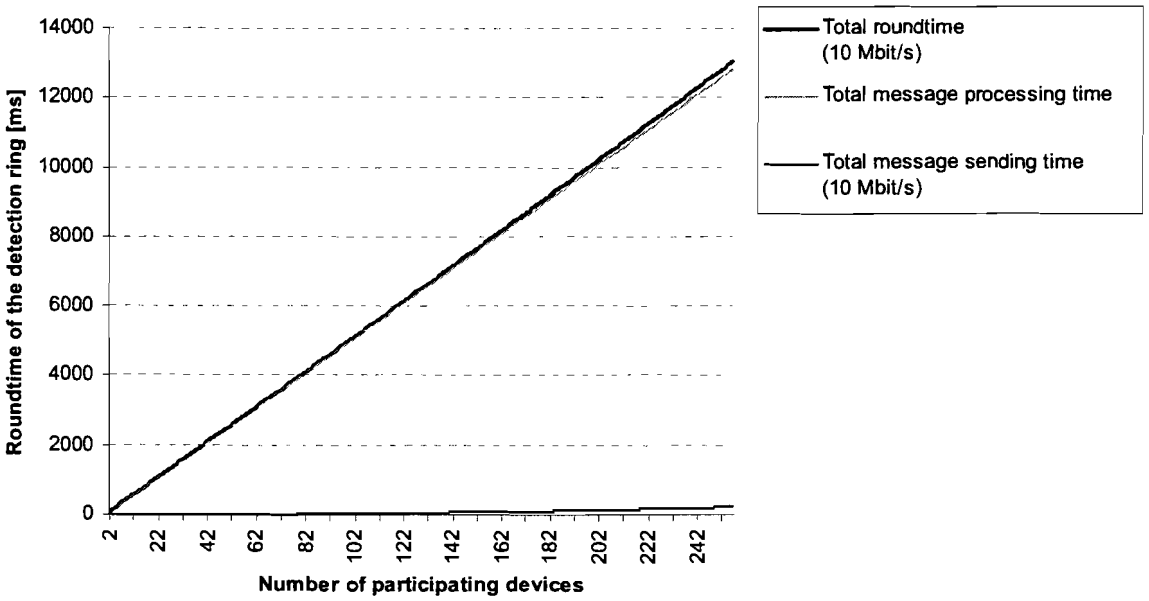

Figure 5.5. Total roundtime of the detection ring versus the number of participating devices.

In this example, 50 ms is used for the time per device it takes to process the incoming detection message. The total roundtime that could be reached has been calculated in case every device that receives a detection *alive* message processes this message and immediately hereafter sends the adjusted message to the next device in the detection ring. Therefore, the devices do not wait for the time interval *synctime* between receive and send. The total roundtime has been calculated in ms for both a 10 Mbit/s network and a 100 Mbit/s network, according to:

$$\left(50 + \frac{136 + 300 + number\_of\_devices \cdot 32}{\frac{network\_speed}{1000}}\right) \cdot number\_of\_devices \qquad (5.6)$$

In formula 5.6 the network speed is specified in Mbit/s (e.g. 10). Since the differences between the total roundtime with a 10 Mbit/s network and a 100 Mbit/s network turned out to be very small, only the results for a 10 Mbit/s network are showed, for this is the slowest network.

From Figure 5.5 the conclusion can be drawn that a total roundtime that is less than 1 s, the maximum number of devices that can participate in the redundancy network is about 20. As stated before 10 devices is a good practical value so this will not be a problem.

5.3.5 Example of the detection function

Suppose that the following head-ends participate in the detection ring: head-end 1 (IP address 192.138.0.24), head-end 2 (IP address 192.138.0.13) and head-end 3 (IP address 192.138.0.38). Furthermore, the target roundtime is 2 s. The value of *timeout* stored locally at head-end 2 is 10 ms.

At local time t = 6100, head-end 2 receives the following detection message (the source and destination IP addresses of the IP header are also showed):

- Source IP address: 192.138.0.24
- Destination IP address: 192.138.0.13
- Message type: *alive*
- Frame number: 316
- IP address of the device that adjusts the value of *max_roundtime*: 192.138.0.13
- *max_roundtime*: 2000
- *local_time*: 3900
- *number_of_devices*: 3
- Pointer to the next device in the detection ring: 2
- IP addresses of the participating devices: 192.138.0.24
  192.138.0.13
  192.138.0.38

The detection function starts with the processing of the message. First it checks if it should adjust the value of *max_roundtime* by comparing the appropriate field of the message with its own IP address. Since this is a match, it adjusts the value according to the formulas 5.2 and 5.3:

$$expected\_time = 3900 + 2000 = 5900$$

$$max\_roundtime = 2000 + \frac{5900 - 6100}{2} = 1900 \text{ ms}$$

This new value of *max_roundtime* is used to calculate the value of *synctime* according to formula 5.1:

$$synctime = \frac{1900}{3} = 633 \text{ ms}$$

At local time t = 6124 (thus the processing cost 24 ms) it constructs the new detection message:

- Source IP address: 192.138.0.13
- Destination IP address: 192.138.0.38
- Message type: *alive*
- Frame number: 317
- IP address of the device that adjusts the value of *max_roundtime*: 192.138.0.13
- *max_roundtime*: 1900
- *local_time*: 6124
- *number_of_devices*: 3
- Pointer to the next device in the detection ring: 3
- IP addresses of the participating devices: 192.138.0.24
  192.138.0.13
  192.138.0.38

It first waits for time interval *synctime* and thus for 633 ms, after which at local time t = 6757 it sends the new detection message to head-end 3. Immediately hereafter it sends the following detection message (almost the same as it received) to head-end 1:

- Source IP address: 192.138.0.13
- Destination IP address: 192.138.0.24
- Message type: *ack_alive*
- Frame number: 316
- IP address of the device that adjusts the value of *max_roundtime*: 192.138.0.13
- *max_roundtime*: 2000
- *local_time*: 3900
- *number_of_devices*: 3
- Pointer to the next device in the detection ring: 2
- IP addresses of the participating devices: 192.138.0.24
  192.138.0.13
  192.138.0.38

Suppose that head-end 3 replies at local time t = 7384 (of head-end 2) to the detection message it received, with an *ack_alive* message. The detection function of head-end 2 calculates the new value of *timeout* used for the next round, according to formula 5.4 as:

$$timeout = 10 + \frac{633 - (7384 - 6757)}{2} = 13\,\text{ms}$$

This value is stored and this ends the detection cycle for this round.

Now suppose that head-end 3 is down. After the *alive* message was sent, head-end 2 first waited for 633 + 10 = 634 ms but no *ack_alive* message was received during this time interval. The detection function reacts by forcing the redundancy system to enter the alarm state. Meanwhile, it continues the detection ring and the detection message is adjusted to the new situation:

- Source IP address: 192.138.0.13
- Destination IP address: 192.138.0.24
- Message type: *alive*
- Frame number: 317
- IP address of the device that adjusts the value of *max_roundtime*: 192.138.0.13
- *max_roundtime*: 1900
- *local_time*: 6124
- *number_of_devices*: 2
- Pointer to the next device in the detection ring: 1
- IP addresses of the participating devices: 192.138.0.24
  192.138.0.13

This message is now used as the new input of the function. It ignores the adjustment of the value of *max_roundtime* since this is not relevant this time[1]. Furthermore, it ignores waiting for the time interval *synctime*[2]. Almost immediately after the detection of the failure (at local time t = 7410), the adjusted message is sent to head-end 1.

At local time t = 8350 the *ack_alive* message is received from head-end 1. The detection function of head-end 2 finishes this cycle by calculating the new value of *timeout* according to formula 5.4:

$$timeout = 13 + \frac{\frac{1900}{2} - (8350 - 7410)}{2} = 18$$

This value is stored locally. Note that meanwhile, the redundancy system of head-end 2 could still be in the alarm state due to the failing head-end 3.

[1] The function is still in the same round.
[2] Note that the value of *synctime* still has to be calculated because this new value (based upon only 2 participating devices) defines the time head-end 2 waits for an *ack_alive* message of head-end 1.

5.4 The information sharing function

5.4.1 Introduction

Besides the detection function, the information sharing function is the main function used in the running state of the redundancy system. The information is an important function, for it has to provide the information needed by the take over decision function and the take over activating function. As concluded in section 4.4, this information needs to be present all the time; when a failure has been detected, the take over decision function and the take over activating function have to be able to use the information immediately.

This section starts with a discussion of the different quality levels for taking over a cable modem. These quality levels are used when describing the information sharing function itself. This description starts with a discussion of the actual information that should be shared[1]. Then, the basis of the information sharing function is described. Finally, the information sharing message is defined. This section is finished with an example of the information sharing function and its interaction. In chapter 6 a more detailed example including the other functions is presented.

5.4.2 Quality levels for taking over a cable modem

When taking over a cable modem, the user point of view is one of the main interests. When using this view, four different quality levels for taking over a cable modem can be defined:

- **Level 1**: The modem can be taken over in an ideal way. This means that there is no interruption in the connection. This also means that after the take over action, the service level the cable modem gets, is the same as it got before that take over action. In practice, this is only possible if there is a complete redundant device available and if all information needed for a smooth continuation of the service is available at that redundant device.
- **Level 2**: The modem can be taken over in an ideal way, only with a degraded service level. This means that there is no interruption in the connection, only this connection is provided with a degraded service level after the take over action. This only requires a device that is able to take over the modem; it does not need to be redundant. That device should have all information needed for the take over action.
- **Level 3**: The modem can be taken over. Now there will be a (very short) interruption in the connection and after the take over action the connection is provided with a degraded service level. This requires a device that can take over the modem. In worst case, the modem itself has to sign on again. At least it has to adjust some of its (connection) parameters after the take over action[2]. This only requires information to be sure that a modem can sign on again.
- **Level 4**: The modem cannot be taken over. Actually this is not really a quality level. This means that it is physically not possible to take over this particular modem.

In section 5.3.4 the conclusion was drawn that in case of a failing device a worst case detection time of about 1 s is possible. In addition to this worst case detection time there will be some time needed by the take over decision system and some time needed by the take over activating system. This means that the total time needed for a successful take over action will probably be in the order of a few seconds or even tens of seconds. For the cable modems currently available, such a worst case timeout of the connection will result in a reboot of the modem. Since the reboot results in a new sign on procedure of the modem, all information present at a device to take over a modem without such a sign on, is useless. As stated before no more network traffic should be generated than necessary and simplicity is one of the main design requirements. Therefore, the take over actions are based upon quality level 3. Note that the sign on procedure does not necessarily mean that the protocol that uses the connection will suffer from a timeout. For example an FTP connection can cope with timeouts of more than half a minute.

[1] The information sharing function has to provide the information for the take over decision function and the take over activating function. Therefore, the reasons why some of the information needs to be shared, will also be discussed in section 5.5 and 5.6.
[2] This causes the interruption in the connection of the cable modem.

5.4.3 Information that should be shared

The most important part of the information sharing function, is the correct definition of the information it is has to handle. On the one hand this depends on the information itself and on the other hand this depends on the way this information is represented in the information sharing message. In short this information consists of the status of the devices that participate in the redundancy network.

Naming the various signals that describe the status of a device correctly, is the basis of a good definition of the information. The name of a signal needs to be unique for the complete redundancy network; otherwise this could result in errors. For example, if a part of the CATV network does not have a unique name and it has to be taken over by a head-end that already serves another part of the CATV network that has the same name, this may result in a conflict and the take over action cannot be performed. Giving the signals a unique name is partly the responsibility of the operator and partly prescribed by the redundancy system.

The information itself should be divided into smaller parts to make it easier to work with. This could be done based upon the purpose of the information or the characteristics of the information. The first method leads to:

- Information needed by the take over decision function. This includes information like the physical configuration of the total network, the configuration of the different devices and the load of a device.
- Information needed by the take over activating function. This includes information like the connections that need to be taken over by a device and (again) the physical configuration of the total network. This latter reason is because the signals need to be actually switched.

This method leads to an overlap between the information. The second method leads to:

- Static information. This is information that will not change for a long period of time (months, perhaps even years). This includes information like the physical configuration of the total network and the configuration of the devices.
- Dynamic information. This is information that changes all the time. This includes information like the load of a device, the number of connections and the details of those connections.

This method does not lead to an overlap and therefore, this method is used. Note that the operator generates the static information when configuring the total network. Furthermore, the operator is always allowed adjust the static information in case he changes the configuration of the total cable modem head-end system. The redundancy system is also allowed to adjust parts of that static information, for example the state of the switches. The dynamic information is generated by the devices itself due to normal operation of those devices.

In section 4.2 the general situation has been described. In order to define all information, a more detailed view of that general situation is required. Figure 5.6 shows this view.

This view is a more accurate for two reasons. The first reason is that there are two HFC matrix switches present instead of one. In practice, this also could be the situation. The second reason is that now the different CATV networks are split into DS and US networks, or areas[1].

Figure 5.6 is used to describe the different types of information and to group them. Note that for convenience the up-converters between the DS boards and the physical cables to the DS areas are not depicted. Also, note that in practice the number of devices (e.g. DS and US boards, head-ends, switches) can be different. The names (numbers) present in the figure can be ignored for now; they will be used in the example as presented in section 5.4.7.

[1] In practice an operator also has to define DS and US areas. Generally speaking, one DS area contains multiple US areas. For a more detailed discussion of these areas, see references [1] and [2].

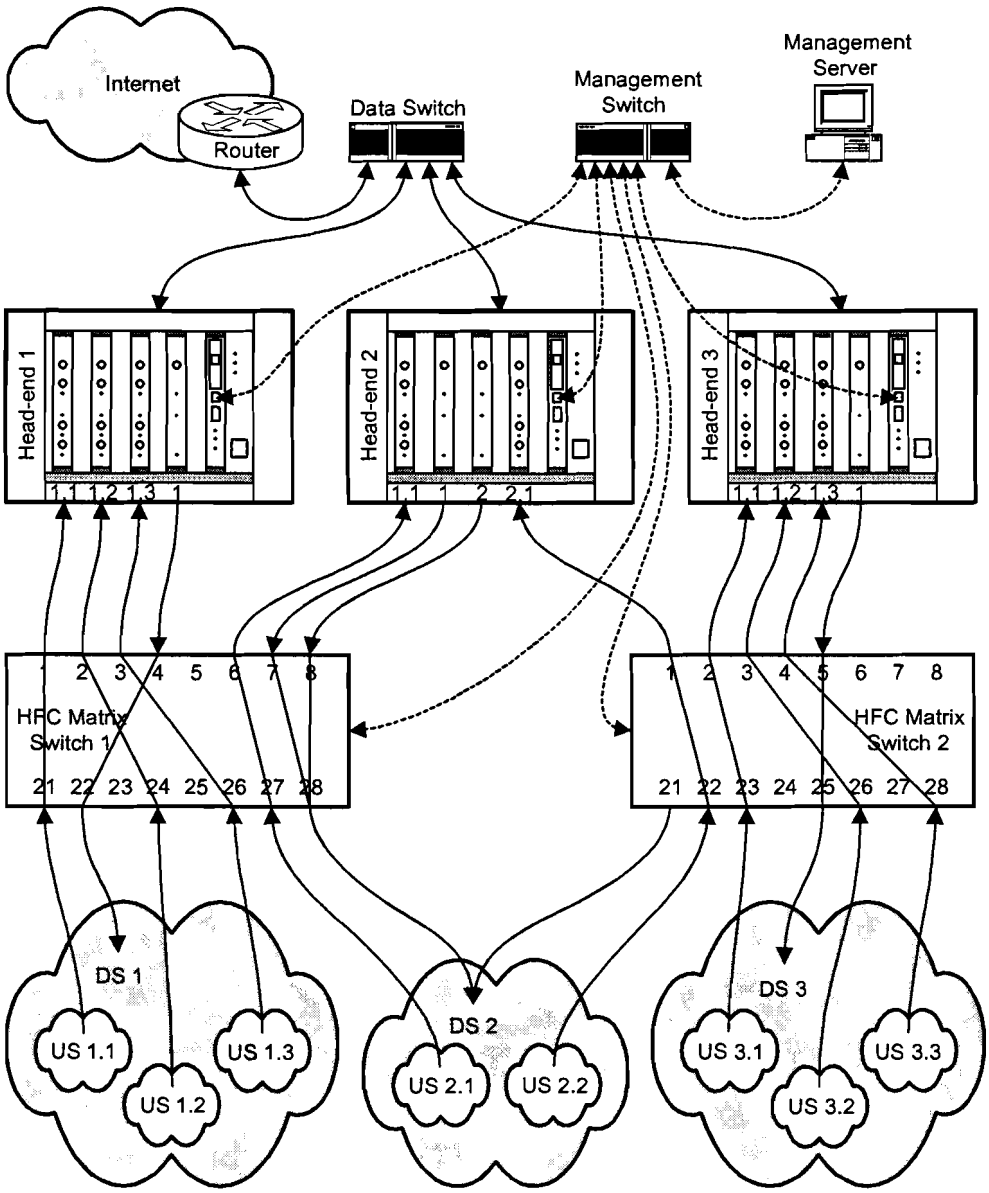

Figure 5.6. General situation; an example detailed for the information sharing function.

Static information

Initially, static information is defined by the operator unless otherwise noticed. It can also be altered by that same operator and by the redundancy system. The static information that a device needs to know of another device in order to be able to take it over, starts with the following information:

- The name of the DS or US area. This name has to be unique for the complete cable modem head-end system. The name of an US area has to be linked with the name of the DS area it is part of.
- The name of the HFC matrix switch that is connected to this area. This is defined by the way the operator physically connects the incoming cables. It should be possible to connect an area to multiple switches. This name has to be unique for the complete system.
- The name of the port of the HFC matrix switch that is connected to this area. Again this is defined by the way the operator physically connects the incoming cables. Besides the connection to multiple switches, it should also be possible to connect an area to multiple ports of the same switch. The name only has to be unique within a single HFC matrix switch.

Note that the combination of the name of the HFC matrix switch and the name of the port of that switch is unique for the complete cable modem head-end system.

At the other side of the HFC matrix switch we have:

- The name of the DS or US channel. This name is has to be unique within a single head-end. The name of an US channel has to be linked with the name of the DS channel that serves the DS area its US area is part of. Note that channel instead of board is used. The reason for this is that an US board of the CableDock 200 contains two US channels.
- The name of the head-end in which the DS or US board containing the DS or US channel is placed. This name has to be unique for the complete cable modem head-end system.
- The name of the HFC matrix switch that is connected to this channel. This is defined by the way the operator physically connects the head-ends with the switches. It should be possible to connect a channel to multiple switches. The name has to be unique for the complete system.
- The name of the port of the HFC matrix switch that is connected to this channel. Again, this is defined by the way the operator physically connects the head-ends with the switches. Besides the connection to multiple switches, it should also be possible to connect a channel to multiple ports of the same switch. As stated before this name only has to be unique within a single switch.

Note that the combination of the name of the DS or US channel and the name of the head-end is unique for the complete cable modem head-end system.

Concerning the HFC matrix switches, more static information can be defined:

- The interconnection of the different ports of the HFC matrix switch. Initially this is programmed by the operator, but this can be re-programmed by the redundancy system. It should be possible to connect a port to multiple other ports. However, this depends on the HFC matrix switch used.
- The possible ways to interconnect the ports of a switch. This is vital information for the redundancy system because this defines what is possible and what is not. For example, it could be possible that a port at the head-end side of the switch can only be interconnected with two ports at the CATV network side of the switch at the same time. For US traffic this means that only two US areas can be added together to form the input of an US channel. For DS traffic this means that a DS channel can only serve two DS areas. This depends on the HFC matrix switch used[1].

The configuration of the DS and US channels also is static information[2]. Concerning a DS channel this static information is the following:

- The output frequency of the DS channel. Actually this refers to the output frequency of the up-converter since all DS channels have the same IF output frequency. The cable modems served by this DS channel program themselves to use this frequency for DS traffic. A change of that frequency would require the modems to re-program the DS frequency and this can only be done by a reboot.
- The output symbol rate of the DS channel. The modems are programmed to use a certain symbol rate for DS traffic. This programmed symbol rate cannot be changed. Therefore, this restricts the number of possible take over solutions in case of a failing device.
- The output modulation type of the DS channel. The modems program themselves to use this modulation type and a change of it would require the modems to re-program this parameter resulting in a reboot of the connected modems.
- The output power level of the DS channel. This depends on the distance between the DS area and the head-end. The cable modems all have automatic gain control at their input so they are able to cope with a certain range of input power levels. However, problems can if a DS channel serves a DS area that is physically very close (thus a low DS power level is used) and then also has to take over a DS area that is very far away and therefore needs a very high DS power level. These modems may suffer from a low signal to noise ratio (S/N). Therefore, they cannot be taken over by this DS channel.

[1] This parameter is also discussed in section 5.4.6.
[2] For a detailed discussion of this configuration, see references [1] and [2].

- The maximum load in Mbit/s this DS channel can handle.
- The maximum number of modems (connections) this DS channel can serve.

Concerning US channels, the static information is the following:

- The input frequency of the US channel. This mainly depends on the US spectrum of the CATV network. The US channel uses the DS channel to program the modems to use this US frequency. If the modems of an US area that has been taken over use the wrong output frequency, their generated US traffic is filtered out as noise by the bandpass filters present at an US board. In worst case a wrong output frequency can even disturb the US traffic of the modems already served by this particular US channel. This may happen if the US frequencies are very close.
- The input symbol rate of the US channel.
- The input modulation type of the US channel. As with the DS symbol rate the modems are programmed to use a certain modulation type for US traffic. This cannot be re-programmed so this restricts the number of possible solutions in case of a failing device.
- The target input power level. The power of the US signal of every modem should have this power level at the input of the US channel. The US channel uses the DS channel to command a modem to adjust its US power until the target power level is reached. The power level needed to send depends on the distance between the sender and the receiver. In practice, the input power needed by the different US channels will be the same and the distance does not change in case of a take over action. Therefore, this static parameter will hardly have any influence.
- The relative time delay compared to the standard timing. This parameter is needed to shift the incoming US traffic in time compared to the fixed clock. An US channel only supports a difference of 1,5 timeslots (1 timeslot = 1 heartbeat of 3 ms) positive or negative to the expected arrival of US traffic. In case of very long distances between the US area and the head-end, the US timeslots are shifted back in time. This parameter depends on the distance and therefore, cannot be changed. This restricts the number of possible solutions in case of a failing device.
- The maximum load in Mbit/s this US channel can handle.
- The maximum number of modems (connections) this US channel can serve.

For US channels, changing the static configuration is less serious. If DS communication with the modems that have been taken over is possible (if the DS parameters match), these modems can be commanded to re-program most of their US parameters. Standard SNMP messages are used for this purpose. However, re-programming the US parameters does cost some time and therefore, it has its influence on the quality level of the take over action.

In section 4.4 the statement has been made that an operator should be able to include and exclude devices from the redundancy system. In practice this means two things: it should be possible to specify if a device should be taken over in case it fails and it should be possible to exclude a device as a possible candidate to take over a failing device. These static parameters are defined per channel. To exclude a complete head-end, all of its DS and US channels should be excluded. This results in the following static information:

- The requirement that a DS or an US channel has to be taken over in case of a failure. This parameter can be used to exclude devices from the automatic redundancy system but still detect them in case of a failure.
- The availability to take over other DS or US channels. With this parameter an operator can exclude the device as a possible candidate to take over a failing device. Note that in case the device itself fails, the redundancy system will still check if another device is able take over this device, except for devices that have the first parameter set to false.

Finally, the static information concerning the configuration of the head-end itself:

- The speed of the data Ethernet interface. This is defined by the physical configuration of the head-end. This parameter is required because a head-end with a 100 Mbit/s Ethernet interface having a full load, cannot be taken over (without graceful degradation) by another head-end with a 10 Mbit/s Ethernet interface; even if it had no load of its own.

The static information just described can always be expanded in order to adapt to certain needs of an operator or the market. For example, it does not include the possibility to use more than one router for payload data. The just described information is the minimum static information needed to use the redundancy system.

Dynamic information
Dynamic information is generated by the device itself, due to normal operation[1]. The general dynamic information that is required, is the following:

- The actual load of a DS or US channel in Mbit/s. The loads of all DS and US channels present in a single head-end should be added to result in the current load of a head-end in Mbit/s. A changed load of a device should be shared only if it differs much from the last shared load.
- The number of currently active modems served by a DS or US channel. As stated before the user point of view is used to define the quality level of the take over action. The degradation of the individual service level can only be calculated if the number of active modems is known. Only active modems are included in the calculations because those are the modems that suffer in case of a failure. Active is defined as: modems that have generated payload data traffic the past certain time interval (probably a few minutes). As with the load, this value should only be shared if it differs much from the previous value. Note that only the number of modems is needed not which modems are in fact active.

The previous defined static and dynamic information is needed to decide which device should take over a failing device. Some of this information is also used for the actual take over action. The actual take over action requires some more dynamic information: information of the modems that need to be taken over. This depends on the quality level of the take over action. As stated before, the take over actions are based upon quality level 3. This requires the following dynamic information per modem:

- IP address of the modem. This address is needed to remove the modem from the routing tables of the router at the Internet side of the network. The next time this router receives a DS packet[2] for this IP address, it will start an ARP request procedure because it does not have this IP address present in its routing tables anymore. A head-end will handle this ARP request as it would normally do. After the ARP request procedure, the router has the new situation in its tables and therefore directs packets for this modem to the new head-end. In fact, the so created situation is the same as if the modem has never been connected to the failing device.
- Name of the modem. This name is needed for two reasons. The first reason is to be absolutely sure that the modem can sign on again. Normally, a sign on procedure also includes some (security) checks. Since we want this modem to sign on, a device that receives a sign on request of this particular modem should skip these checks. The device is forced to accept the sign on of this modem. The second reason is that this name is needed for the just described ARP request procedure. If the router starts an ARP request procedure, the head-end needs to handle that procedure and therefore needs to know that this request is actually directed to one of its modems.
- The level of service (Class of Service (CoS) for a DVB based system) the modem used to have at the failing device. This is needed to be absolutely sure that a modem gets back the same level of service. Note that due to graceful degradation the actual level of service can be lower but this remains the target level of service; in time, this should be the level of service that the modem has.

5.4.4 The information sharing message

The information sharing messages are used to share the information described in the previous section. As stated before, the static and dynamic information is grouped by the devices they describe.

The information sharing message contains the following fields:

- *data_type*. This field is the name of the table that is present in the data field.
- Frame number. This field is for logging and debugging purposes.
- IP address of the device that is described by the information in the *data* field.

[1] For a more detailed discussion of the dynamic parameters used, see references [2] and [20].
[2] DS packet means originating from the Internet and heading for a user's cable modem.

- *message_type*. Actually this field can be seen as the reason for this information sharing message. This field is discussed in section 5.4.5.
- *data*. This field contains the actual information. This information will be presented in a table.

The *data* field and thus actual information can be either static information or dynamic information. The tables containing the information as described in the previous section are presented below. The first row of a table is the name of the parameter presented in that column. This name is used to refer to this parameter throughout the rest of this and the following chapters. The second row of the table gives a short description of that parameter. The columns prior to the double line are the index to the table. Per row, the fields of these columns form a unique key to that row of the table.

The static information is represented by the six tables below.

Table 5.1. The SwitchToCatvCxn table.

Switch name	Output port name	DS/US area name
The name of the HFC matrix switch	The name of the port of the HFC matrix switch at the CATV network side	The name of the DS or US area that is connected to this output port

Table 5.2. The SwitchState table.

Switch name	Input port name	Output port name	Names of possible output ports
The name of the HFC matrix switch	The name of the port of the HFC matrix switch at the head-end side	The name(s) of the ports of the HFC matrix switch at the CATV network side that is (are) interconnected with this input port	The names of all output ports that can possibly be interconnected with this input port

Table 5.3. The SwitchToDeviceCxn table.

Switch name	Input port name	Device name	DS/US channel name
The name of the HFC matrix switch	The name of the port of the HFC matrix switch at the head-end side	The name of the head-end that is connected to this input port	The name of the DS or US channel that is connected to this input port

Table 5.4. The MainDeviceState table.

Device name	Ethernet speed
The name of the head-end	The speed of the Ethernet interface used for payload data (the Internet)

Table 5.5. The DsDeviceState table.

Device name	DS channel name	To be taken over	Available for take over	Frequency	Symbol rate	Modulation type	Power level	Maximum load	Maximum number of modems
The name of the head-end	The name of the DS channel	Should this DS channel be taken over in case of a failure?	Is this DS channel available to take over another failing DS channel?	The frequency of this DS channel	The symbol rate of this DS channel	The modulation type of this DS channel	The output power level of this DS channel	The maximum load in Mbit/s this DS channel can handle	The maximum number of modems this DS channel can serve

Table 5.6. The UsDeviceState table.

Device name	US channel name	To be taken over	Available for take over	Frequency	Symbol rate	Modulation type	Power level	Maximum load	Maximum number of modems	Relative time delay
The name of the head-end	The name of the US channel	Should this US channel be taken over in case of a failure?	Is this US channel available to take over another failing US channel?	The frequency of this US channel	The symbol rate of this US channel	The modulation type of this US channel	The target input power level of this US channel	The maximum load in Mbit/s this US channel can handle	The maximum number of modems this US channel can serve	The time delay in timeslots relative to the standard 3ms heartbeat

The dynamic information is represented by two tables below.

Table 5.7. The DeviceLoad table.

Device name	DS/US channel name	Load	Connections
The name of the head-end	The name of the DS or US channel	The actual load in Mbit/s this DS or US channel has	The number of active modems this DS or US channel serves

Table 5.8. The CxnTakeOver table.

Device name	DS channel name	Modem name	IP address	CoS
The name of the head-end	The name of the DS channel	The name of the modem that is served by this DS channel	The IP address of the modem that is served by this DS channel	The class of service this modem has

As stated before the contents of these tables can be adjusted to the particular needs of an operator or market. Also, Table 5.8 now represents the minimum information needed to take over a modem according to the in section 5.4.2 defined quality level 3. This table needs be expanded when a higher quality level of the take over action is wished.

5.4.5 Basis of the information sharing function

As stated in section 4.4 the information sharing function is based upon delta information; only information that has changed, is distributed to the redundancy systems of other devices. This delta information is broadcasted on the management network.

Before the actual basis of the information sharing function is discussed, one additional note has to be made. Until now we stated that **every** device should have **all** information of the other devices of the redundancy network. This was in order to provide the take over decision function and the take over activating function with input in case of a failing device. However, this requirement has to be refined for the following reason. Whereas the information represented in Table 5.1 up to and including Table 5.7 does not require many bytes, the information represented in Table 5.8 may do so. This can be concluded from the index of this table. Since a single DS channel can serve as much as 2,000 modems [2] and as much as 255 head-ends are supported in the redundancy network, this table could have more than 500,000 entries. In worst case, this could even be more because a single head-end can have more than 1 DS channel. With the information of Table 5.8, 500,000 entries would only require a few Mbytes in total. However, when the information of this table is expanded (by adding a few columns) this could easily get out of hand. There is simply not enough memory space available to store all this information at every device. Therefore, the following strategy is used:

- Information will still be shared using broadcast. The management network is fast enough to handle these amounts of data. Besides, the system is based upon delta information anyway.
- Static and dynamic information needed by the take over decision function is stored at every device participating in the redundancy network. This is the information represented by Table 5.1 up to and including Table 5.7. From now on, this information will be referred to as Dec information.

- Dynamic information needed by the take over activating function (additional to the Dec information) is stored only at the device that precedes the device the information is about in the detection ring. This information is represented by Table 5.8. From now on, this information will be referred to as Act information.

Recall the situation as described in Figure 5.2. In this case the Act information of head-end 2 is only present at head-end 1. The same principle applies for head-end 3, so its Act information is only present at head-end 2.

In practice, it is very much possible that the device that precedes a failing device in the detection ring, is not the best candidate to take over that failing device. In this case the device that actually has to take over the failing one, does not have the Act information of the failing device needed for this action. Therefore, the device that detects the failure and therefore, has the Act information provides the best candidate with the information needed.

Again, recall the situation of Figure 5.2. Suppose that head-end 1 would detect that head-end 2 is failing and it decides that head-end 3 is the best candidate to take over head-end 2. Head-end 3 then is notified by head-end 1 and furthermore, it gets the Act information of head-end 2.

Note that this example points out another thing that needs to be done in case of a take over action. The failing head-end 2 will be removed from the detection ring. This means that head-end 1 now precedes head-end 3 in this detection ring and therefore should have the Act information of this head-end. Head-end 3 provides head-end 1 with this information that until that moment was stored only at the failing head-end 2. This is task of the redundancy system is performed by the information sharing function.

Since the information that is shared is delta information, every change of the information is only sent once. This is not a problem for the network, for we use a LAN network[1]. However, it is always possible that information stored at a devices becomes corrupted, e.g. due to cosmic radiation. Therefore, there has to be a mechanism that re-sends information from time to time (probably a few weeks or even months) in order to provide devices with the possibility to check if their local information is still correct. Note that this applies to all information, both static and dynamic.

When an information sharing message is received by the information sharing function, it first checks if it should process the message. This depends on the *message_type* field and the IP address within the message. If it should further process the message, this processing consists of the update of a locally stored tables with the information present in the *data* field of the message. The table in the *data* field of the message will be processed row by row. Note that a single message can contain multiple changes (rows) of one table. Concerning the information of every row there are three possibilities:

- It is new information. For example, the addition of a DS board (and thus DS channel) or a modem that has signed on. In this case there is no row present in the local table with this index (key). A new row will be created and filled with the information of the message.
- It is changed information. For example, the load of a device or the number of active modems served by a device. In this case there is already a row present in the local table with this index (key). This row will be updated with the information of the message.
- It is information to be removed. For example, a modem that has signed off or an US channel that can no longer be used. In this case there is already a row present with this index (key). The row in the message has a predefined value meaning 'remove this row'. For example the actual data after the index could all be set to '0'. This row will be deleted from the local table.

[1] Ethernet interfaces support re-transmission of Ethernet packets in case of collision. Therefore, on a LAN network packets always arrive.

The processing of an information sharing message depends on the *message_type* field. A detailed discussion of the different message type follows below. The following message types are possible:

- *delta_information*. This message type is used to describing a normal change of the information.
- *checkup_information*. This message type is used to provide a check of the information stored at the devices participating in the redundancy network.
- *failing_device_clearup*. This message type is used to clear the information of a failing device.
- *detection_ring_change*. This message type is used when devices are added to or removed from the detection ring.
- *take_over_information*. This message type is used to provide the best candidate to take over a failing device with the information of that failing device.
- *device_change*. This message type is used in the initialization or shutdown state of a redundancy system.

delta_information

When something changes in the situation of a device, the software of that device triggers the information sharing function that the local situation has changed. In some cases a change will be shared immediately (for example the addition of an extra DS board) and in some cases the changes are buffered until the change is big enough to share (for example the load of a device). The information sharing function reacts by adjusting the appropriate table. Then an information sharing message will be generated with the *message_type* set to *delta_information.*

When the information sharing function receives such a message it will first check if it should further process the message. As stated in the previous sections, Dec information is stored at every device. Act information is stored only at the preceding device in the detection ring. When the IP address present in the message matches the IP address of the next device in the detection ring, the message is processed further. This processing is done according to the just described procedure.

checkup_information

As stated before it is necessary to provide a checking mechanism to see if the information stored locally is not corrupted. For this reason the *checkup_information* message type exists. The checkup procedure is initiated by the device the information describes. The reason for this is obvious. If this information were corrupted, this device would probably not function correctly anymore and this would have been detected by another device. Therefore, every few weeks the software of a device triggers the information sharing function to generate a *checkup_information* message. This message contains all information of a certain *data_type* concerning this device. Since each message can only contain one *data_type*, this procedure needs to be done eight times. Note that it is not required to do it all at the same time; each *data_type* can have its own time interval.

In case the information sharing function receives such a message, this message will be processed in the same way as normally would be done. The only difference is that the message does not contain just a few rows but it contains all information of a certain *data_type* concerning a certain device.

failing_device_clearup

In case a failing device is detected, that device will be removed from the detection ring. This makes the information concerning this device that the other devices have stored, useless. Therefore, it needs to be cleared. This procedure is initiated by the device that detects the failing one. The reason for this is that this device has all information (Dec and Act) of the failing device. To command the other devices to clear the locally stored information of the failing one, all information regarding this failing device is presented as 'to be removed'. In practice, this results in the broadcast of six information sharing messages; each one with one of the following tables: Table 5.3 up to and including Table 5.8. Note that the information concerning a failing device is first removed from the local tables by all other devices and then later on some of this information is added again as delta information of the device that has taken over the failing one.

When such a message is received, it is processed as normal. Note that the information sharing function is able to conclude from the received message that something is wrong. The reason for this is

that the IP address of the sender (the source IP address in the IP header) does not match the IP address present in the information sharing message (the IP address of the device this information describes).

detection_ring_change
Act information (Table 5.8) of a device is only stored at the preceding device in the detection ring. Since the detection ring can change, the *detection_ring_change* message type is present. There are two possibilities concerning the change of the detection ring:

- A device is removed from the detection ring. This happens if the device is either removed by another device in case of a failure or if it enters the shutdown state.
- A device is added to the detection ring. This happens if a device enters its initialization state.

For now, only the removal from the detection ring in case of a failure will be described. The other two are described in section 5.7.

The device that detects a failing device first broadcasts a *failing_device_clearup* message. The locally stored information of this failing device is not cleared at this time because it is used as input for the take over decision function. Meanwhile, the detection function will remove the failing device from the detection ring. Therefore, the Act information of the device preceded in the new detection ring should be stored locally. To get this information, the device that wants the Act information broadcasts an information sharing message with the *message_type* set to *detection_ring_change*. The *data_type* in this message is set to the name of the information table it wants to have and the *data* field itself is left blank. In case a failing device has been removed from the detection ring, the only information the device wants to have, is the Act information, for it should already have all other information. The IP address present in the message, is set to the IP address of the device it wants the information of.

When the information sharing function receives such a message, it first checks if it should reply to this message. If the IP address in the message matches its own IP address, another device wants its information. It reacts with a broadcast of the same message, only with the requested information presented in the *data* field. The requesting device processes the reply because it recognizes the IP address in the message as the IP address of the device it wants information of.

take_over_information
This message type is used when a failing device has to be taken over by an other device than the device that detected this failing device. In this case the device that should perform the take over action does not have the Act information needed. To provide this device with this information, the device that has detected the failing one, broadcasts an information sharing message with the *message_type* set to *take_over_information*. Furthermore, the IP address in the message is set to the IP address of the failing device.

The device that is the best candidate to take over the failing one, recognizes the IP address in the message as the IP address of the device it should take over. In order to actually recognize this IP address, this device first needs to be noticed that is should enter the alarm state and start the take over activating function with this particular device as input. This is described in section 5.6.

device_change
This final message type is used only by redundancy systems that are in their initialization or shutdown state. This message type is discussed in detail in section 5.7.2 and 5.7.3.

5.4.6 Remarks regarding the information sharing function

During the project some issues concerning the implementation of the information sharing function were already discussed. These issues are presented in this section.

First, the statement that all information sharing messages should be broadcasted on the management network. It may seem to be a cumbersome manner to broadcast information instead of unicast it, this is in fact the power of the information sharing function. As can be seen in the previous section, the

choice whether to process a message of not, basically comes down to the same two things every time. The first is that Dec information should always be processed and stored. The second is that Act information should only be processed and stored in case it describes the next device in the detection ring or in case it should be used for a take over action. Concerning the broadcasted information, the only thing that differs is the event that triggers the broadcast of the message and the amount of data present in the message. Besides this simplicity, the power can also be seen in the fact that the sender of a message does not (need to) know who the receiver is. Therefore, it just handles every message according to the same principle and this results in a very simple function. In fact, it only performs its own simple tasks and does not (need to) keep track of all complex things that happen around its device. This simplicity is exactly one of the main design requirements. Note that this simplicity also results in the fact that the information sharing function is very well (stand-alone) testable.

In practice multicast would probably be a better option than broadcast. Suppose that an operator has redundancy networks at several different geographical locations. For example, in different districts of a city. If these redundancy networks are interconnected (this is the case if the management networks are interconnected), broadcasting information on one redundancy network generates unwanted (and useless) load on another redundancy network. This could be solved by using multicast messages.

The unique naming of the different signals that describe the various parts of the cable modem head-end system is very important for the redundancy system. Some possible ways to implement a unique naming are discussed below.

First, the unique naming of the DS and US areas. One of the requirements of that naming is that the link between a DS and its US areas can be recognized. One way to solve this problem is to assign a unique number to every DS area and give the linked US areas a number according to *'DS area number'.X* where *X* is an increasing value. An example of this method for naming the DS and US areas is presented in Figure 5.6.

The same method can be used to name the DS and US channels within a head-end. An example of this numbering is also presented in Figure 5.6. Note that currently DS and US channels of a CableDock 200 are not named in this way [2].

An obvious choice for an unique name of the head-end is its IP address. Since the HFC matrix switches need to support to be programmed by IP, they also have an IP address. This IP address can be used to give the switches a unique name.

The MAC address of a cable modem can be used as unique name for this cable modem. The ARP request procedure described in section 5.4.3 also uses this MAC address. Also, its NIU index[1] can be used. However, in this case a translation from this NIU index to the MAC address has to be available at the device that receives an ARP request from the router.

Furthermore, a remark has to be made concerning the tables representing the different data types. In practice, it may be desirable to change the key (index) for every table. For example, Table 5.3 now uses the name of the HFC matrix switch together with the name of an input port as key. In practice, this table is going to be used to determine the names of the switches and the names of the input ports that are connected to the channels of a failing device. Therefore, a key consisting of the name of a device together with the name of a DS or US channel would probably speed up the table look-ups. Note that this is a unique key too.

The information represented by Table 5.2 may need to be expanded. The last column of this table only describes the possible output ports for a certain input port. It could be desirable that it describes the possible output port combinations for a certain input port. The reason for this is that the take over decision system needs to now if a certain DS or US channel (and thus input port) is able to take over an output port while still being able to serve its own DS or US area. This means that in case of an

[1] For DVB based systems, a cable modem is also called Network Interface Unit (NIU). These NIU's have a NIU index assigned to them as a unique name withing a single CableDock 200 [2].

US channel that needs to be taken over, it is important to know if two output ports can be added together and interconnected with an input port. In case of a DS channel, it is important to know if an input port can be split and interconnected to two output ports.

As stated in section 5.4.3, one cannot change the symbol rate of a DS channel and the modulation type of an US channel. It is possible to change the modulation type of a DS channel and the symbol rate of an US channel. In practice, not all DS modulation types and US symbol rates are supported by the cable modems. Since this could restrict the number of possible solutions, this information should be available as input for the take over decision function. However, it is not present in Table 5.5 and Table 5.6. The reason for this is that in practice, an operator will probably not have cable modems that use different DS modulation types and US symbol rates. Representation of these restrictions can be a future expansion of the information concerning these parameters.

At this time, multiple connections per modem are not supported by the redundancy system. The tables only represent information of the modem. Currently, the CableDock 200 also does not support multiple connections per modem [2]. However, in the near future this could change and then a maximum of 10 connections per modem is possible. This implies that Table 5.8 needs to be expanded.

It has been noticed that the actual CoS a modem gets after a take over action, can differ from the CoS it used to have at the failing device. The reason for this is that in case the resulting load of a device after the take over action is higher than the Ethernet speed of that device, the individual bandwidth of the modems has to be decreased. Also, DS and US boards cannot handle loads of more than 40 Mbit/s and 6 Mbit/s respectively. In order to use the influence of this reduced bandwidth in the decision of the best candidate, the CoS of the active connections should be known. The reason for this is that some CoS levels cannot be degraded because then the connection that uses this CoS will be broken. For example, VoIP uses a high CoS because delays in the speech are unwanted. If a connection used by VoIP would be limited, the resulting connection is useless for VoIP. So when using the influence of a degraded service level in the decision for the best candidate to take over, the CoS the different connections use should also be taken into account. However, this would require that every device has all CoS information of all active connections and this is contrary to the fact that Table 5.8 should not be stored at all devices. Therefore, this is not an option. A middle coarse would be to expand Table 5.7 with the number of active modems (connections) per CoS level. This information can be used by the take over decision function.

Finally, we want to notice that quite some rules were presented about when information should be shared, what information is necessary and how this information should be handled. As stated before, it is very simple to adjust the tables and therefore increase the functionality of the complete redundancy system. However, one should always keep in mind what the costs and the benefits of such a change are. If the take over action gets just a little bit better (faster) the few times a year that it would be required to take over a failing device and the costs of this change are that much more information needs to be broadcasted and processed all the time, one could wonder if this was a useful change. The information sharing function as discussed in this section represents the minimum functionality of this part of the redundancy system, required.

5.4.7 Example of the information sharing function

The example presented in this section is based upon the situation as described in Figure 5.6. Suppose there is a detection ring that works in the following order: head-end 1 (IP address 192.138.0.24) and head-end 2 (IP address 192.138.0.13). Furthermore, there is a head-end 3 (IP address 192.138.0.38), but its redundancy system is inactive. There are two switches: HFC matrix switch 1 (IP address 192.138.0.2) and HFC matrix switch 2 (IP address 192.138.0.3).

The example starts with the situation that head-end 1 and head-end 2 are already active for a while. Therefore, both head-ends have the following tables stored. In these tables a value of '...' means that there is a value present, but this value is not specified in this example.

Table 5.9. SwitchToCatvCxn table.

Switch name [IP]	Output port name	DS/US area name
192.138.0.2	21	1.1
192.138.0.2	22	1
192.138.0.2	24	1.2
192.138.0.2	26	1.3
192.138.0.2	27	2.1
192.138.0.2	28	2
192.138.0.3	21	2
192.138.0.3	22	2.2

Table 5.10. SwitchState table.

Switch name [IP]	Input port name	Output port name	Names of possible output ports
192.138.0.2	1	21	21,22,23,24,25,26,27,28
192.138.0.2	2	24	21,22,23,24,25,26,27,28
192.138.0.2	3	26	21,22,23,24,25,26,27,28
192.138.0.2	4	22	21,22,23,24,25,26,27,28
192.138.0.2	6	27	21,22,23,24,25,26,27,28
192.138.0.2	7	28	21,22,23,24,25,26,27,28
192.138.0.2	8	28	21,22,23,24,25,26,27,28
192.138.0.3	1	22	21,22,23,24,25,26,27,28
192.138.0.3	2	23	21,22,23,24,25,26,27,28

Table 5.11. SwitchToDeviceCxn table.

Switch name [IP]	Input port name	Device name [IP]	DS/US channel name
192.138.0.2	1	192.138.0.24	1.1
192.138.0.2	2	192.138.0.24	1.2
192.138.0.2	3	192.138.0.24	1.3
192.138.0.2	4	192.138.0.24	1
192.138.0.2	6	192.138.0.13	1.1
192.138.0.2	7	192.138.0.13	1
192.138.0.2	8	192.138.0.13	2
192.138.0.3	1	192.138.0.13	2.1

Table 5.12. MainDeviceState table.

Device name [IP]	Ethernet speed [Mbit/s]
192.138.0.24	10
192.138.0.13	100

Table 5.13. DsDeviceState table.

Device name	DS channel name	To be taken over	Available for take over	Freq.	Symbol rate	Mod. type	Power level	Max. load	Max. number of modems
192.138.0.24	1	true	true	...	...	...	...	40	2000
192.138.0.13	1	true	true	...	...	...	...	40	2000
192.138.0.13	2	true	true	...	...	...	...	40	2000

Table 5.14. UsDeviceState table.

Device name	US channel name	To be taken over	Available for take over	Freq.	Symbol rate	Mod. type	Power level	Max. load	Max. number of modems	Relative time delay
192.138.0.24	1.1	true	true	...	...	...	...	6	2000	...
192.138.0.24	1.2	true	true	...	...	...	...	6	2000	...
192.138.0.24	1.3	true	true	...	...	...	...	6	2000	...
192.138.0.13	1.1	true	true	...	...	...	...	6	2000	...
192.138.0.13	2.1	true	true	...	...	...	...	6	2000	...

Table 5.15. DeviceLoad table.

Device name [IP]	DS/US channel name	Load [Mbit/s]	Connections
192.138.0.24	1	35	1700
192.138.0.24	1.1	5	500
192.138.0.24	1.2	4	500
192.138.0.24	1.3	5	700
192.138.0.13	1	20	400
192.138.0.13	1.1	3	400
192.138.0.13	2	27	900
192.138.0.13	2.1	6	900

Furthermore, head-end 1 has the following table with Act information of head-end 2 stored.

Table 5.16. CxnTakeOver table.

Device name	DS channel name	Modem name [NIU]	IP address [IP]	CoS
192.138.0.13	1	24	196.126.0.36	2
192.138.0.13	1	25	196.126.0.37	1
etc.	etc.	etc.	etc.	etc.

Head-end 2 has the following table with Act information of head-end 1 stored.

Table 5.17. CxnTakeOver table.

Device name	DS channel name	Modem name [NIU]	IP address [IP]	CoS
192.138.0.24	1	13	196.127.0.40	3
192.138.0.24	1	14	196.127.0.41	3
etc.	etc.	etc.	etc.	etc.

In order to add head-end 3 to the redundancy network, the operator first has to configure the Dec information concerning this head-end. This results in the following tables.

Table 5.18. SwitchToCatvCxn table.

Switch name [IP]	Output port name	DS/US area name
192.138.0.3	23	3.1
192.138.0.3	25	3
192.138.0.3	26	3.2
192.138.0.3	28	3.3

Table 5.19. SwitchState table.

Switch name [IP]	Input port name	Output port name	Names of possible output ports
192.138.0.3	2	23	21,22,23,24,25,26,27,28
192.138.0.3	3	26	21,22,23,24,25,26,27,28
192.138.0.3	4	28	21,22,23,24,25,26,27,28
192.138.0.3	5	25	21,22,23,24,25,26,27,28

Table 5.20. SwitchToDeviceCxn table.

Switch name [IP]	Input port name	Device name [IP]	DS/US channel name
192.138.0.3	2	192.138.0.38	1.1
192.138.0.3	3	192.138.0.38	1.2
192.138.0.3	4	192.138.0.38	1.3
192.138.0.3	5	192.138.0.38	1

Table 5.21. MainDeviceState table.

Device name [IP]	Ethernet speed [Mbit/s]
192.138.0.38	100

Table 5.22. DsDeviceState table.

Device name	DS channel name	To be taken over	Available for take over	Freq.	Symbol rate	Mod. type	Power level	Max. load	Max. number of modems
192.138.0.38	1	true	true	...	...	...	...	40	2000

Table 5.23. UsDeviceState table.

Device name	US channel name	To be taken over	Available for take over	Freq.	Symbol rate	Mod. type	Power level	Max. load	Max. number of modems	Relative time delay
192.138.0.38	1.1	true	true	...	...	...	...	6	2000	...
192.138.0.38	1.2	true	true	...	...	...	...	6	2000	...
192.138.0.38	1.3	true	true	...	...	...	...	6	2000	...

During the initialization state of the redundancy system of head-end 3, the information sharing function broadcasts these tables. The other redundancy systems process and add this information to their already existing local tables. Furthermore, head-end 3 receives the Dec information concerning head-ends 1 and 2 and adds it to its local tables. This procedure is described in section 5.7.1.

When head-end 3 is part of the redundancy network and thus part of the detection ring, some information sharing messages with the *message_type* set to *detection_ring_change* are broadcasted to adjust the local Act information stored at the different head-ends to the new situation. Suppose that head-end 3 has been positioned in the detection ring after head-end 2 en before head-end 1. As a result head-end 3 will get the Act information of head-end 1; head-end 3 receives and stores the following information.

Table 5.24. CxnTakeOver table.

Device name	DS channel name	Modem name [NIU]	IP address [IP]	CoS
192.138.0.24	1	13	196.127.0.40	3
192.138.0.24	1	14	196.127.0.41	3
etc.	etc.	etc.	etc.	etc.

Head-end 2 receives and stores the following Act information of head-end 3.

Table 5.25. CxnTakeOver table.

Device name	DS channel name	Modem name [NIU]	IP address [IP]	CoS
192.138.0.38	1	2	196.122.0.90	1
192.138.0.38	1	4	196.122.0.92	2
etc.	etc.	etc.	etc.	etc.

When head-end 3 starts serving its DS and US areas, load will be generated. This results in a broadcast *delta_information* messages with the *data_type* set to *DeviceLoad*.

Finally, an example of the information sharing message itself. Suppose the load of head-end 2 (in fact the load of DS channel 2 and the load of US channel 2.1) changes with a value higher than the threshold to broadcast this information. This results in the following information sharing message:

- *data_type*: *DeviceLoad*
- Frame number: 378
- IP address: 193.138.0.13
- *message_type*: *delta_information*
- *data*: 192.138.0.13 2 20 900
  192.138.0.13 2.1 4 900

Note that compared to the old situation only the load has changed (DS from 27 to 20 Mbit/s and US from 6 to 4 Mbit/s) and not the number of (active) connections served by the different boards. In practice, this means that there are still 900 cable modems active, but they just generate less data traffic.

5.5 The take over decision function

5.5.1 Introduction

This function of the redundancy system is only activated when a failing device is detected. The task of this function is to decide which device is the best candidate to take over the failing device, from now on referred to as candidate. This decision is based upon knowledge of the network and the other devices. This knowledge is provided by the information sharing function. In section 5.4 some parts concerning the take over decision function were already described. Note that this function is an internal function only; it does not communicate with the take over decision functions of other redundancy systems. There is only interaction with functions of the redundancy system within the same device. In this section the working of the take over decision function will be discussed. An example of this function is presented in chapter 6.

5.5.2 Basis of the take over decision function

There are several ways to make a decision based upon a set of inputs. We chose to base this decision upon a quality factor and different weights. The quality factor of a candidate consists of multiple sub quality factors. A sub quality factor describes 'how ideal' this parameter of the candidate matches that same parameter of the failing device. These sub quality factors are multiplied and given a relative weight to form the quality factor of a candidate. This total quality factor describes 'how ideal' this candidate is to take over the failing device. All quality factors are normalized. Mathematically a sub quality factor is defined as:

$$Q_n = f_n(k) \qquad (5.7)$$

Each sub quality factor Q_n is a function of a certain input parameter k. This input parameter represents the parameter of the candidate that we want to test. Every sub quality factor is normalized:

$$Q_n \in [0,1] \qquad (5.8)$$

These sub quality factors Q_n are used to form the total quality factor of a device Q_{device}. Mathematically this total quality factor is defined as:

$$Q_{device} = \prod_n (Q_n)^{w_n} \qquad (5.9)$$

Each weight w_n defines the relative importance of the n^{th} sub quality factor. The way the sub factors are defined later on this section, states that each weight w_n should equal 1. However, if an operator wants to adjust the results of the take over decision function because some sub quality factors are more important to him than others, he only needs to adjust the weights. So the actual formulas can be coded in the software and an additional set of weights (in the configuration files of the redundancy system) allow the operator to affect the results. The weights are defined as:

$$w_n \in [0,\infty) \qquad (5.10)$$

If w_n = 0 this particular sub quality factor is 'disabled'[1]. By making w_n less than 1, the relative importance of this sub quality factor decreases. By making w_n higher than 1, the relative importance of this sub quality factor increases. Because the result is still normalized, the boundaries of the total quality factor are:

$$Q_{device} \in [0,1] \qquad (5.11)$$

A Q_{device} = 0 means that this particular candidate is unable to take over the failing device. A Q_{device} = 1 means that this particular candidate is ideal to take over the failing one. In practice, this latter result will only occur when the candidate is a complete redundant device.

[1] For, $x^0 \equiv 1$.

The input of the take over decision function is the one hand all parameters of the failing device and on the other hand a set with all (parameters of all) the other devices in the redundancy network. In a special case the device that makes the decision is itself not a part of that set of inputs[1]. The knowledge of the failing device and the candidates is provided by the information sharing function. Since this information is represented by Table 5.1 up to and including Table 5.7 (Dec information) these tables are used to describe the different sub quality factors.

There are two types of information (static and dynamic) and therefore, there are also two types of sub quality factors: the factors based upon static information and the factors based upon dynamic information. Furthermore, the comparison between the parameters of the failing device and the candidate can be split into the following parts:

- For intra head-end redundancy only the failing board has to be compared with all other boards of the same type present in the same device. This subject returns in section 5.8.4.
- For inter head-end redundancy the complete head-end has to be compared with the failing head-end. This comparison can be divided into the comparison of the different boards of the candidate with the boards of the failing device using the information of these boards. This means that the System Controller of the candidate has to be compared with the System Controller of the failing head-end. The information needed for this comparison is present in Table 5.4. Furthermore, the DS boards (and thus DS channels) of the candidate have to be compared with the DS boards of the failing head-end. The information needed for these comparisons is present in Table 5.1, Table 5.2, Table 5.3, Table 5.5 and Table 5.7. Finally, the US boards (and thus US channels) of the candidate have to be compared with the US boards of the failing head-end. The information needed for these comparisons is present in the Table 5.1, Table 5.2, Table 5.3, Table 5.6 and Table 5.7.

Before the actual calculations of the sub quality factors are discussed, some definitions concerning the failing device and the candidate are made. In fact, these definitions can be seen as pre-filtering. When actually programming the take over decision function in software, the following definitions could be substituted in the formulas of sub quality factors.

Since the calculation of the quality factor of a candidate is based upon a comparison of the set of its parameters with the set of parameters of the failing device, it is logically to use mathematical collections to describe the calculations. The basic elements of the collections that are used in these calculations are:

- The element *ds* meaning a DS channel that is participating in the redundancy network.
- The element *us* meaning an US channel that is participating in the redundancy network.
- The element *switch* meaning a HFC matrix switch that is used by the redundancy network.
- The element *input_port* meaning an input port of a HFC matrix switch.
- The element *headend* meaning a head-end that is participating in the redundancy network.

Each element represents some of the parameters of the candidate and the failing device. These parameters represent the information as presented in Table 5.1 up to and including Table 5.7 (Dec information). The links between the information, parameters and elements are presented below.

The element *input_port* consists of:

- *input_port_name* — Table 5.2, 2nd column
- *output_port_name* — Table 5.1, 3rd column
- *possible_output_ports* — Table 5.2, 4th column

[1] This is the case when the intra head-end redundancy system of a head-end cannot solve the problem. See also section 5.2.2.

The element *ds* consists of:
- *ds_channel_name*
- *to_be_taken_over* Table 5.5, 3rd column
- *available_for_take_over* Table 5.5, 4th column
- *frequency* Table 5.5, 5th column
- *symbol_rate* Table 5.5, 6th column
- *modulation_type* Table 5.5, 7th column
- *power_level* Table 5.5, 8th column
- *maximum_load* Table 5.5, 9th column
- *maximum_modems* Table 5.5, 10th column
- *load* Table 5.7, 3rd column
- *active_connections* Table 5.7, 4th column
- *switch_name* Table 5.3, 1st column
- *switch_input_port* = the collection of all elements *input_port* that are connected to this element *ds*

The element *us* consists of:
- *us_channel_name*
- *to_be_taken_over* Table 5.6, 3rd column
- *available_for_take_over* Table 5.6, 4th column
- *frequency* Table 5.6, 5th column
- *symbol_rate* Table 5.6, 6th column
- *modulation_type* Table 5.6, 7th column
- *power_level* Table 5.6, 8th column
- *maximum_load* Table 5.6, 9th column
- *maximum_modems* Table 5.6, 10th column
- *relative_time_delay* Table 5.6, 11th column
- *load* Table 5.7, 3rd column
- *active_connections* Table 5.7, 4th column
- *switch_name* Table 5.3, 1st column
- *switch_input_port* = the collection of all elements *input_port* that are connected to this element *us*

The element *switch* consists of:
- *switch_name*
- *input_ports* = the collection of all elements *input_port* that are present at this switch

The element *headend* consists of:
- *headend_name*
- *Ethernet* Table 5.4, 2nd column
- *DS_channel* = the collection of all elements *ds* that are present in this head-end
- *US_channel* = the collection of all elements *us* that are present in this head-end

Finally, the following definitions are used:

$DS\_channel_X$ = the collection of all elements *ds* that are present in head-end *X*.

$US\_channel_X$ = the collection of all elements *us* that are present in head-end *X*.

$N(X)$ = # elements in the collection *X*.

Furthermore, '*_F*' or the subscript '*f*' is used to refer to the failing device and '*_C*' or the subscript '*c*' is used to refer to the candidate that is being tested to take over the failing device.

Concerning the failing device, not every DS and US channel of the device has to be taken over. Only the DS and US channels that are configured to be taken over in case of a failing device, have to be used as input. This results in the following sets of inputs concerning the failing device:

$$DS_f = \{ds \in DS\_channel_f \mid ds(to\_be\_taken\_over) = true\} \quad (5.12)$$

$$US_f = \{us \in US\_channel_f \mid us(to\_be\_taken\_over) = true\} \quad (5.13)$$

The set of the names of HFC matrix switches that have DS channels of the failing device connected to them, is defined as:

$$SW_{DS_f} = \bigcup_{ds \in DS_f} ds(switch\_name) \quad (5.14)$$

This same principle is used to define the set of names of HFC matrix switches that have US channels of the failing device connected to them:

$$SW_{US_f} = \bigcup_{us \in US_f} us(switch\_name) \quad (5.15)$$

Both sets defined by formula 5.14 and 5.15 are collections of names of HFC matrix switches. An element of such a collection will be referred to as sw_f:

$$sw_f \in SW_{DS_f} \cup SW_{US_f} \quad (5.16)$$

With these definitions, the sets of inputs concerning the candidate can be filtered. Not all DS and US channels of the candidate are allowed to be used to take over channels of the failing device. Concerning the DS channels, only DS channels that are allowed to take over other DS channels and that are connected to HFC matrix switches that also have DS channels of the failing device connected to them, have to be used as input. This results in the following set of DS channels:

$$DS_c = \{ds \in DS\_channel_c \mid ds(available\_for\_take\_over) = true \wedge ds(switch\_name) \in SW_{DS_f}\} \quad (5.17)$$

The same principle is applied to the US channels of the candidate, resulting in:

$$US_c = \{us \in US\_channel_c \mid us(available\_for\_take\_over) = true \wedge us(switch\_name) \in SW_{US_f}\} \quad (5.18)$$

The sets of inputs defined by formula 5.12, 5.13, 5.14, 5.15, 5.17 and 5.18 describe all DS and US channels and names of switches that are used as input for the calculations of the sub quality factors. As will become clear in the next section, most of the sub quality factors are calculated per switch. Therefore, the situation per switch has to defined to make calculations easier.

For DS channels, the name of the switch should be part of the collections defined by formula 5.14. Per name sw_f of a switch, the collection of elements *ds* that the candidate has connected to that switch, is defined as:

$$DS_{sw_c} = \{ds \in DS_c \mid ds(switch\_name) = sw_f\} \quad (5.19)$$

The collections of elements *ds* that the failing device has connected to the same switch is defined as:

$$DS_{sw_f} = \{ds \in DS_f \mid ds(switch\_name) = sw_f\} \quad (5.20)$$

In these formulas the name of a switch sw_f is part of the collection:

$$sw_f \in SW_{DS_f} \quad (5.21)$$

This same principle is applied to the US channels. Per name sw_f of a switch, the collection of elements *us* that the candidate has connected to that switch, is defined as:

$$US_{sw_c} = \{us \in US_c \mid us(switch\_name) = sw_f\} \quad (5.22)$$

The collections of elements *ds* that the failing device has connected to the same switch is defined as:

$$US_{sw_f} = \{us \in US_f \mid us(switch\_name) = sw_f\} \quad (5.23)$$

In these formulas the name of a switch sw_f is part of the collection:

$$sw_f \in SW_{US_f} \quad (5.24)$$

Formula 5.19, 5.20, 5.22 and 5.23 are used to when referring to the situation of a single switch. Note that the collections defined by these formulas are partial collections of the collections defined by formula 5.12, 5.13, 5.17 and 5.18. For example, the following equation holds:

$$\bigcup_{sw_f \in SW_{DS_f}} DS_{sw_f} = DS_f \qquad (5.25)$$

The following definition concerning the load of the failing device are made:

$$load\_ds_f = \sum_{ds \in DS_f} ds(load) \quad and \quad load\_us_f = \sum_{us \in US_f} us(load) \qquad (5.26)$$

$$load_f = load\_ds_f + load\_us_f \qquad (5.27)$$

This means that the load of the failing device used in the calculations, is only the load of the DS and US channels that should be taken over. For the candidate the loads are defined as:

$$load\_ds_c = \sum_{ds \in DS\_channel_c} ds(load) \quad and \quad load\_us_c = \sum_{us \in US\_channel_c} us(load) \qquad (5.28)$$

$$load_c = load\_ds_c + load\_us_c \qquad (5.29)$$

This means that the load of the candidate is the load of all DS and US channels present in that device.

Finally, two special cases are considered. The first one occurs when none of the DS and US channels of the failing device should be taken over. Mathematically this is described as:

$$N(DS_f) = 0 \wedge N(DS_f) = 0 \qquad (5.30)$$

Since there is nothing to take over, the take over decision function notifies the management system that the failing device should not be taken over. Then it ends its operation and the redundancy system returns to the running state.

The second special case occurs when none of the DS and US channels of the candidate (and therefore the whole candidate) are allowed to take over failing channels or if none of the DS and US channels are connected to a switch that also has DS or US channels of the failing device connected to it. In this case, this particular candidate is useless. Mathematically this is described as:

$$N(DS_c) = 0 \wedge N(DS_c) = 0 \qquad (5.31)$$

This particular candidate is removed from the set of possible candidates and the take over decision function continues with the next candidate.

5.5.3 Calculation of the sub quality factors

The sub quality factors that define the quality factor of a candidate are:

- Q_{device_load} Is the Ethernet interface of the candidate able to handle the load of the failing device?
- Q_{DS_ch} Does the candidate have enough DS channels available to take over the DS channels of the failing device?
- Q_{US_ch} Does the candidate have enough US channels available to take over the US channels of the failing device?
- Q_{DS_sw} Is it possible to re-program the switch in such way that the maximum number of DS channels can be taken over?
- Q_{US_sw} Is it possible to re-program the switch in such way that the maximum number of US channels can be taken over?
- Q_{DS_load} Are the DS channels of the candidate able to handle the load of the DS channels of the failing device?
- Q_{US_load} Are the US channels of the candidate able to handle the load of the US channels of the failing device?
- Q_{DS_CXN} Are the DS channels of the candidate able to also handle the number of active connections that are served by the DS channels of the failing device?
- Q_{US_CXN} Are the US channels of the candidate able to also handle the number of active connections that are served by the US channels of the failing device?

- $Q_{DS\_frequency}$ Does the candidate have enough DS channels available using the same output frequency as the DS channels of the failing device?
- $Q_{US\_frequency}$ Does the candidate have enough US channels available using the same frequency as the US channels of the failing device?
- $Q_{DS\_modulation}$ Does the candidate have enough DS channels available using the same modulation type as the DS channels of the failing device?
- $Q_{US\_modulation}$ Does the candidate have enough US channels available using the same modulation type as the US channels of the failing device?
- Q_{DS_symbol} Does the candidate have enough DS channels available using the same symbol rate as the DS channels of the failing device?
- Q_{US_symbol} Does the candidate have enough US channels available using the same symbol rate as the US channels of the failing device?
- Q_{DS_power} Does the candidate have enough DS channels available using the same output power level as the DS channels of the failing device?
- Q_{US_delay} Does the candidate have enough US channels available using the same relative time delay as the US channels of the failing device?

The Q_{device_load} factor is the only factor that can be calculated for the complete device at one go. All other factors need to be calculated per switch. The switches that are used in the calculations, have names that are part of the collection:

$$SW_f = SW_{DS_f} \cup SW_{US_f} \qquad with \qquad sw_f \in SW_f \tag{5.32}$$

Note that this is a collection of names (strings) only, not elements of the previously defined type switch. As stated in the previous section, formula 5.19, 5.20, 5.22 and 5.23 are used to refer to the situation of a single switch.

Q_{device_load}

This sub quality factor describes if the candidate has enough bandwidth available to take over the load of the failing device at the Ethernet side. Using formula 5.27 and 5.29 the ideal case would be:

$$load_c + load_f \le headend_c(Ethernet) \tag{5.33}$$

It is important to notice that the resulting load may become higher than the maximum load of Ethernet interface of the candidate. This may sound strange however, the reason for this is that the bandwidth of the individual modems can be decreased. This would result in less load generated per modem. This results in less load per DS and US channel and thus less load per device. Also, note that the resulting load can never be higher than the maximum loads of the Ethernet interfaces added together:

$$0 \le load_c + load_f \le headend_c(Ethernet) + headend_f(Ethernet) \tag{5.34}$$

Since the resulting load is related with the speed of the Ethernet interface of the candidate, this resulting load is normalized using that speed. The resulting normalized load $nload_c$ is defined as:

$$nload_c = \frac{load_c + load_f}{headend_c(Ethernet)} \tag{5.35}$$

The resulting sub quality factor Q_{device_load} is defined as:

$$Q_{device\_load} = \begin{cases} 1 & when \ nload_c < 1 \\ \dfrac{1}{nload_c} & when \ 1 \le nload_c \le 1 + \dfrac{headend_f(Ethernet)}{headend_c(Ethernet)} \\ 0 & else \end{cases} \tag{5.36}$$

Graphically this sub quality factor is depicted in Figure 5.7.

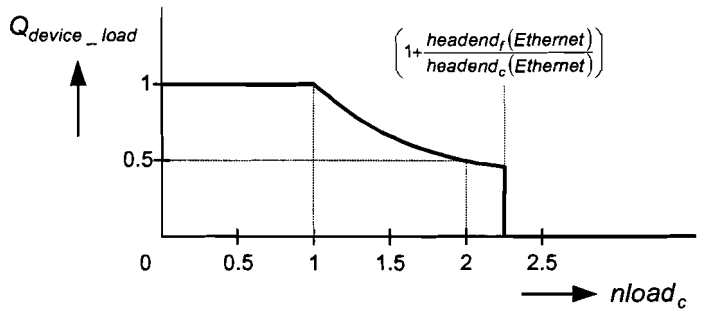

Figure 5.7. Q_{device_load} versus $nload_c$.

Q_{DS_ch}

This sub quality factor describes how many DS channels the candidate has available at a switch that also has DS channels that should be taken over of the failing device connected to it. The ideal situation would be if the candidate has at least the same number of DS channels present at every switch that has DS channels of the failing device connected to it.

The collection of DS channels the candidate has present at a certain switch, is defined by formula 5.19. The collection of DS channels the failing device has present at that same switch is defined by formula 5.20. The number of available DS channels of the candidate is normalized by the number of DS channels needed for the failing device. The quality factor per switch sw_f is defined as:

$$Qch\_DS_{sw_f} = \begin{cases} \dfrac{N(DS_{sw_c})}{N(DS_{sw_f})} & when \quad \dfrac{N(DS_{sw_c})}{N(DS_{sw_f})} \le 1 \\ 1 & else \end{cases} \qquad (5.37)$$

This formula is illustrated in Figure 5.8.

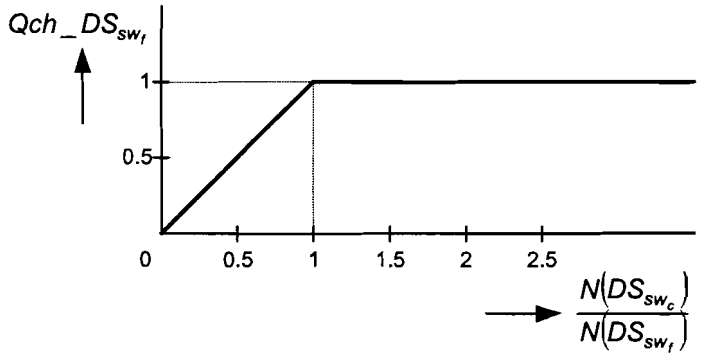

Figure 5.8. Qch_DS versus the normalized number of DS channels available at a certain switch sw_f.

The factor *Qch_DS* has to be calculated for every switch. Mathematically this is defined as:

$$\forall sw_f \in SW_{DS_f} : Qch\_DS_{sw_f} = \begin{cases} \dfrac{N(DS_{sw_c})}{N(DS_{sw_f})} & when \quad \dfrac{N(DS_{sw_c})}{N(DS_{sw_f})} \le 1 \\ 1 & else \end{cases} \qquad (5.38)$$

The resulting sub quality factor should only equal 0 if there is not a single DS channel that can be used. Therefore, the resulting sub quality factor Q_{DS_ch} for all switches is defined as:

$$Q_{DS\_ch} = \frac{\sum_{sw_f \in SW_{DS_f}} Qch\_DS_{sw_f}}{N(SW_{DS_f})} \quad (5.39)$$

This sub quality factor is actually normalized by the number of switches and it is based upon the normalized situation per switch.

Q_{US_ch}
This sub quality factor describes how many US channels the candidate has available at a switch that also has US channels that should be taken over of the failing device connected to it. The derivation of this sub quality factor is the same as the derivation of the sub quality factor Q_{DS_ch} only now applied to US channels. Therefore, only the resulting sub quality factor is showed. This resulting sub quality factor Q_{US_ch} is defined as:

$$Q_{US\_ch} = \frac{\sum_{sw_f \in SW_{US_f}} Qch\_US_{sw_f}}{N(SW_{US_f})} \quad (5.40)$$

Q_{DS_sw}
This sub quality factor describes how many of the available DS channels at a certain switch can actually be used to take over DS channels of the failing device due to restrictions of that switch. This sub quality factor is based upon the information of Table 5.2 where the restriction of the switch are represented in the 4th column. As stated in section 5.4.6 a switch can have two types of restrictions. The first one is that it may be not able to switch all of its input ports to all of it output ports. The second one (and more important one) is that it may be not able to switch two or more output ports to one input port simultaneously.

There are various ways to define a quality factor that represents the restrictions of the HFC matrix switch. We chose to base this sub quality factor (for DS) upon the number of DS channels that the candidate has at a certain switch that actually can be used to take over DS channels of the failing device. This is the best method since the representation of the possible interconnections of the switch does not show what combinations of interconnections are possible[1].

The method to calculate this sub quality factor starts with defining the output ports used by the DS channels of the failing device. This is done by checking the interconnections of the input ports connected to the DS channels of the failing device with the output ports of a switch. Then, the output ports that can be interconnected with the input ports connected to the DS channels of the candidate are defined. Finally, both sets of output ports are compared. The sub quality factor is normalized by the number of output ports used by the failing device.

Mathematically the method described above is defined as follows. The collection of elements *input_port* of a certain switch sw_f that are connected with the DS channels of the failing device is defined as:

$$IN\_F_{sw_f} = \bigcup_{ds \in DS_{sw_f}} ds(switch\_input\_port) \quad (5.41)$$

With:

$$input\_port \in IN\_F_{sw_f} \quad (5.42)$$

[1] If the information present in the 4th column of Table 5.2 is expanded that it does show what combinations of interconnections are possible, the calculation of this sub quality factor can still be done in the same way. The only difference with the presented way is that the number of possible interconnections should be counted, not the number of possible one on one interconnections.

The collection of names of output ports of this same switch sw_f that are interconnected with the input ports in the collection as defined by formula 5.41 is defined as:

$$OUT\_F_{sw_f} = \bigcup_{input\_port \in IN\_F_{sw_f}} input\_port(output\_port\_name) \quad (5.43)$$

The collection of elements *input_port* of that same switch sw_f that are connected to the DS channels of the candidate is defined as:

$$IN\_C_{sw_f} = \bigcup_{ds \in DS_{sw_c}} ds(switch\_input\_port) \quad (5.44)$$

The collection of names of output ports of this same switch sw_f that can be interconnected with the input ports as defined by formula 5.44 is defined as:

$$OUT\_C_{sw_f} = \bigcup_{input\_port \in IN\_C_{sw_f}} input\_port(possible\_output\_ports) \quad (5.45)$$

Finally, the collection of names of output ports of this switch sw_f that are in both collections as defined by formula 5.43 and 5.45 is defined as:

$$OUT_{sw_f} = OUT\_F_{sw_f} \cap OUT\_C_{sw_f} \quad (5.46)$$

The quality factor per switch sw_f is defined as:

$$Qsw\_DS_{sw_f} = \begin{cases} \dfrac{N(OUT_{sw_f})}{N(OUT\_F_{sw_f})} & when \ \dfrac{N(OUT_{sw_f})}{N(OUT\_F_{sw_f})} \leq 1 \\ 1 & else \end{cases} \quad (5.47)$$

This factor can equal 0 at a certain switch. This is the case when not a single output port can be interconnected and thus taken over by the DS channels of the candidate. This factor needs to be calculated for every switch sw_f that has DS channels of the failing device connected to it:

$$\forall sw_f \in SW_{DS_f} : Qsw\_DS_{sw_f} = \begin{cases} \dfrac{N(OUT_{sw_f})}{N(OUT\_F_{sw_f})} & when \ \dfrac{N(OUT_{sw_f})}{N(OUT\_F_{sw_f})} \leq 1 \\ 1 & else \end{cases} \quad (5.48)$$

Finally, the sub quality factor Q_{DS_sw} for all switches is defined as:

$$Q_{DS\_sw} = \frac{\sum_{sw_f \in SW_{DS_f}} Qsw\_DS_{sw_f}}{N(SW_{DS_f})} \quad (5.49)$$

Q_{US_sw}

This sub quality factor describes how many of the available US channels at a certain switch can actually be used to take over US channels of the failing device due to the restrictions of that switch. The derivation of this sub quality factor is the same as the derivation of the sub quality factor Q_{DS_sw}. Therefore, only the resulting sub quality factor Q_{US_sw} is showed, defined as:

$$Q_{US\_sw} = \frac{\sum_{sw_f \in SW_{US_f}} Qsw\_US_{sw_f}}{N(SW_{US_f})} \quad (5.50)$$

Q_{DS_load}

This sub quality factor describes if the DS channels of the candidate have enough bandwidth available to take over the load of the DS channels of the failing device. Since only channels connected to the same switch can take over each other, this sub quality factor needs to be calculated per switch.

The total load of the DS channels of the candidate at a certain switch sw_f is defined as:

$$load\_C_{sw_f} = \sum_{ds \in DS_{sw_c}} ds(load) \quad (5.51)$$

The maximum load these DS channels can handle is defined by:

$$mload\_C_{sw_f} = \sum_{ds \in DS_{sw_c}} ds(maximum\_load) \quad (5.52)$$

Similar, the total load of the DS channels of the failing device at that same switch sw_f is defined as:

$$load\_F_{sw_f} = \sum_{ds \in DS_{sw_f}} ds(load) \quad (5.53)$$

The maximum load of these DS channels of the failing device at that switch sw_f is defined as:

$$mload\_F_{sw_f} = \sum_{ds \in DS_{sw_f}} ds(maximum\_load) \quad (5.54)$$

The ideal situation would be if the following equation holds:

$$load\_F_{sw_f} + load\_C_{sw_f} \leq mload\_C_{sw_f} \quad (5.55)$$

Note that the boundaries are defined as:

$$0 \leq load\_F_{sw_f} + load\_C_{sw_f} \leq mload\_C_{sw_f} + mload\_F_{sw_f} \quad (5.56)$$

Similar to the calculation of the sub quality factor Q_{device_load}, the resulting DS load may become higher than the maximum load the DS channels of the candidate can handle. Again, the reason for this is that the individual bandwidth of the served modems can be decreased. The quality factor per switch is normalized by the maximum DS load the candidate is able to handle. This results in:

$$nload\_C_{sw_f} = \frac{load\_F_{sw_f} + load\_C_{sw_f}}{mload\_C_{sw_f}} \quad (5.57)$$

The quality factor per switch sw_f is defined as:

$$\forall sw_f \in SW_{DS_f} : Qload\_DS_{sw_f} = \begin{cases} 1 & when\ \ nload\_C_{sw_f} < 1 \\ \dfrac{1}{nload\_C_{sw_f}} & when\ \ 1 \leq nload\_C_{sw_f} \leq 1 + \dfrac{mload\_F_{sw_f}}{mload\_C_{sw_f}} \\ 0 & else \end{cases} \quad (5.58)$$

The envelope of formula 5.58 per switch sw_f is similar to the one as depicted in Figure 5.7. The resulting sub quality factor should never equal 0. However, the individual factors per switch sw_f never become equal to 0. Therefore, the resulting sub quality factor Q_{DS_load} for all switches is defined as:

$$Q_{DS\_load} = \prod_{sw_f \in SW_{DSf}} Qload\_DS_{sw_f} \quad (5.59)$$

Q_{US_load}

This sub quality factor describes if the US channels of the candidate have enough bandwidth available to take over the load of the US channels of the failing device. Since only channels connected to the same switch can take over each other, this sub quality factor needs to be calculated per switch. The derivation of this sub quality factor is the same as the derivation of the sub quality factor Q_{US_load}. Therefore, only the resulting sub quality factor Q_{US_load} is showed, defined as:

$$Q_{US\_load} = \prod_{sw_f \in SW_{USf}} Qload\_US_{sw_f} \quad (5.60)$$

Q_{DS_CXN}

This sub quality factor describes if the DS channels of the candidate are able to handle the active modems of the DS channels of the failing device besides their own active modems. With the calculations of the sub quality factors Q_{DS_load} and Q_{US_load} the resulting load was allowed to be higher than the maximum load the channels could handle. The reason for this was that the individual bandwidth of the modems could be decreased after the take over action. With the connections (modems) this is not possible. If the resulting number of modems served is higher than the maximum number of modems that can be served, connections have to be broken. Users that do not use their connection (and thus are inactive), a broken connection will not be noticed. Therefore, the focus lies on the active modems.

The number of active modems served by the DS channels of the candidate at a certain switch sw_f is defined as:

$$cxn\_C_{sw_f} = \sum_{ds \in DS_{sw_c}} ds(active\_connections) \quad (5.61)$$

The maximum number of modems that can be served by these DS channels at that same switch sw_f is defined as:

$$mcxn\_C_{sw_f} = \sum_{ds \in DS_{sw_c}} ds(maximum\_modems) \quad (5.62)$$

Similar, the number of active modems served by the DS channels of the failing device at that same switch sw_f is defined as:

$$cxn\_F_{sw_f} = \sum_{ds \in DS_{sw_f}} ds(active\_connections) \quad (5.63)$$

With the maximum number of active modems these DS channels can serve at that same switch sw_f defined as:

$$mcxn\_F_{sw_f} = \sum_{ds \in DS_{sw_f}} ds(maximum\_modems) \quad (5.64)$$

The ideal situation concerning the number of active modems at a switch sw_f would be:

$$cxn\_C_{sw_f} + cxn\_F_{sw_f} \leq mcxn\_C_{sw_f} \quad (5.65)$$

Note that the boundaries concerning the number of active modems at a switch sw_f are defined as:

$$0 \leq cxn\_C_{sw_f} + cxn\_F_{sw_f} \leq mcxn\_C_{sw_f} + mxcn\_F_{sw_f} \quad (5.66)$$

If the left side of formula 5.65 would become higher than the right side, this means that some of the active modems cannot be served anymore by the DS channels of the candidate after the take over action. Therefore, this is a serious parameter concerning how suitable this candidate is to take over the failing device.

The factor is normalized by the maximum number of modems that can be served by the DS channels of the candidate at a certain switch sw_f. This results in:

$$ncxn\_C_{sw_f} = \frac{cxn\_C_{sw_f} + cxn\_F_{sw_f}}{mcxn\_C_{sw_f}} \quad (5.67)$$

The resulting quality factor per switch sw_f is similar to the sub quality factor Q_{device_load}. However, because the seriousness of too much active modems has to be emphasized, the sloped part of the envelope is squared. For every switch sw_f the resulting factor is defined as:

$$\forall sw_f \in SW_{DS_f} : Qcxn\_DS_{sw_f} = \begin{cases} 1 & when\ ncxn\_C_{sw_f} < 1 \\ \left(\frac{1}{ncxn\_C_{sw_f}}\right)^2 & when\ 1 \le ncxn\_C_{sw_f} \le 1 + \frac{mcxn\_F_{sw_f}}{mcxn\_C_{sw_f}} \\ 0 & else \end{cases} \quad (5.68)$$

Per switch sw_f this factor is depicted in Figure 5.9.

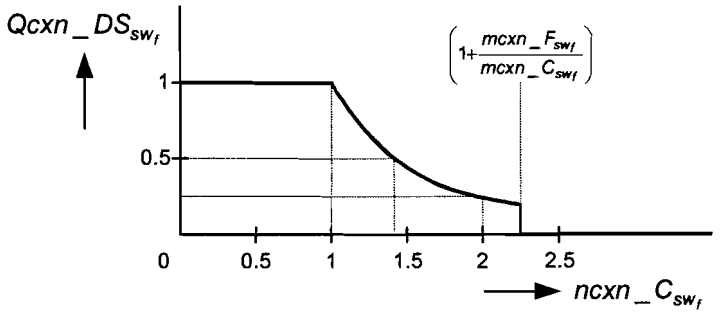

Figure 5.9. $Qcxn\_DS_{sw\_f}$ versus the normalized number of active modems at a certain switch sw_f.

The resulting sub quality factor Q_{DS_CXN} for all switches is defined by:

$$Q_{DS\_CXN} = \prod_{sw_f \in SW_{DS_f}} Qcxn\_DS_{sw_f} \quad (5.69)$$

When an operator finds it unacceptable that the connection of an active user has to be broken after a take over, he could further emphasize this sub quality factor by taking this factor to a certain power in the quality factor of the device. Note that taking the limit of the power of the factor per switch sw_f results in:

$$\lim_{n\to\infty} \left(Qcxn\_DS_{sw_f}\right)^n = \lim_{n\to\infty} \left\{ \begin{cases} 1 \\ \left(\frac{1}{ncxn\_C_{sw_f}}\right)^2 \\ 0 \end{cases} \right)^n = \begin{cases} 1 & when\ ncxn\_C_{sw_f} \le 1 \\ 0 & else \end{cases} \quad (5.70)$$

In other words, the factor per switch only equals 1 if the number of active modems after the take over action is less than the maximum number of modems the DS channels connected to that switch can serve. Substituting this result in formula 5.69 results in a sub quality factor Q_{DS_CXN} that will almost equal 0 in case active connections have to be broken at one of the switches. Since the quality factor of a candidate is just the multiplication of all sub quality factors of that candidate, a sub quality factor that almost equals 0 will have a big influence on the quality factor of that candidate. Therefore, this sub quality factor may be one of the sub quality factors an operator wants to emphasize by giving it a relative weight w_n in the formula of the quality factor of the candidate that is very high.

Q_{US_CXN}
This sub quality factor describes if the US channels of the candidate are able to also handle the number of active modems that are served by the US channels of the failing device. The derivation of this sub quality factor is the same as the derivation of the sub quality factor Q_{DS_CXN}. Therefore, only the resulting sub quality factor Q_{US_CXN} is showed, defined as:

$$Q_{US\_CXN} = \prod_{sw_f \in SW_{US_f}} Qcxn\_US_{sw_f} \qquad (5.71)$$

$Q_{DS\_frequency}$
This sub quality factor describes how many DS channels the candidate has available at a certain switch that are use the same DS output frequency as the DS channels of the failing device. If a DS channel of the failing device has a frequency that is not used by any of the DS channels of the candidate, this particular DS channel cannot be taken over without a reboot of all served modems[1]. Note that this does not mean that the candidate is useless. Only, since modems that were served by the failing device at an unsupported frequency have to reboot, the take over action will result in loss of connections and it will take longer to have them back. This should return in the quality factor of the candidate and that is the reason this sub quality factor is used.

The collection of frequencies used by the DS channels of the failing device at a certain switch sw_f is defined as:

$$F_{sw_f} = \bigcup_{ds \in DS_{sw_f}} ds(frequency) \qquad (5.72)$$

An element of this collection of frequencies at a certain switch sw_f is referred to as *freq*:

$$freq \in F_{sw_f} \qquad (5.73)$$

The collection of elements *ds* of the failing device that use a certain frequency *freq* at a certain switch sw_f is defined as:

$$DS\_F_{freq} = \left\{ ds \in DS_{sw_f} \mid ds(frequency) = freq \right\} \qquad (5.74)$$

The collection of elements *ds* of the candidate that use that same frequency *freq* at the same switch sw_f is defined as:

$$DS\_C_{freq} = \left\{ ds \in DS_{sw_c} \mid ds(frequency) = freq \right\} \qquad (5.75)$$

The quality factor is based upon the number of DS channels the candidate has available at a certain switch sw_f that use a particular frequency *freq*. This quality factor has to be calculated for every frequency *freq* the DS channels of the failing device use at every switch sw_f these channels are connected to. The quality factor is normalized by the number of DS channels with a particular frequency *freq* of the failing device. Mathematically this is defined as:

$$\forall freq \in F_{sw_f} : Q_{freq} = \begin{cases} \dfrac{N(DS\_C_{freq})}{2 \cdot N(DS\_F_{freq})} + 0.5 & when \ \dfrac{N(DS\_C_{freq})}{N(DS\_F_{freq})} \le 1 \\ 1 & else \end{cases} \qquad (5.76)$$

This factor is depicted in Figure 5.10 per frequency *freq* for a certain switch sw_f.

For every switch sw_f the resulting quality factor for all frequencies *freq* at that switch is defined as:

$$\forall sw_f \in SW_{DS_f} : Qf\_DS_{sw_f} = \prod_{freq \in F_{swf}} Q_{freq} \qquad (5.77)$$

[1] When the DS frequencies do not match, the modems have to re-scan their input spectrum. See also section 5.4.3.

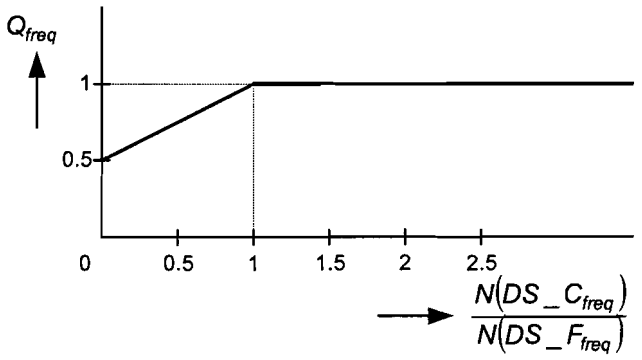

Figure 5.10. Qfreq versus the normalized number of DS channels with a certain frequency freq at a certain switch sw_f.

The resulting sub quality factor $Q_{DS\_frequency}$ is defined as:

$$Q_{DS\_frequency} = \prod_{sw_f \in SW_{DS_f}} Qf\_DS_{sw_f} \qquad (5.78)$$

$Q_{US\_frequency}$

This sub quality factor describes how many US channels the candidate has available at a certain switch that are use the same US input frequency as the US channels of the failing device. If an US channel of the failing device uses a frequency that is not used by any of the US channels of the candidate, this particular US channel cannot be taken over directly. The modems have to be commanded to re-program their US output frequency. The US channel uses the DS channel to do so. However, this requires that the DS frequencies do match. This requirement is already represented by the sub quality factor $Q_{DS\_frequency}$ so this requirement does not need to return in the sub quality factor $Q_{US\_frequency}$. Re-programming the US output frequency of the modems does not require the modems to reboot. Therefore, this sub quality factor should have less influence on the final quality factor of the candidate. Besides this influence, the derivation of this sub quality factor is the same as the derivation of the sub quality factor $Q_{DS\_frequency}$. Therefore, only the result is showed. The resulting sub quality factor $Q_{US\_frequency}$ is defined as:

$$Q_{US\_frequency} = \sqrt{\prod_{sw_f \in SW_{US_f}} Qf\_US_{sw_f}} \qquad (5.79)$$

The square root of the sub quality factor results in the square root of the individual factors per frequency per switch. Graphically the envelope of such a factor is similar to the one depicted in Figure 5.10 only it starts at the square root of 0.5 = 0.707 and then goes to 1.

$Q_{DS\_modulation}$

This sub quality factor describes if the candidate has enough DS channels available that use the same DS modulation type as the DS channels of the failing device. This factor has to be calculated per switch. A DS channel of the failing device that uses a DS modulation type that is not used by any of the DS channels of the candidate, cannot be taken over without a reboot of the modems[1].

The collection of DS modulation types used by the DS channels of the failing device at a certain switch sw_f is defined as:

$$MT_{sw_f} = \bigcup_{ds \in DS_{sw_f}} ds(modulation\_type) \qquad (5.80)$$

An element of this collection of DS modulation types at a certain switch sw_f is referred to as *mt*:

$$mt \in MT_{sw_f} \qquad (5.81)$$

[1] See also section 5.4.3.

The collection of elements *ds* of the failing device that use a certain modulation type *mt* at a certain switch sw_f is defined as:

$$DS\_F_{mt} = \{ds \in DS_{sw_f} \mid ds(modulation\_type) = mt\} \quad (5.82)$$

The collection of elements *ds* of the candidate that use that same modulation type *mt* at the same switch sw_f is defined as:

$$DS\_C_{mt} = \{ds \in DS_{sw_c} \mid ds(modulation\_type) = mt\} \quad (5.83)$$

As stated before, this quality factor has to be calculated per switch sw_f for every modulation type *mt* that is used by a DS channel of the failing device at that switch. The number of available DS channels of the candidate is used to define the quality. The quality factor is normalized by the number of DS channels that need a certain modulation type *mt*. Mathematically this is defined as:

$$\forall mt \in MT_{sw_f} : Q_{mt} = \begin{cases} \frac{N(DS\_C_{mt})}{2 \cdot N(DS\_F_{mt})} + 0.5 & when \ \frac{N(DS\_C_{mt})}{N(DS\_F_{mt})} \le 1 \\ 1 & else \end{cases} \quad (5.84)$$

The envelope of this factor per modulation type *mt* and for a certain switch sw_f is similar to one depicted in Figure 5.10.

For every switch sw_f the resulting quality factor for all DS modulation types *mt* is defined as:

$$\forall sw_f \in SW_{DS_f} : Qmt\_DS_{sw_f} = \prod_{mt \in MT_{sw_f}} Q_{mt} \quad (5.85)$$

The resulting sub quality factor $Q_{DS\_modulation}$ for all switches is defined as:

$$Q_{DS\_modulation} = \prod_{sw_f \in SW_{DS_f}} Qmt\_DS_{sw_f} \quad (5.86)$$

$Q_{US\_modulation}$

This sub quality factor describes if candidate has enough US channels available that use the same US modulation type as the US channels of the failing device. This factor has to be calculated per switch. Unlike the case with the DS modulation type, the modems cannot be commanded to re-program their US modulation type. Therefore, all US channels of the failing device that use a certain US modulation type that is not used by any of the US channels of the candidate, cannot be taken over at all[1]. Therefore, this sub quality factor should have a much bigger influence on the total quality factor of the candidate than the sub quality factor $Q_{DS\_modulation}$. Furthermore, this sub quality factor should become equal to 0 if none of the US channels of the failing device at a certain switch can be taken over. The derivation of this factor differs from the derivation of its DS counterpart. Therefore, the complete derivation is showed.

The collection of US modulation types used by the US channels of the failing device at a certain switch sw_f is defined as:

$$MT_{sw_f} = \bigcup_{us \in US_{sw_f}} us(modulation\_type) \quad (5.87)$$

An element of this collection of US modulation types at a certain switch sw_f is referred to as *mt*:

$$mt \in MT_{sw_f} \quad (5.88)$$

The collection of elements *us* of the failing device that use a certain modulation type *mt* at a certain switch sw_f is defined as:

$$US\_F_{mt} = \{us \in US_{sw_f} \mid us(modulation\_type) = mt\} \quad (5.89)$$

[1] See also section 5.4.3 and 5.4.6.

The collection of elements *us* of the candidate that use that same modulation type *mt* at the same switch sw_f is defined as:

$$US\_C_{mt} = \{us \in US_{sw_c} \mid us(modulation\_type) = mt\} \quad (5.90)$$

The number of available US channels of the candidate is used to define the quality. This quality factor has to be calculated per switch sw_f for every modulation type *mt* that is used by an US channel of the failing device at that switch. The quality factor is normalized by the number of US channels that need a certain modulation type *mt*. Mathematically this is defined as:

$$\forall mt \in MT_{sw_f} : Q_{mt} = \begin{cases} \frac{N(US\_C_{mt})}{N(US\_F_{mt})} & when \ \frac{N(US\_C_{mt})}{N(US\_F_{mt})} \leq 1 \\ 1 & else \end{cases} \quad (5.91)$$

Graphically, the envelope of this factor for a certain modulation type *mt* at a certain switch sw_f is similar to one depicted in Figure 5.8. Note that indeed this quality factor per US modulation type *mt* per switch sw_f can become equal to 0.

For every switch sw_f the resulting quality factor for all US modulation types *mt* is defined as:

$$\forall sw_f \in SW_{US_f} : Qmt\_US_{sw_f} = \frac{\sum_{mt \in MT_{sw_f}} Q_{mt}}{N(MT_{sw_f})} \quad (5.92)$$

The resulting sub quality factor $Q_{US\_modulation}$ for all switches is defined as:

$$Q_{US\_modulation} = \frac{\sum_{sw_f \in SW_{US_f}} Qmt\_US_{sw_f}}{N(SW_{US_f})} \quad (5.93)$$

Q_{DS_symbol}

This sub quality factor describes if the candidate has enough DS channels available that use the same DS symbol rate as the DS channels of the failing device. This factor has to be calculated per switch. Since the modems are not able to re-program their DS symbol rates[1], this factor should have a serious influence on the total quality factor of the candidate. Also, a DS channel that uses a certain symbol rate that is not used by any of the DS channels of the candidate, cannot be taken over. In this case, even a reboot of the modems does not change this. Therefore, this quality factor should become equal to 0 in this situation (similar to the sub quality factor $Q_{US\_modulation}$)

The collection of DS symbol rates used by the DS channels of the failing device at a certain switch sw_f is defined as:

$$SR_{sw_f} = \bigcup_{ds \in DS_{sw_f}} ds(symbol\_rate) \quad (5.94)$$

An element of this collection of DS symbol rates at a certain switch sw_f is referred to as *sr*:

$$sr \in SR_{sw_f} \quad (5.95)$$

The collection of elements *ds* of the failing device that use a certain symbol rate *sr* at a certain switch sw_f is defined as:

$$DS\_F_{sr} = \{ds \in DS_{sw_f} \mid ds(symbol\_rate) = sr\} \quad (5.96)$$

[1] See also section 5.4.3 and 5.4.6.

The collection of elements *ds* of the candidate that use that same symbol rate *sr* at the same switch sw_f is defined as:

$$DS\_C_{sr} = \{ds \in DS_{sw_c} \mid ds(symbol\_rate) = sr\} \quad (5.97)$$

The quality factor is based upon the number of DS channels of the candidate that are available that use a certain symbol rate *sr* at a certain switch sw_f. This factor needs to be calculated for every DS symbol rate *sr* and for every switch sw_f. The factor is normalized by the number of DS channels of the failing device that use a certain DS symbol rate. For every symbol rate *sr* at a certain switch sw_f the factor is defined as:

$$\forall sr \in SR_{sw_f} : Q_{sr} = \begin{cases} \dfrac{N(DS\_C_{sr})}{N(DS\_F_{sr})} & when \quad \dfrac{N(DS\_C_{sr})}{N(DS\_F_{sr})} \leq 1 \\ 1 & else \end{cases} \quad (5.98)$$

Graphically, the envelope of this factor for a certain symbol rate *sr* at a certain switch sw_f is similar to one depicted in Figure 5.8. Note that indeed this quality factor can become equal to 0.

For every switch sw_f the resulting quality factor for all DS symbol rates *sr* is defined as:

$$\forall sw_f \in SW_{DS_f} : Qsr\_DS_{sw_f} = \frac{\sum_{sr \in SR_{sw_f}} Q_{sr}}{N(SR_{sw_f})} \quad (5.99)$$

The resulting sub quality factor Q_{DS_symbol} for all switches is defined as:

$$Q_{DS\_symbol} = \frac{\sum_{sw_f \in SW_{DS_f}} Qsr\_DS_{sw_f}}{N(SW_{DS_f})} \quad (5.100)$$

Q_{US_symbol}

This sub quality factor describes if the candidate has enough US channels available that use the same US symbol rate as the US channels of the failing device. This factor has to be calculated per switch. If an US channel of the failing device uses an US symbol rate that is not used by any of the US channels of the candidate, then this US channel cannot be taken over directly. The modems need to be commanded to re-program their US symbol rate. Therefore, the DS connections with the modems have to be available[1]. This factor does not have the same big influence on the final quality factor of the candidate as the sub quality factor Q_{DS_symbol}. The reason for this is that a modem can still be taken over if the condition that the DS connection is available, holds. The derivation of this sub quality factor differs from the derivation of the sub quality factor Q_{DS_symbol}. Therefore, the complete derivation is presented.

The collection of US symbol rates used by the US channels of the failing device at a certain switch sw_f is defined as:

$$SR_{sw_f} = \bigcup_{us \in US_{sw_f}} us(symbol\_rate) \quad (5.101)$$

An element of this collection of US symbol rates at a certain switch sw_f is referred to as *sr*:

$$sr \in SR_{sw_f} \quad (5.102)$$

The collection of elements *us* of the failing device that use a certain symbol rate *sr* at a certain switch sw_f is defined as:

$$US\_F_{sr} = \{us \in US_{sw_f} \mid us(symbol\_rate) = sr\} \quad (5.103)$$

[1] See also section 5.4.3.

The collection of elements *us* of the candidate that use that same symbol rate *sr* at the same switch sw_f is defined as:

$$US\_F_{sr} = \{us \in US_{sw_c} \mid us(symbol\_rate) = sr\} \quad (5.104)$$

As with the calculation of the sub quality factor Q_{DS_symbol} this quality factor is based upon the number of US channels of the candidate that are available that use a certain symbol rate. This factor needs to be calculated for every US symbol rate *sr* and for every switch sw_f. The factor is normalized by the number of US channels of the failing device that use a certain US symbol rate For every symbol rate *sr* at a certain switch sw_f the factor is defined as:

$$\forall sr \in SR_{sw_f} : Q_{sr} = \begin{cases} \frac{N(US\_C_{sr})}{4 \cdot N(US\_F_{sr})} + 0.75 & when \ \frac{N(US\_C_{sr})}{N(US\_F_{sr})} \le 1 \\ 1 & else \end{cases} \quad (5.105)$$

Graphically, the envelope of this factor for a certain symbol rate *sr* at a certain switch sw_f is similar to one depicted in Figure 5.10. The only difference is that the envelope of this factor starts at 0.75 and then goes to 1.

For every switch sw_f the resulting quality factor for all US symbol rates *sr* is defined as:

$$\forall sw_f \in SW_{US_f} : Qsr\_US_{sw_f} = \prod_{sr \in SR_{sw_f}} Q_{sr} \quad (5.106)$$

The resulting sub quality factor Q_{US_symbol} for all switches is defined as:

$$Q_{US\_symbol} = \prod_{sw_f \in SW_{US_f}} Qsr\_US_{sw_f} \quad (5.107)$$

Q_{DS_power}

This sub quality factor describes if the DS channels of the candidate have enough output power to serve the DS areas that were served by the DS channels of the failing device. In order to be able to serve such a DS area, the DS power level of a DS channel may differ ±10 dB compared to the power level of the DS channel of the failing device[1]. This factor has to be calculated per switch.

The collection of different output power levels of the DS channels of the failing device at a certain switch sw_f is defined as:

$$PW_{sw_f} = \bigcup_{ds \in DS_{sw_f}} ds(power\_level) \quad (5.108)$$

An element of this collection of power levels at a certain switch sw_f is referred to as pw_{ds}:

$$pw_{ds} \in PW_{sw_f} \quad (5.109)$$

The collection of elements *ds* of the failing device that use a certain power level pw_{ds} at a certain switch sw_f is defined as:

$$DS\_F_{pw_{ds}} = \{ds \in DS_{sw_f} \mid ds(power\_level) = pw_{ds}\} \quad (5.110)$$

As stated before, the power level of a DS channel of the candidate may differ ±10 dB. Therefore, the collection of elements *ds* of the candidate that have the same power level pw_{ds} at the same switch sw_f is defined as:

$$DS\_C_{pw_{ds}} = \{ds \in DS_{sw_c} \mid ds(power\_level) = pw_{ds} \pm 10\} \quad (5.111)$$

The quality factor is based upon the number of available DS channels that are within ±10 dB range of the power level of a DS channel of the failing device. This factor has to be calculated per switch sw_f.

[1] See also section 5.4.3 and reference [1].

This factor is normalized by the total number of DS channels of the failing device that use this particular power level pw_{ds}. The resulting quality factor for all DS power levels pw_{ds} per switch sw_f is defined as:

$$\forall pw_{ds} \in PW_{sw_f} : Q_{pw_{ds}} = \begin{cases} \frac{N(DS\_C_{pw_{ds}})}{N(DS\_F_{pw_{ds}})} & when \ \frac{N(DS\_C_{pw_{ds}})}{N(DS\_F_{pw_{ds}})} \leq 1 \\ 1 & else \end{cases} \quad (5.112)$$

Note that this factor per power level pw_{ds} at a certain switch sw_f can equal 0. This happens when the power levels pw_{ds} of the DS channels of the candidate are all out of the ±10 dB range. The quality factor per switch sw_f is defined as :

$$\forall sw_f \in SW_{DS_f} : Qpw\_DS_{sw_f} = \frac{\sum Q_{pw_{ds}}}{N(PW_{sw_f})} \quad (5.113)$$

The resulting sub quality factor Q_{DS_power} for all switches is defined as:

$$Q_{DS\_power} = \frac{\sum Qpw\_DS_{sw_f}}{N(SW_{DS_f})} \quad (5.114)$$

Q_{US_delay}

This sub quality factor describes if the number of US channels of the failing device that can be taken over, is restricted by relative time delay of the different US channels. If an US channel already serves an US area, the relative time delay cannot be adjusted[1]. If the US channel is redundant, the relative time delay can be programmed by the redundancy system. This factor has to be calculated per switch.

The collection of different relative time delays of the US channels of the failing device at a certain switch sw_f is defined as:

$$RTD_{sw_f} = \bigcup_{us \in US_{sw_f}} us(relative\_time\_delay) \quad (5.115)$$

An element of this collection of relative time delays at a certain switch sw_f is referred to as rtd_{us}:

$$rtd_{us} \in RTD_{sw_f} \quad (5.116)$$

The collection of elements *us* of the failing device that use a certain relative time delay rtd_{us} at a certain switch sw_f is defined as:

$$US\_F_{rtd_{us}} = \{ us \in US_{sw_f} \mid us(relative\_time\_delay) = rtd_{us} \} \quad (5.117)$$

The relative time delay of the US channels of the candidate, may differ ±1.5 timeslots compared with the relative time delay of the US channels of the failing device [2]. Therefore, the collection of elements *us* of the candidate that use that same relative time delay rtd_{us} at the same switch sw_f is defined as:

$$US\_C_{rtd_{us}} = \{ us \in US_{sw_c} \mid us(relative\_time\_delay) = rtd_{us} \pm 1.5 \} \quad (5.118)$$

The quality factor is based upon the number of available DS channels that are within ±1.5 range of the relative time delay of the US channels of the failing device. This factor has to be calculated for every switch sw_f. The quality factor is normalized by the number of US channels with a certain relative time delay rtd_{us} of the failing device.

[1] See also section 5.4.3 and reference [2].

Mathematically this factor is defined as:

$$\forall rtd_{us} \in RTD_{sw_f} : Q_{rtd_{us}} = \begin{cases} \frac{N(US\_C_{rtd_{us}})}{N(US\_F_{rtd_{us}})} & when \quad \frac{N(US\_C_{rtd_{us}})}{N(US\_F_{rtd_{us}})} \leq 1 \\ 1 & else \end{cases} \qquad (5.119)$$

This resulting factor is can become equal to 0. However, this should not mean that the factor per switch sw_f also should become equal to 0. Therefore, the resulting factor for all relative time delays rtd_{us} per switch sw_f is defined as:

$$\forall sw_f \in SW_{US_f} : Qrtd\_US_{sw_f} = \frac{\sum_{rtd_{us} \in RTD_{sw_f}} Q_{rtd_{us}}}{N(RTD_{sw_f})} \qquad (5.120)$$

The resulting sub quality factor Q_{US_delay} is defined as:

$$Q_{US\_delay} = \frac{\sum_{sw_f \in SW_{US_f}} Qrtd\_US_{sw_f}}{N(SW_{US_f})} \qquad (5.121)$$

This sub quality factor equals 0 only if all US channels of the candidate cannot take over US channels of the failing device due to a relative time delay that is out of the ±1.5 timeslot range. In this case, this candidate is not able to take over the failing device. This means that the total quality factor of the candidate should equal 0 and this is indeed the case if this sub quality factor equals 0.

5.5.4 Calculation of the quality factor

With the sub quality factors describing the different parameters of the candidate as defined in the previous section, the total quality factor of this candidate can be defined. Using formula 5.36, 5.39, 5.40, 5.49, 5.50, 5.59, 5.60, 5.69, 5.71, 5.78, 5.79, 5.86, 5.93, 5.100, 5.107, 5.114 and 5.121 the final quality factor of a candidate Q_{device} is defined as:

$$\begin{aligned} Q_{device} = \prod_n (Q_n)^{w_n} = &(Q_{device\_load})^{w_1} \cdot (Q_{DS\_ch})^{w_2} \cdot (Q_{US\_ch})^{w_3} \cdot (Q_{DS\_sw})^{w_4} \cdot (Q_{US\_sw})^{w_5} \cdot \_ \\ &\_(Q_{DS\_load})^{w_6} \cdot (Q_{US\_load})^{w_7} \cdot (Q_{DS\_CXN})^{w_8} \cdot (Q_{US\_CXN})^{w_9} \cdot \_ \\ &\_(Q_{DS\_frequency})^{w_{10}} \cdot (Q_{US\_frequency})^{w_{11}} \cdot (Q_{DS\_modulation})^{w_{12}} \cdot (Q_{US\_modulation})^{w_{13}} \cdot \_ \\ &\_(Q_{DS\_symbol})^{w_{14}} \cdot (Q_{US\_symbol})^{w_{15}} \cdot (Q_{DS\_power})^{w_{16}} \cdot (Q_{US\_delay})^{w_{17}} \end{aligned} \qquad (5.122)$$

And, as stated in section 5.5.2 with the weight w_n defined as:

$$w_n \in [0, \infty) \qquad (5.123)$$

The way the sub quality factors are defined in the previous section implies that all relative weights w_n equal 1. However, if an operator wants to adjust the relative influence of a sub factor on the total quality factor of the device, all he has to do is to adjust the weights. Note that the values of the weights w_n should be in the configuration files of the redundancy system.

The take over decision function needs to calculate the Q_{device} factor for every device that is part of the redundancy network in order to be able to decide which device is the best candidate to take over the failing device. The results of the (sub) calculations are stored by the take over decision function. The reason for this is that the take over decision function triggers the take over activating function. It provides this function with the following things:

- The name of the failing device. With this name the take over activating function can get the information it needs from the locally stored information.

- The name of the device that has to take over the failing device. This is the candidate that has the highest quality factor Q_{device}.
- The name of the backup device to take over the failing device. In case the preferred device to take over the failing device cannot do so, there is one backup that can be commanded to take over the failing device.
- The results of the (sub) calculations of both devices that can take over the failing device. This is provided to give the take over activating function of the device that is actually going to take over the failing device for example, some information over the load it should expect after the take over action. This information is used to configure the processes that can decrease the individual bandwidths of the modems.

5.5.5 Remarks regarding the take over decision function

As stated in the previous section, all results of the (sub) calculations should be stored. In practice, it is also possible to just keep the results of the devices that until that moment have the highest quality factor Q_{device}.

In order to speed up the calculations every device could calculate the sub quality factors that are based upon static information in advance. In that case, for every device that could possibly fail the take over decision function of every other device already has the best candidate to take over this failing device present, based upon just static information. However, when the number of devices that participate in the redundancy network increases, this method will result in lots of calculations and thus information that has to be stored.

Another way to speed up the calculations, is to define a threshold value of, for example $Q_{device} = 0.7$. In case the take over decision function calculates a certain Q_{device} that is above this threshold, the functions stops and decides that this particular device is the best candidate to take over the failing device. This threshold value can be based upon previous achieved quality factors Q_{device}.

Furthermore, the input set of candidates could be sorted in such way that devices that have the highest possibility that they can take over another device, are offered first. An operator can base this possibility to take over another device on the configuration of a device. Common configuration are highly compatible and therefore better able to take over each other than a device that for example serves an area that is very far away. Together with a threshold value, the chance that the take over decision system finds a suitable device to take over a failing device in the first few candidates becomes very high. Especially when complete redundant devices are available; these should always be offered first. When redundant devices are connected to the correct switches and contain enough DS and US channels, they may get a Q_{device} that is at least near 1, perhaps even equal to 1.

Note that if a value is not specified in the tables handled by the information sharing function, this value can be any value. A completely redundant device should for example, not have been programmed to use a certain DS power level or a certain DS output frequency. In this way, it can be used for (almost) every DS power level and DS output frequency. Therefore, it may get a very high Q_{device} factor. If this device were to be the best candidate, the take over activating function can program the US and or DS channels of this device in such way that all parameters match the parameters of the US and DS channels of the failing device.

Finally, the method described in this section uses quality factors to make a distinction between the candidates to take over a failing device. Therefore, the resulting quality factor Q_{device} of a candidate should not be seen as a factor that literally describes how ideal a candidate is to take over the device. In fact, it should be seen as how ideal a candidate is compared with the other candidates to take over a failing device. When one wants to bring the resulting quality factor Q_{device} somewhat more to the domain of how ideal it is, the following formula could be used:

$$\%\ ideal\ to\ take\ over = 100\% \cdot \sqrt[n]{Q_{device}} \qquad (5.124)$$

Where n represents the number of sub quality factors used for the calculation of the quality factor Q_{device}. In this case (when using formula 5.122) n would equal 17.

5.6 The take over activating function

5.6.1 Introduction

This function of the redundancy system, is the function that actually performs the take over action. The take over activating functions of different redundancy systems use take over activating messages to interact. The device that has been chosen by the take over decision function as the preferred device to take over the failing device first is notified by the take over activating function. Furthermore, this preferred device is provided with all necessary information of the failing device. The take-over activating function of the preferred device then prepares the take over action, based upon the information it has received. Afterwards, the take over action itself is carried out. Finally, some post processing is done.

The take over activating function depends on the information sharing function and the take over decision function[1] for a correct working. In this section first the basis of the take over activating function is described. Then, the take over activating messages are discussed. An example of the take over activating function is presented in chapter 6.

5.6.2 Basis of the take over activating function

The take over activating function is triggered by the take over decision function of the own redundancy system. It is provided with the information as presented in section 5.5.4:

- The name of the failing device.
- The name of the preferred device to take over this failing device.
- The name of the backup device to take over the failing device.
- Results of the (sub) calculations of the two devices that can be used to take over the failing device.

First, the take over decision function checks if it itself is the preferred device to take over the failing device. If so, it can continue with the preparation of the actual take over action. If not, it notifies the take over activating function of the device that is the preferred device to take over the failing device. This is done using a take over activating message. This message contains the information passed on to the take over activating function by the take over decision function. After sending the notification message, the take over activating function notifies the information sharing function that it has to broadcast information sharing messages with the message type set to *take_over_information*. Note that the information sharing function has to send messages with all the information[2].

When the take over activating function of the preferred device receives a notification that it has to take over the failing device, it forces the redundancy system to enter its alarm state. Furthermore it notifies the information sharing function that it expects 7 information sharing messages. Each message has a table included concerning the failing device. This table is one of Table 5.2 up to and including Table 5.8. When the information sharing function has received these messages it again notifies the take over activating function that all information needed is available. The take over activating function reacts by sending an acknowledge to take over activating function of the device that has detected the failing device, that it is ready to start with the preparation of the actual take over action.

The take over function of the device that detected the failing device now starts waiting for a message of the preferred device that the take over action has been performed successfully. If it does not

[1] See also section 5.4 and 5.5.

[2] The reason for this is that when the failing device has been detected, it has been removed from the detection ring by the detection function. This has resulted in *failing_device_clearup* messages broadcasted by the information sharing function. Therefore, none of the other devices still has information of the failing device. See also section 5.4.

receive such a message within a certain time interval, it notifies the preferred device again. The reason for this is that the actual take over action could just have not be finished yet. If again it gets no reaction, it concludes that something has gone wrong at the preferred device. Therefore, it restarts its procedure, but now with the backup device to take over the failing device. If this take over action would fail too, the management system is notified that the take over action could not be performed successfully. The redundancy system then returns to the running state.

The preparation of the actual take over action includes the following actions:

- Process the input information, originating from the take over decision function that decided this was the best candidate to take over the failing device. This information includes the number (and names) of channels that have to be taken over. Furthermore, it includes the names of the HFC matrix switches together with the input ports and output ports that have to be used in this take over action. Finally, it also includes the expected load after the take over action, both for the complete device and for the individual DS and US channels. This information has to be used for the process that limits the individual bandwidth of the modems.
- Process the information provided by the information sharing function. This information consists of all table entries concerning the failing device.
- Generate the messages to re-program the HFC matrix switches.
- Generate the messages to remove all modems served by the failing device from the routing tables of the router. These messages are based upon the RIP protocol [21].
- Generate the tables with the names and service levels of all modems that were served by the failing device. This table is used when such a modem signs on after the take over action, for it should always get a connection with the CoS level it had before the take over action.

When this preparation is finished, the actual take over action is performed. First, the messages to the switches and the router are sent. At that same time, the current DS traffic is frozen. This gives the HFC matrix switches the time to change the interconnections. After a few seconds[1], the HFC matrix switches are ready and the normal DS traffic is continued. Finally, a DVB broadcast is performed to inform the modems that were served by failing device with the new information concerning their US parameters. Note that this last action is only necessary when DS communication with the modems is possible. Otherwise, these modems will reboot themselves and perform a sign on procedure. After the broadcast, a take over activating message is sent to the take over activating function of the device that detected the failing device. This ends the alarm state at that device, for it knows that the take over action has been carried out successfully. The final action of that take over activating function is to inform the information sharing function that the local information of the failing device can now be cleared. Then the redundancy system returns to the running state.

At the device that actually has taken over the failing device, the take over activating function continues with the post processing. This post processing includes overruling the normal sign on procedure for the modems that were served by the failing device. After a time interval *take_over_time* the take over activating function ends its task. The modems that were served by the failing device that still have not performed the sign on procedure, are cleared from the table. Then, the take over activating function informs the information sharing function with the new situation concerning the static information of the device[2]. This also informs the information sharing function to clear the information used for the take over action. Finally, the redundancy system returns to the running state.

The fact that the modems that still have not performed the sign on procedure are clear from the table means that now they perhaps cannot get their original connection with the CoS level they had, back. However, this escape for the take over activating function is necessary because otherwise the redundancy system of this device stays in the alarm state. The time interval *take_over_time* has to be configured by the operator and it has to be based upon practical information on the time it takes to have hundreds of modems signed on again.

[1] Perhaps even ms; this depends on the HFC matrix switches used. This should be based upon partical information.
[2] Note that this cannot be seen by the device itself because this type of delta information does not originate from eighter the operator or the normal software of the device.

Note that all information changes after the actual take over action, is handled as normal delta information of this device. Therefore, new modems (and thus new Act information) is broadcasted by the information sharing function as normally would be done.

5.6.3 The take over activating message

The take over activating function uses take over activating messages to interact with the take over activating functions of other redundancy systems. There are three message types: *take_over, ack_take_over* and *take_over_ready.* These different message types are described below.

take_over
This message type is used to notify the take over activating function of the redundancy system of the preferred device that it should enter the alarm state and start its take over procedure. This message is sent by the take over activating function of the device that performed the take over decision task. It is a unicast message and besides the message type *take_over* and a frame number, it includes the information provided by the take over decision function[1].

ack_take_over
This message type is used to notify the take over activating function of the device that performed the take over decision task, that all information needed has been received and that the actual take over procedure is started. This is a unicast message. Besides this message type, the message only includes the name of the failing device and a frame number.

take_over_ready
This message type is used to notify the take over activating function of the device that performed the take over decision task, that the actual take over action has been performed. This is a unicast message and besides the message type it only includes the name of the failing device and a frame number.

When the *take_over_ready* message is not received by the take over activating function of the device that performed the take over decision task, it again notifies the preferred device with a *take_over* message. If this device is still performing the actual take over task, it is able to react and it does so with an *ack_take_over* message. This instructs the other take over activating function to restart the waiting for the *take_over_ready* message. If the *ack_take_over* message is not received, it restarts its procedure with the backup device to take over the failing device.

5.6.4 Remarks regarding the take over activating function

The processing of the information that is used for the actual take over action needs to be discussed some more. When no parameters have to be changed and when every DS channel and every US channel of the failing device can be taken over one on one, the take over action will be very simple. However, if for example multiple DS areas have to be taken over by one DS channel of the preferred device, some logic may be required for the processing of the information. The reason for this is that the tables have to be processed and the take over activating function has to recognize for example which DS channel it has to use for which DS area that was served by the failing device. This is not as easy at it may seem. In practice this processing may be speeded up by providing more information generated by the take over decision function. This practical implementation of the logic remains an item to be done.

[1] See also section 5.5.5 and 5.6.2.

5.7 The states of the redundancy system

5.7.1 Introduction

In the previous sections the function view of the redundancy system has been described. In this section returns to the state view of the redundancy system. Since all functions together form the complete redundancy system, these functions are used to describe the actions that occur in every state of the redundancy system. First the initialization state and the shutdown state are discussed. Then the running state is described. Note that this state is already almost completely described in the sections 5.3 and 5.4. Finally, the alarm state is described.

5.7.2 The initialization state

The initialization state of the redundancy system is the state during which the redundancy system is included in the redundancy network. The detection function and the information sharing function work together during this state to include this redundancy system.

When a device is up and running, the redundancy system is started. First the configuration files are processed. These configuration files are defined by the operator and include the following information:

- The initial value for *max_roundtime*[1] and *timeout*.
- The value of the time interval the redundancy system has to wait for the initial *alive* message before it is allowed to conclude that it is the only device around.
- The static information tables as defined in section 5.4.4.
- The multicast IP address that has to be used by the information sharing function[2].
- The weights w_n of the sub quality factors used by the take over decision function.
- The value for *take_over_time* used by the take over activating function.
- The IP address of the router used by the take over activating function.

Then, the redundancy system waits a certain random time interval. After this time interval the information sharing function broadcasts an information sharing message with the message type set to *device_change* and the IP address of the message set to its own IP address. This notifies the other devices that there is a device that wants to be added to or removed from the redundancy network. The reason it waits for a random time interval before broadcasting this *device_change* message is the following. If it would send this message immediately and other devices are starting their redundancy systems at the same time, lots of these messages would be broadcasted and this could result in a chaotic situation.

The information sharing systems of all devices present in the redundancy network receive this message and pass it on to their detection functions. The detection function of the redundancy system that is waiting to send the *alive* detection message to the next device in the detection ring reacts. First it will check if this IP address is already present in the detection ring. If not, it is a new device that wants to be added to the detection ring. It replies to the device with the new *alive* message. This new *alive* message has the new device inserted between the device that reacts and the device that used to be the next device in the detection ring.

Also, it will command the information sharing system to provide the new device with the information needed for the take over decision function and the take over activating function. This results in *detection_ring_change* messages with all the Dec information thus the information represented by Table 5.1 up to and including Table 5.7. Furthermore a *detection_ring_change* message is broadcasted with the Act information of the device that now is preceded by the new device. This is the information represented by Table 5.8. Finally, a *detection_ring_change* message is broadcasted to get the Act information of the new device.

[1] This value is only used when this device is the device that starts the detection ring.
[2] In case broadcast is used instead of multicast, this field is set to 255.255.255.255.

When the new device does not receive the *alive* message within a certain time interval, it concludes that there is no other device present. In this case it will remain in the initialization state until it receives a *device_change* information sharing message. Such a message would imply that now there is at least one other device present so a detection ring can be set up. In this case it reacts as if it is the device that is waiting to send an *alive* message, which is in fact the case.

When the device does receive the *alive* message, this message will be handled by the detection function, as it would normally do. Furthermore, the information sharing function will handle the *detection_ring_change* messages, as it would normally do. Since there is delta information concerning the own device (first there was no information and now it is up and running) it will broadcast this delta information. Note that one of the *detection_ring_change* messages is a request for its Act information. This message is handles as normally would be done so this results in a broadcast of another *detection_ring_change* message, now with the requested Act information included.

The previous defined actions are in fact, just normal actions of the detection function and the information sharing function. The only difference is that normally they work on their own and during this state they work together. Note that this also happens in case of a failing device.

An example of the initialization state is presented in chapter 6.

5.7.3 The shutdown state

The shutdown state of the redundancy system is used to provide the operator a proper way to remove a device from the redundancy network. Similar to the initialization state, the detection function and the information sharing function work together during this state.

When the shutdown state is initiated by the operator, the shutdown state starts with the broadcast of an information sharing message by the information sharing system. This message has the message type set to *delta_information* and all information is set to 'to be removed'. Since there are 8 data types (8 tables), 8 messages need to be sent. When the information sharing functions of the other redundancy systems have processed these message, they do not have information of this device anymore and therefore, this device cannot be used as a candidate for take over actions anymore[1].

Then, the device has to wait for an *alive* message to continue with the shutdown state. Note that this only takes a few seconds worst case. When the detection function receives such a message, it removes its own IP address from the detection ring. Then it continues the detection ring with the next device. Furthermore, it replies to the initial *alive* message with an *ack_alive* message. This message also contains the new detection ring. At that same time, it triggers the information sharing function to broadcast a *detection_ring_change* message with the Act information of the next device in the detection ring. This broadcast is received by the information sharing function of the device that used to precede the device in its shutdown state in the detection ring. Since, the information is about the device it precedes in the new detection ring, it processes the message and stores the Act information.

Finally the device in its shutdown state has to wait for the *ack_alive* message of the next device in the old detection ring. If it does not receive such a message, it should first handle this failing device, as it would normally do[2]. Otherwise, this *ack_alive* message ends the shutdown state.

With the redundancy system is down, the operator is free to continue the normal shutdown procedure of the complete device.

[1] In case a take over decision function is performing its task, this device is already included in the set of possible candidates and therefore, it could become the preferred device. This is why the take over decision function provides a backup device to take over a failing device.

[2] Note that since it already broadcasted the *delta_information* messages with its own information set to 'to be removed', the take over decision function will not use its own device as a possible candidate to take over the failing device.

There is another way to remove a device from the redundancy network that is faster but less proper. The operator can just shut down the device, but this will initiate the alarm state at the device that precedes this device in the detection ring. However, if the operator first adjusts the redundancy system configuration on the device he wants to shut down, everything works out well. All he has to do is adjust the following parameters:

- The 'available for take over' setting of every DS and US channel of the device has to be set to false. This excludes the device as a candidate for take over actions in case a failing device is detection somewhere in the detection ring.
- The 'to be taken over' setting of every DS and US channel of the device also has to be set to false. In this way the device will not be taken over.

Then, he is allowed to shut down the device. The device that precedes this device in the detection ring will still detect the 'failure'. Therefore, it will remove this device from the detection ring. Note that with the proper shutdown procedure, this is done by the device that is in its shutdown state itself. Furthermore, the information sharing system will broadcast information sharing messages with the message type set to *failing_device_clear-up*. These information sharing messages have the same effect as the *delta_information* messages with the information set to 'to be removed'. The take over decision system will notice that none of the US and DS channels of the device have to be taken over and therefore, no further action will take place. In this way the device is also removed from the network, only it is not a proper way; the work is done by the other devices.

The shutdown state can also be used to shut down a device but have the modems served by that device, taken over. To do so, the operator has to initiate the shutdown procedure as normal. Only this time when this device waits for the *alive* message to continue with its removal from the detection ring, it has to start the take over decision function with its own device as the device to be taken over. The rest of the procedure is exactly the same. After some processing, the take over decision function will trigger the take over activating function. This will start the normal take over activating procedure. At the end of this procedure, the device that should be shut down, has been taken over.

An example of the shutdown state is presented in chapter 6.

5.7.4 The running state

The running state of the redundancy system is the normal operation mode of this system. During this state the detection function and the information sharing function work independently. Therefore, this complete state has been described with the individual functions as described in section 5.3 and 5.4. An example of the running state will be presented in chapter 6.

5.7.5 The alarm state

The alarm state of the redundancy system is entered when a failing device has been detected. During this state the take over decision function and the take over activating function are active. Also, during this state the detection function removes the failing device from the detection ring and the information sharing system commands to clear the information all other redundancy systems have stored of this failing device. During this state the various function interact with each other. This interaction has already been described in sections 5.3, 5.4, 5.5 and 5.6. In chapter 6 an example of this interaction is presented.

5.8 Intra head-end redundancy

5.8.1 Introduction

In the previous sections the complete redundancy system for inter head-end redundancy has been described. During these sections there has already been referred to intra head-end redundancy but the mayor part of this form of redundancy still has to be defined. However, since the redundancy systems for intra head-end redundancy and inter head-end redundancy are similar, only the differences need to be described. In this section these differences are discussed per function of the redundancy system.

For inter head-end redundancy, the System Controller is the device that has the redundancy system running. The System Controller is the board that participates in the inter head-end redundancy network on behalf of the head-end. For intra head-end redundancy the network consists of all boards connected to the central communication bus of the head-end. For a CableDock 200 this bus is the CompactPCI bus. Only boards with processing power can be used. For example, the power supply is not able to run software and therefore, it cannot participate in the intra head-end redundancy network. Each board that participates in the network has an intra-head redundancy system running. Note that this means that the System Controller has two redundancy systems running: one for inter head-end redundancy and one for intra head-end redundancy.

5.8.2 The detection function

The detection function for inter head-end redundancy has already been described in section 5.3. This section is used to describe the differences between the detection function for the two forms of redundancy.

In case of inter head-end redundancy, the detection function is used to detect failing head-ends. In case of intra head-end redundancy, this function is used to detect failing boards. Note that boards are used, not channels. The reason for this is that although some boards may contain multiple channels, some of the hardware is used for all the channels. In case this hardware fails, the complete board fails. The basis of the detection function as described in section 5.3 is exactly the same for the detection function of the intra head-end redundancy system. This means that there is a detection ring and that each board within a single head-end detects if the next board of that detection ring is still working properly.

The detection function first waits for an *alive* message. If it receives one, it waits for a time interval *synctime* and then it sends the new *alive* message to the next board in the detection ring. The hardware watchdog as described in section 3.2.4 could be used in this context. Only when the watchdog guarantees the correct working of the board, the detection function of the board is allowed to continue the detection ring. Immediately after the continuation of the detection ring, it replies to the initial *alive* message with an *ack_alive* message. Now it waits for the time interval *synctime*. Then within the time interval *timeout* it should receive the *ack_alive* message from the next board in the detection ring. If not, this board is failing and the detection function forces the board to enter its alarm state. Meanwhile, it continues the detection ring with the failing board removed.

Since the CompactPCI bus is much faster then an typical Ethernet network, the total roundtime of the detection ring can be much shorter than the roundtime in case of inter head-end redundancy. The values of the different time intervals should be adjusted proportionately in order not to generate too much traffic.

The detection message is also the same for the two forms of redundancy. Only, since intra head-end redundancy uses a CompactPCI bus as network IP addresses cannot be used. Instead, hardware addresses could be used. Also, to simulate the packet based communication property of an Ethernet network, message queues could be used. The operating system used for the CableDock 200 is OSE[1]

[1] Enea OSE.

and OSE supports the use of message queues. A message queue can be described as an output and input buffer used to send and receive message to and from other OSE devices.

5.8.3 The information sharing function

The information sharing function for inter head-end redundancy has been described in section 5.4. The differences with the intra head-end redundancy counterpart of this function are discussed in this section.

For intra head-end redundancy the basis of the function remains the same as for inter head-end redundancy. In fact, the information that needs to be shared remains the same[1]. The only field in the different tables that is not really necessary for intra head-end redundancy is the field 'device name'. Since device related information is no longer interesting, Table 5.4 is not needed anymore. The information in the other tables still is necessary information since this information also represents the situation for intra head-end redundancy.

The Dec information (as represented by Table 5.1 up to and including Table 5.7) is stored at all boards. The Act information (as represented by Table 5.8) concerning a certain board is only stored at the board that precedes this board in the detection ring. The only exception of this rule is the System Controller. This board stores all information. The reason for this is that it already needs to store all information for the inter head-end redundancy system. Note that this means that a DS board may have information of an US board stored, even though it is never able to take over this US board in case it fails. This is not really a disadvantage because in case of inter head-end redundancy it is also very much possible that the head-end that has to take over the failing head-end is not the head-end that detected the failing head-end.

The message types are the same for the two forms of redundancy. *Delta_information* still is used for information due to a change in the local situation. *Checkup_information* is needed for intra head-end redundancy too and a *failing_device_clearup* message now will be used to command the other redundancy systems to clear their information concerning a failing board. A failing board will be removed from the detection ring and therefore, it is necessary that a board can get the Act information of the board that it precedes in the new (updated) detection ring. For this purpose the *detection_ring_change* messages are used. Furthermore, the *take_over_information* messages are required in case a failing board needs to be taken over by a board that does not have all information needed for such a take over action. Finally, since hotswap is supported by the CableDock 200 [12], it is possible that a board is added or removed while the other boards remain active. Therefore, the intra head-end redundancy system also uses the initialization state and the shutdown state. For this purpose the *device_change* messages are needed.

Note that the intra head-end information sharing function indirectly communicates with the inter head-end information sharing function. For example, when the situation of a DS board changes this results in the broadcast of a *delta_information* message. This message is also received by the information sharing function of the intra head-end redundancy system of the System Controller. As stated before, this board stores all information. However, this changed information does also mean a change in the situation of the complete head-end. Therefore, this also results in a *delta_information* message for the inter head-end redundancy network.

5.8.4 The take over decision function

The take over decision function that has been described for inter head-end redundancy in section 5.5, is presented below for the intra head-end redundancy system.

Since the information provided by the information sharing function is almost the same for the two forms of redundancy, the take over decision function is also almost the same for the two forms of

[1] The Dec and Act information as represented by Table 5.1 up to and including Table 5.8.

redundancy. The reason for this is that the take over decision function is based upon the information it gets. The input of this function still consists of the set of parameters of the failing device and the sets of parameters of the candidates to take over this failing device. In this case this failing device is a failing DS or US board. Note that this cannot be a failing System Controller because there is no candidate available to take over a failing System Controller within a single head-end. The quality factor Q_{device} has to be calculated for every board present in the head-end. Of coarse it is possible to ignore all US candidates (in advance) in case of a failing DS board and vice versa. However, this is not required since the calculations will result in $Q_{device} = 0$ for every US board[1]. The only sub quality factor that can be ignored in case of intra head-end redundancy is the factor Q_{device_load}.

5.8.5 The take over activating function

The take over activating function of the intra head-end redundancy system is the only function that has an important difference compared with its inter head-end redundancy counterpart as described in section 5.6. This has a physical reason. The take over activating function of a DS or US board is simply not able to re-program the HFC matrix switches because it is not connected to those switches. It has to use the System Controller to perform this part of its task. Therefore, the very first part of the take over activating function is actually the only part of this function that is present in the redundancy system of a DS or US board. This is the part that is triggered by the take over decision function to start with the take over procedure. Only now this procedure is to notify the take over activating function of the System Controller that the board chosen by the take over decision function should take over the failing board. This notification ends the take over activating function at a board because it does not have to send any static or dynamic information[2] and it also does not have to wait for a successful take over action because the System Controller can deal with a failing take over action too.

This also implies that the take over activating function of the System Controller is somewhat more complex. This function directs the actual take over action. However, compared with the take over activating function of the inter head-end redundancy system, it is less complex. The reason for this is that now it only has to direct the take over action of just one board taking over another board within the same head-end. It does not have to switch over a complete head-end and it also does not have to communicate with the router. Thus, the take over activating function for intra head-end redundancy of the System Controller is a simplified version of the take over activating function for inter head-end redundancy.

5.8.6 Remarks regarding intra head-end redundancy

The previous sections show that the redundancy system as it is defined for inter head-end redundancy, can also be translated into a redundancy system for intra head-end redundancy. The functions of the redundancy system are almost the same for the two forms of redundancy. Therefore, one basis (the four functions of the redundancy system) can be used for two redundancy systems.

However, one should wonder if this form of intra head-end redundancy is the best solution. Due to some of the restrictions of the intra head-end redundancy system, the advantages of the this part of the redundancy system are perhaps no longer worth the extra software and processing power needed for this intra head-end redundancy system. These restrictions express themselves in two mayor differences between the two forms of redundancy. First, the background of inter head-end redundancy is that generally speaking every head-end is able to take over another head-end. The reason for this is that every head-end combines the same services: DS and US data traffic handling and a management function. For intra head-end redundancy this is not the case. A DS board will never be able to take over an US board and vice versa.

The second difference is that in case of inter head-end redundancy, every head-end is able to perform the actual take over action, including the re-programming of the HFC matrix switches. Again, for intra

[1] An US board only has US channels present. Therefore, the calculation of for example, the Q_{DS_ch} factor will result in $Q_{DS\_ch} = 0$.
[2] The System Controller already has this information, for it stores all information in case of intra head-end redundancy.

head-end redundancy this is not the case. Every actual take over action has to be directed by the System Controller.

As stated before, the System Controller also has all information of the other boards that are present in the head-end because it needs to share this information with the other head-ends. Since it has all information required and since it has to direct an actual take over action anyway, it could also easily perform the take over decision task in case of a failing board. In fact, this would not even require any additional software because the take over decision function of the inter head-end redundancy system could also be used for intra head-end redundancy as showed in section 5.8.4. The only change that is needed is that the input of the function can also be the set of parameters of the failing board and the sets of parameters of the candidate boards. In this case the sub quality factor Q_{device_load} should also be ignored (set to 1).

A central take over decision function and a central take over activating function result in a System Controller that is an SPF. However, a System Controller will always be an SPF within a single head-end. In case it fails, the DS and US boards are not able to communicate with the Internet anymore so they are useless at that moment. The inter head-end redundancy system detects this failing System Controller and start the take over procedure of the complete head-end. Therefore, this central take over decision function and take over activating function does not decrease the availability of the head-end. Furthermore, as stated in section 5.8.5, the take over activating function of the intra head-end redundancy system is in fact a simplified version of its intra head-end redundancy system counterpart.

Now that the DS and US boards do not use (or have) their take over decision functions anymore, they do not need to store the information needed for this function. When no information needs to be shared, they also do not need the information sharing function. Note that the System Controller will still notice changes in the individual situation of a board, for the normal software of the System Controller also plays a central role in the normal payload data handling and management data handling of the CableDock 200.

The only thing left of the intra head-end redundancy system at boards other than the System Controller, is the detection function. In this case the intra head-end redundancy system would be very simple.

This detection function could also be taken over by the System Controller. However, this requires a feature of the CableDock 200 to be fully implemented. This feature is the hardware watchdog that is present at every DS and US board. As described in section 3.2.4, this watchdog can be used to guarantee the correct working of the board. When the microprocessor of a board periodically checks the status of the watchdog and pass this status on to the System Controller, the System Controller can recognize a board that is failing.

Although the redundancy system as it has been described for inter head-end redundancy, can be used for intra head-end redundancy as well, this may be not the best choice in case of a CableDock 200. The system as described in this section provides both inter and intra head-end redundancy with only one (expanded) inter head-end redundancy system. This redundancy system only needs some minor additions to work for intra head-end redundancy as well.

6 CASE

6.1 Introduction

In chapter 5 the complete redundancy system has been described for inter head-end redundancy. An intra head-end counterpart of this system also has been described, but as stated in section 5.8.6 this intra head-end version of the redundancy system may be not the best choice. In this chapter a case is presented to show the working of the complete inter head-end redundancy system. This includes not only the interaction of the four functions of this system with the same functions of another system, but also the interaction of these four functions within the same redundancy system. An example of intra head-end redundancy also is presented. However, this example is not based on the intra head-end counterpart of the redundancy system, but on the intra head-end redundancy system as presented in section 5.8.6[1].

First the situation will be presented. Then a head-end will be added to the inter head-end redundancy network. Furthermore, a failing head-end and a failing board will be introduced. Finally, a head-end will be shut down and therefore, removed from the redundancy network.

During this chapter the following acronyms are used:
Det X = Detection function of the redundancy system of device X.
Inf X = Information sharing function of the redundancy system of device X.
Dec X = Take over decision function of the redundancy system of device X.
Act X = Take over activating function of the redundancy system of device X.
Dec information = All information represented by Table 5.1 up to and including Table 5.7 present at a single head-end.
Act information = All information represented by Table 5.8 present at a single head-end.

6.2 The situation

The situation used for this case is based upon the situation as presented in Figure 5.6. Only, for this case some more detail and complexity is required. The resulting situation is illustrated in Figure 6.1. For convenience, the up-converters used by the DS channels are not displayed.

Figure 6.1 shows 3 head-ends and 2 HFC matrix switches. Furthermore, the operator has defined 3 DS areas and 5 US areas. He has already connected the various parts of the cable modem head-end system with the HFC matrix switches. For convenience, the HFC connections dealing with DS area 1 are presented as solid lines. The HFC connections dealing with DS area 2 are presented as dashed lines and finally, the HFC connections dealing with DS area 3 are presented as striped lines. The Ethernet connections dealing with payload data are presented as bold, sold lines and the Ethernet connections dealing with management data are presented as bold, dashed lines.

[1] This means that the DS and US boards have a hardware watchdog implemented and that the inter head-end redundancy system of the System Controller is expanded with some functionality. This functionality includes the ability to periodically check the status of the watchdogs and the ability to perform the take over decision task for intra head-end redundancy.

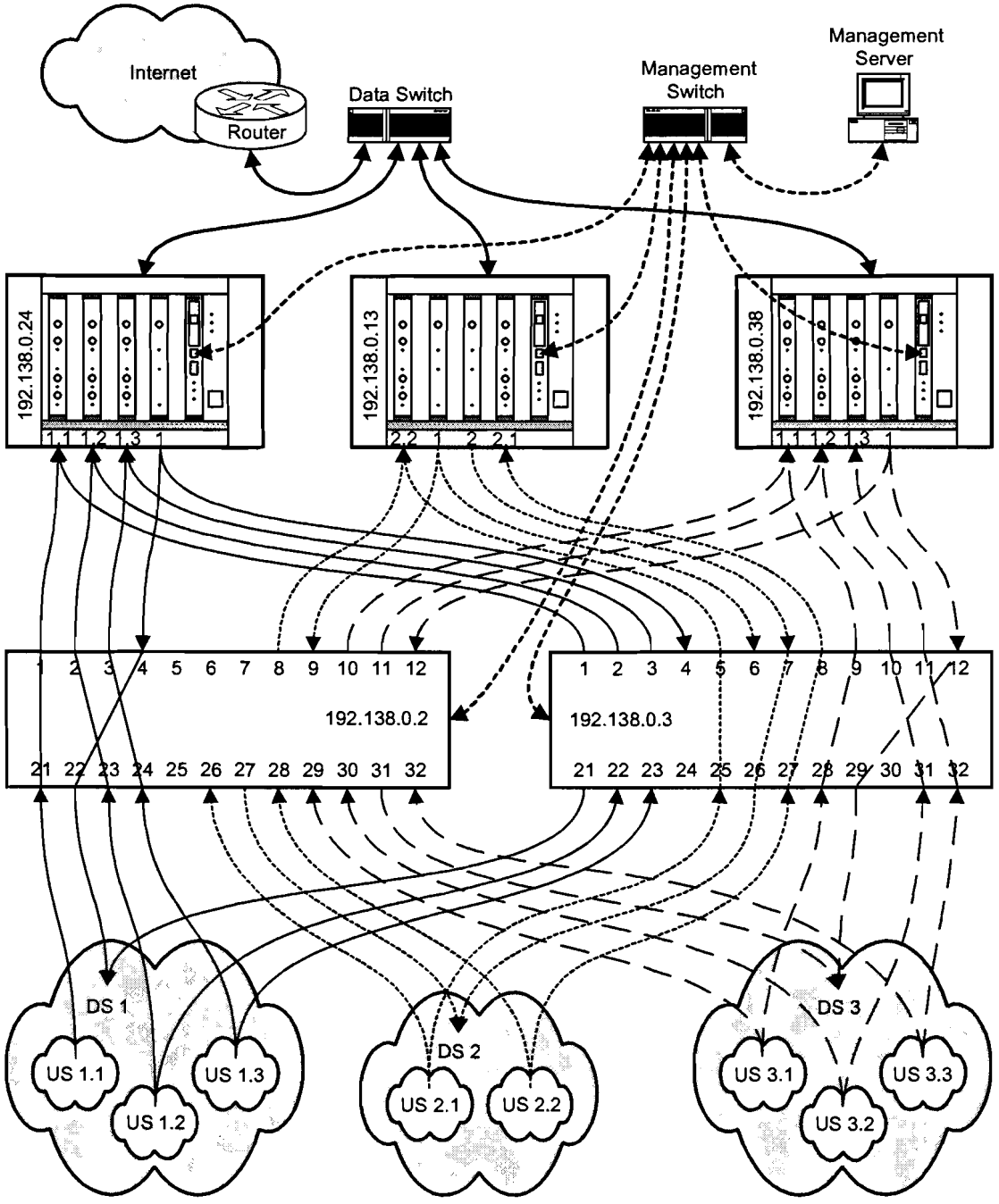

Figure 6.1. Situation used for the case.

The configuration files of the redundancy system of head-end 1 (IP address 192.138.0.24) are defined by the operator as:

Table 6.1. SwitchToCatvCxn configuration table of head-end 1.

Switch name [IP]	Output port name	DS/US area name
192.138.0.2	21	1.1
192.138.0.2	22	1
192.138.0.2	23	1.2
192.138.0.2	24	1.3
192.138.0.3	21	1
192.138.0.3	22	1.2
192.138.0.3	23	1.3

Table 6.2. SwitchState configuration table of head-end 1.

Switch name [IP]	Input port name	Output port name	Possible output ports
192.138.0.2	1	21	21,22,23,24,25,26,27,28,29,30,31,32
192.138.0.2	2	23	21,22,23,24,25,26,27,28,29,30,31,32
192.138.0.2	3	24	21,22,23,24,25,26,27,28,29,30,31,32
192.138.0.2	4	22	21,22,23,24,25,26,27,28,29,30,31,32

Table 6.3. SwitchToDeviceCxn configuration table of head-end 1.

Switch name [IP]	Input port name	Device name [IP]	DS/US channel name
192.138.0.2	1	192.138.0.24	1.1
192.138.0.2	2	192.138.0.24	1.2
192.138.0.2	3	192.138.0.24	1.3
192.138.0.2	4	192.138.0.24	1
192.138.0.3	1	192.138.0.24	1.1
192.138.0.3	2	192.138.0.24	1.2
192.138.0.3	3	192.138.0.24	1.3
192.138.0.3	4	192.138.0.24	1

Table 6.4. MainDeviceState configuration table of head-end 1.

Device name [IP]	Ethernet speed [Mbit/s]
192.138.0.24	100

Table 6.5. DsDeviceState configuration table of head-end 1.

Device name [IP]	DS channel name	To be taken over	Available for take over	Freq. [MHz]	Symbol rate [Mbaud]	Mod. type	Power level [dBμV]	Max. load [Mbit/s]	Max. number of modems
192.138.0.24	1	true	true	810	8	QAM64	90	40	2000

Table 6.6. UsDeviceState configuration table of head-end 1.

Device name [IP]	US channel name	To be taken over	Available for take over	Freq. [MHz]	Symbol rate [Mbit/s]	Mod. type	Power level [dBμV]	Max. load [Mbit/s]	Max. number of modems	Relative time delay [timeslots]
192.138.0.24	1.1	true	false	16	6	QPSK	70	6	2000	0
192.138.0.24	1.2	true	true	13	6	QPSK	70	6	2000	0
192.138.0.24	1.3	true	true	13	6	QPSK	70	6	2000	0

The configuration files of the redundancy system of head-end 2 (IP address 192.138.0.13) are defined by the operator as:

Table 6.7. SwitchToCatvCxn configuration table of head-end 2.

Switch name [IP]	Output port name	DS/US area name
192.138.0.2	26	2.1
192.138.0.2	27	2
192.138.0.2	28	2.2
192.138.0.3	25	2.1
192.138.0.3	26	2
192.138.0.3	27	2.2

Table 6.8. SwitchState configuration table of head-end 2.

Switch name [IP]	Input port name	Output port name	Possible output ports
192.138.0.3	5	25	21,22,23,24,25,26,27,28,29,30,31,32
192.138.0.3	7	26	21,22,23,24,25,26,27,28,29,30,31,32
192.138.0.3	8	27	21,22,23,24,25,26,27,28,29,30,31,32

Table 6.9. SwitchToDeviceCxn configuration table of head-end 2.

Switch name [IP]	Input port name	Device name [IP]	DS/US channel name
192.138.0.2	8	192.138.0.13	2.2
192.138.0.2	9	192.138.0.13	1
192.138.0.3	5	192.138.0.13	2.2
192.138.0.3	6	192.138.0.13	1
192.138.0.3	7	192.138.0.13	2
192.138.0.3	8	192.138.0.13	2.1

Table 6.10. MainDeviceState configuration table of head-end 2.

Device name [IP]	Ethernet speed [Mbit/s]
192.138.0.13	100

Table 6.11. DsDeviceState configuration table of head-end 2.

Device name [IP]	DS channel name	To be taken over	Available for take over	Freq. [MHz]	Symbol rate [Mbaud]	Mod. type	Power level [dBμV]	Max. load [Mbit/s]	Max. number of modems
192.138.0.13	1	false	true			QAM64		40	2000
192.138.0.13	2	true	true	810	8	QAM64	90	40	2000

Table 6.12. UsDeviceState configuration table of head-end 2.

Device name [IP]	US channel name	To be taken over	Available for take over	Freq. [MHz]	Symbol rate [Mbit/s]	Mod. type	Power level [dBμV]	Max. load [Mbit/s]	Max. number of modems	Relative time delay [timeslots]
192.138.0.13	2.1	true	true	16	6	QPSK	70	6	2000	0
192.138.0.13	2.2	true	true	13	6	QPSK	70	6	2000	0

Finally, the configuration files of the redundancy system of head-end 3 (IP address 192.138.0.38) are defined by the operator as:

Table 6.13. SwitchToCatvCxn configuration table of head-end 3.

Switch name [IP]	Output port name	DS/US area name
192.138.0.2	29	3.1
192.138.0.2	30	3.2
192.138.0.2	31	3
192.138.0.2	32	3.3
192.138.0.3	28	3.1
192.138.0.3	29	3
192.138.0.3	31	3.2
192.138.0.3	32	3.3

Table 6.14. SwitchState configuration table of head-end 3.

Switch name [IP]	Input port name	Output port name	Possible output ports
192.138.0.3	9	28	21,22,23,24,25,26,27,28,29,30,31,32
192.138.0.3	10	31	21,22,23,24,25,26,27,28,29,30,31,32
192.138.0.3	11	32	21,22,23,24,25,26,27,28,29,30,31,32
192.138.0.3	12	29	21,22,23,24,25,26,27,28,29,30,31,32

Table 6.15. SwitchToDeviceCxn configuration table of head-end 3.

Switch name [IP]	Input port name	Device name [IP]	DS/US channel name
192.138.0.2	10	192.138.0.38	1.1
192.138.0.2	11	192.138.0.38	1.2
192.138.0.2	12	192.138.0.38	1
192.138.0.3	9	192.138.0.38	1.1
192.138.0.3	10	192.138.0.38	1.2
192.138.0.3	11	192.138.0.38	1.3
192.138.0.3	12	192.138.0.38	1

Table 6.16. MainDeviceState configuration table of head-end 3.

Device name [IP]	Ethernet speed [Mbit/s]
192.138.0.38	100

Table 6.17. DsDeviceState configuration table of head-end 3.

Device name [IP]	DS channel name	To be taken over	Available for take over	Freq. [MHz]	Symbol rate [Mbaud]	Mod. type	Power level [dBμV]	Max. load [Mbit/s]	Max. number of modems
192.138.0.38	1	true	true	825	8	QAM64	95	40	2000

Table 6.18. UsDeviceState configuration table of head-end 3.

Device name [IP]	US channel name	To be taken over	Available for take over	Freq. [MHz]	Symbol rate [Mbit/s]	Mod. type	Power level [dBμV]	Max. load [Mbit/s]	Max. number of modems	Relative time delay [timeslots]
192.138.0.38	1.1	true	true	10	6	QPSK	70	6	2000	0
192.138.0.38	1.2	true	true	10	6	QPSK	70	6	2000	0
192.138.0.38	1.3	true	true	10	3	QPSK	70	6	2000	+1

6.3 Adding a head-end to the redundancy network

Suppose that the head-ends 1 and 2 are running. They have already set up the redundancy network and they are the only two devices participating in this network. This means that both head-ends have locally stored tables with Dec information[1]. Also, head-end 1 has Act information of head-end 2 and vice versa.

After some time the operator switches head-end 3 on. When this head-end is done booting, its redundancy system is started. This redundancy systems starts in the initialization state as described in section 5.7.2. First the configurations files of the redundancy system are processed. Then the procedure to be added to the redundancy network starts. This procedure is illustrated in Figure 6.2.

In Figure 6.2 the solid lines represent signals (messages) that are sent over the management network and the dashes lines represent signals between or the processing of the various functions of the redundancy system within a single head-end. Note that the length of the signals in time does not represent the actual duration of the signals. For convenience signals that do not deal with the initialization state of the redundancy system of head-end 3 are colored gray. The numbered signals represent signals of the redundancy system as described chapter 5. They are explained in detail below. When referring to a numbered signal, the number will be presented between brackets ().

The figure starts with Det 1 sending an *alive* message to Det 2. Immediately hereafter Det 1 replies to the previous device in the detection ring with an *ack_alive* message. Since there are only two device in this detection ring at this time, this message is also sent to Det 2. Det 2 first process the *alive* message and then starts waiting for the time interval *synctime* to continue the detection ring. Det 1 is waiting for the *ack_alive* message to end its detection cycle. Note that in this case, first the new detection cycle is started and then the old one is stopped.

At about the same time, the redundancy system of head-end 3 enters its initialization state. First Inf 3 waits for a random time interval (1). Then it broadcasts a *device_change* message (3) and it triggers Det 3 to wait for an *alive* message (2). The broadcasted message (3) is received by Inf 1 and Inf 2 and they recognize it and pass it on to Det 1 and Det 2 (5). Meanwhile Det 3 is waiting for the first *alive* message (4). If it would not receive this message after a certain time interval configured by the operator, it would conclude that there are no other head-ends present. In this case, its redundancy system will not do a thing until a *device_change* message is received.

Since head-end 2 is the head-end that is waiting to send an *alive* message, this head-end reacts to the *device_change* message (3). First Det 2 processes the message to check if it is either a device that wants to be added to or removed from the redundancy network (6). The IP address of head-end 3 is not present in the detection ring, so it concludes that this device wants to be added to the redundancy network. It adjusts the detection message (6) according to the procedure presented in section 5.3.5. This means that head-end 3 is inserted between head-end 2 and head-end 1 in the detection ring. Then it sends the new *alive* message to Det 3 (8). Immediately hereafter it replies to the Det 1 with an *ack_alive* message.

[1] In this case this Dec information consists of Table 6.1 and Table 6.7 combined, Table 6.2 and Table 6.8 combined, Table 6.3 and Table 6.9 combined, Table 6.4 and Table 6.10 combined, Table 6.5 and Table 6.11 combined and Table 6.6 and Table 6.12 combined. Finally, this includes a table with the information as presented in Table 5.7, combined for both head-ends.

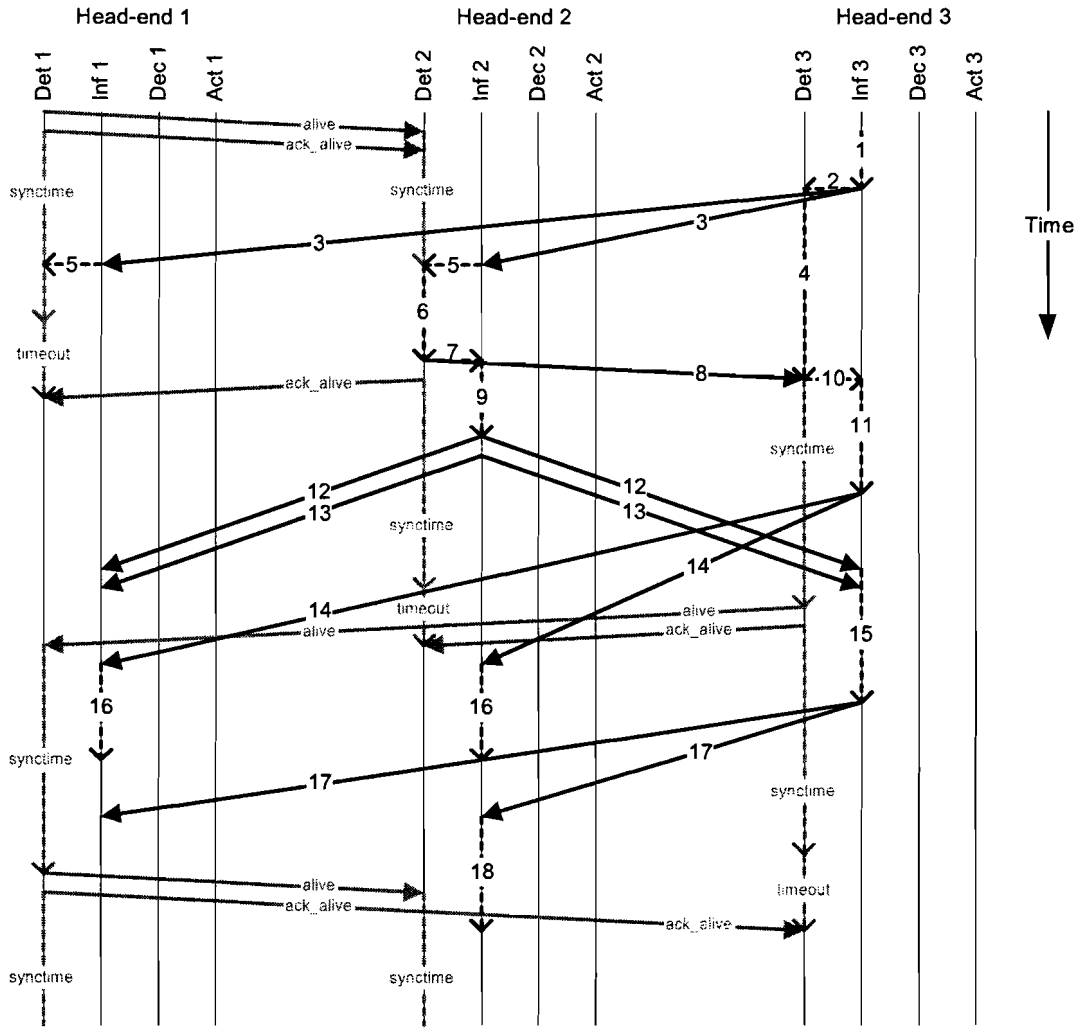

Figure 6.2. Flow chart (in time) of the redundancy network in case head-end 3 enters its initialization state.

Since the detection ring has changed, Det 2 triggers Inf 2 to start the procedure that belongs to this change (7). This procedure consists of two parts because the reason why the detection ring changed is the addition of a device. The first part is to inform the head-end 3 about the general situation of the redundancy network (all Dec information) and about the situation of head-end 1[1] (the Act information of head-end 1). The second part is to get the Act information of head-end 3, since head-end 2 precedes this head-end in the new detection ring. Inf 2 generates all *detection_ring_change* messages for these two purposes (9).

Meanwhile, Det 3 receives the *alive* message (8) of Det 2. This starts the normal detection cycle at this head-end. This also informs the redundancy system of head-end 3 that it is not alone. Det 3 informs Inf 3 that it can start with normal operation (10). One could say that this ends the initialization state because from this moment on, the functions will only perform standard tasks. However, note that the information head-end 3 has, is far from complete. It only has its own information. Therefore, the initialization state ends when the information is complete.

Inf 3 starts its normal operation and therefore starts with sharing the changed information. All initial information present in its configuration files is changed information, for at first it was not present.

[1] In the new detection ring, the Act information concerning head-end 1 is stored at head-end 3 instead of head-end 2.

Therefore, Inf 3 starts with generating the *delta_information* messages that represent this changed information (11). When done, it starts broadcasting these messages. There are 6 messages (14), each one representing the information of one of the configuration tables (Table 6.13 up to and including Table 6.18). Inf 1 and Inf 2 receive these messages (14) and since they all represent Dec information, they all are processed (16).

When Inf 2 is finished with the generation of the messages (9), it starts broadcasting these messages. First the *detection_ring_change* messages containing all Dec information and Act information (12). In fact, this includes 7 messages with Dec information of the previous redundancy network (head-end 1 and head-end 2) and 1 message with the Act information of head-end 1. Immediately hereafter it broadcasts the *detection_ring_change* message to get the Act information of head-end 3 (13). Besides Inf 3, Inf 1 also receives all messages (12) and (13) but it ignores them since they are not interesting for this head-end. Inf 3 processes all messages (15). First the messages with the Dec information of the previous redundancy network and the Act information of head-end 1 (12). After this processing it has all Dec information and Act information needed, so this processing ends the initialization state of head-end 3. However, during this processing it also received a *detection_ring_change* message to get its Act information (13) and this message is processed as normally would be done. So, Inf 3 reacts with the broadcast of the requested Act information in a *detection_ring_change* message (17). Again, Inf 1 ignores the message. Inf 2 processes the message (18) because it is the information it needs.

6.4 A failing head-end

Suppose that all three of the head-ends as presented in Figure 6.1 are up and running for some time. The detection ring is ordered in the following way: head-end 1 (192.138.0.24), head-end 2 (192.138.0.13) and head-end 3 (192.138.0.38). Since all head-ends are running for some time, they all have the following Dec information table stored.

Table 6.19. DeviceLoad table.

Device name [IP]	DS/US channel name	Load [Mbit/s]	Connections
192.138.0.24	1	25	1000
192.138.0.24	1.1	3	300
192.138.0.24	1.2	3	300
192.138.0.24	1.3	4	400
192.138.0.13	1	0	0
192.138.0.13	2	20	1400
192.138.0.13	2.1	2	600
192.138.0.13	2.2	2	800
192.138.0.38	1	35	1840
192.138.0.38	1.1	1	350
192.138.0.38	1.2	2	390
192.138.0.38	1.3	4	1100

Then, the System Controller of head-end 1 crashes and therefore head-end 1 goes down. Det 3 detects this and forces the redundancy system of head-end 3 to enter its alarm state. This starts Dec 3. The signaling that occurs in this case, is illustrated later on this section. First, the calculations of Dec 3 as presented in section 5.5.2, 5.5.3 and 5.5.4 are presented. The same order of calculations and naming of the parameters as presented in the sections just mentioned, is used. Therefore, the names of the individual formulas are not referred to.

Dec 3 starts with the definition of the various input collection and the filtering of these input sets:

$failingdevice(headend) = \{192.138.0.24\}$

$candidates(headend) = \{192.138.0.13, 192.138.0.38\}$

$DS_f = \{1\}$

$US_f = \{1.1, 1.2, 1.3\}$

$SW_{DS_f} = \{192.138.0.2, 192.138.0.3\}$

$SW_{US_f} = \{192.138.0.2, 192.138.0.3\}$

Then Dec 3 starts with the calculations of the sub quality factors for both candidates. These calculations are present in Appendix C. The resulting quality factor Q_{device} of head-end 2 as defined by formula 5.122 is:

$$Q_{device} = 1 \cdot 1 \cdot 0.5 \cdot 1 \cdot 1 \cdot 1 \cdot 0.429 \cdot 1 \cdot 1 \cdot 1 \cdot 0.459 \cdot 1 \cdot 0.5 \cdot 1 \cdot 0.764 \cdot 1 \cdot 0.5 = 0.0188$$

After head-end 2, Dec 3 starts with the calculations of the sub quality factors for head-end 3. These calculations are also present in Appendix C. The resulting quality factor Q_{device} of head-end 3 is:

$$Q_{device} = 1 \cdot 1 \cdot 0.833 \cdot 1 \cdot 1 \cdot 0.444 \cdot 0.923 \cdot 0.246 \cdot 1 \cdot 0.25 \cdot 0.25 \cdot 1 \cdot 0.833 \cdot 1 \cdot 0.917 \cdot 1 \cdot 0.833 = 0.00334$$

Since the resulting quality factor Q_{device} of head-end 2 is higher than the one of head-end 3, Dec 3 decides that head-end 2 should take over the failing head-end 1.

Note that one cannot state that head-end 2 will take over head-end 1 for 1.88% of the ideal take over action. To bring these values to a more real percentage, formula 5.124 can be used. Using this formula, one could say the head-end 2 will take over head-end 1 for 79.2% of the ideal take over action. Head-end 3 would take over head-end 1 for 71.5% of the ideal take over action.

Figure 6.3 shows the detection of the failing head-end 1 and all actions that follow after this detection. The solid lines represent messages that are sent over the management network and the dashed lines represent signals between or processing time of the functions of a single redundancy system. Note that all messages sent by the information sharing systems are broadcasted messages. For example, message (9) is broadcasted and therefore Inf 1 also receives this message. However, for convenience these messages are not showed since head-end 1 is down anyway.

The figure starts when Det 3 sends an *alive* message to Det 1 (1). Immediately hereafter Det 3 sends an *ack_alive* message to Det 2. Then it starts waiting, first for the time interval *synctime* (2) and then for the time interval *timeout* (3). After this time interval it should have received the *ack_alive* message from Det 1. Since this is not the case, Det 3 concludes that there must be a problem with head-end 1 and therefore it forces the redundancy system of head-end 3 to enter its alarm state.

This alarm state starts with signal (4) to notice Inf 3 that a failing device is present in the redundancy network and that the information of this device should be removed. At the same time Dec 3 is noticed that head-end 1 is failing (5). Dec 3 starts with the take over decision task as described earlier this section (8).

Inf 3 processes its information tables to see what information concerning head-end 1 should be removed (7). When done, it starts broadcasting the *failing_device_clearup* messages (9). Note that this includes 7 messages, all representing a part of the Dec information. Also note that Inf 3 does not delete the locally stored information since this is still needed by Dec 3. These message (9) are received by Inf 2 and it starts processing these messages (12).

Meanwhile, Det 3 is processing the detection message to remove head-end 1 from the detection ring (6). When done, it sends an *alive* message to the next device in the new detection ring, Det 2. From now on, the detection functions work in the standard way again. At the same time Det 3 sends the *alive* message, it also notifies Inf 3 that the detection ring has changed and that Inf 3 should take the appropriate actions (10).

Triggered by Det 3 (10), Inf 3 starts the procedure to get the Act information of the head-end it precedes in the new detection ring (11). This means that it generates a *detection_ring_change* message with the IP address of the message set to 192.138.0.13 (head-end 2) and the *data_type* set to *CxnTakeOver* with the data field itself left blank. Then it broadcasts this message (13).

The broadcasted *detection_ring_change* message (13) is also received by Inf 2. Since Inf 2 recognizes the IP address of the message as its own, it processes the message and it fills in the data field (14). When done, it broadcasts the resulting *detection_ring_change* message (15). This message is received by Inf 3 and since it is the information it wants, it processes the message (16).

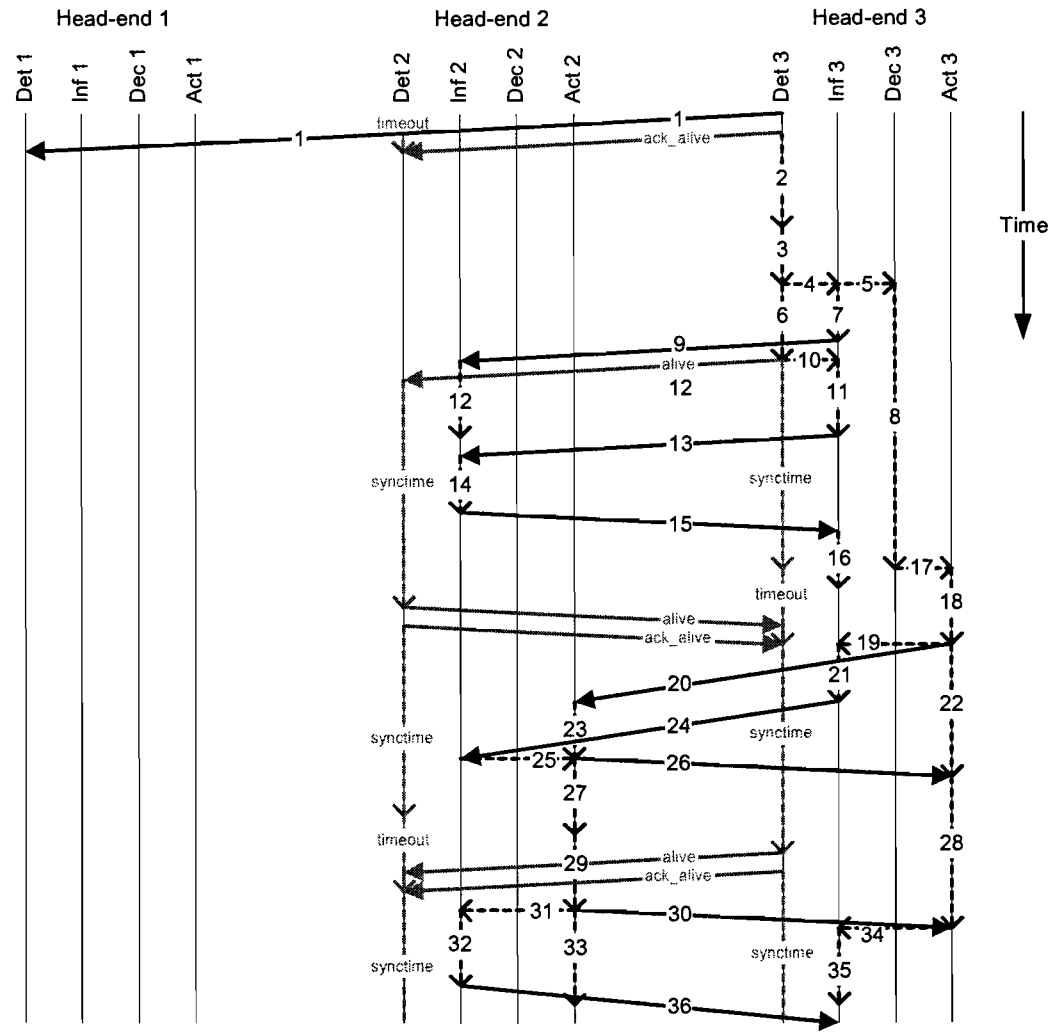

Figure 6.3. Flow chart (in time) of the redundancy network in case head-end 1 fails.

After some time, Dec 3 is finished with the take over decision task (8). Based on the calculations earlier this section, it concludes that head-end 2 should take over the failing head-end 1. The backup device is head-end 3 because it had the second highest quality factor Q_{device} (that did not equal 0). It passes this decision on to Act 3 (17).

Act 3 now first check if its own head-end should take over the failing device (18). This is not the case, so it sends an *take_over* message to Act 2 to inform the redundancy system of this head-end that it should enter its alarm state and start the actual take over task (20). At the same time, Act 3 notifies Inf 3 that head-end 2 needs the Act information concerning head-end 1 (19). Inf 2 first generates the appropriate messages (21) and then broadcasts these messages (24). Note that it should send all information concerning head-end 1, so also some of the Dec information[1]. Act 3 now is waiting for Act 2 to acknowledge that it has all information needed (22).

Act 2 receives the *take_over* message sent by Act 3. Since it does not have all information needed, it first waits for this information to arrive (23). When Inf 2 receives the required information it notifies Act 2 that new information is available (25). When Act 2 has all information required it sends an *ack_take_over* message to Act 3 (26). Act 3 now knows that the actual take over action is starting.

[1] See also section 5.6.

Act 3 now starts waiting for a message of Act 2 that the actual take over action has been performed successfully (28).

Meanwhile, Act 2 starts preparing the actual take over action (27). It generates the messages to re-program the switches, the messages to inform the router and the tables with the modems that were served by head-end 1 before its crash. Furthermore, it configures the connections parameters for the modems after the take over action[1]. When done, it starts with the actual take over action (29).

When the switches are re-programmed and the router has cleared all entries concerning the modems that have been taken over, Act 2 sends a *take_over_ready* message to Act 3 (30). Act 3 reacts to this message by informing Inf 3 that all local information concerning head-end 1 now can be cleared (34). Inf 2 processes its local tables and clears all entries concerning head-end 1 (35). When done, the redundancy system of head-end 3 returns to the running state.

Act 2 also informs Inf 2 that the static information concerning head-end 2 has changed (31). Inf 2 processes these changes (32) and this results in *delta_information* messages that are broadcasted (36). Finally, Act 2 waits for a certain time interval (33) to allow modems that have been taken over, to get back their original connection. After this time interval, the redundancy system of head-end 2 returns to the running state.

Note that immediately after the actual take over action, there will occur a lot of changes concerning head-end 2. For example, the load will change, the number of active connections, the Act information etc. This results many (normal) *delta_information* messages.

6.5 A failing board

In section 5.8.6 a method is presented to use the inter head-end redundancy system for intra head-end redundancy too. In this section an example of this method is presented.

Suppose that head-end 3 has only boards with one channel per board. These boards are named in the following way:

- DS board 1: DS channel 1
- US board 1: US channel 1.1
- US board 2: US channel 1.2
- US board 3: US channel 1.3

At a certain time, US board 2 crashes. Suppose that at this time, the situation concerning head-end 3 as presented in Table 6.19 still applies. Det 3 would detect this failing US board. Since there is no intra head-end detection ring, there is also no intra head-end detection message that needs to be adjusted. Det 3 only forces the redundancy system of head-end 3 to enter its alarm state. This starts Dec 3. Furthermore it notifies Inf 3 to broadcast *failing_device_clearup* messages. Compared with the situation as described in the previous section, this time these messages only contain the Dec and Act information concerning US board 2 (and thus US channel 1.2). Also, note that the Act information needs to be sent, for head-end 2 should know that the modems served by US channel 1.2 are no longer served.

Dec 3 starts with the definition of the various input collection and the filtering of these input sets:

$failing\,device(board) = \{2\}$

$candidates(board) = \{1,3\}$

$DS_f = \{\ \}$

$US_f = \{1.2\}$

[1] For example, in case the load of a DS channel will become too high after the take over action, the individual bandwidth of the modems has to be decreased. In case of a CableDock 200, this can be achieved by adjusting the leaky bucket parameters [12].

$SW_{DS_f} = \{\ \}$

$SW_{US_f} = \{192.138.0.2, 192.138.0.3\}$

Note that although US channel 1.2 is connected with both the HFC matrix switches, only the situation for switch 192.138.0.3 needs to be calculated because all channels involved are connected to this switch. The connections with 192.138.0.2 are not going to provide any additional information. Therefore:

$SW_{US_f} \Rightarrow \{192.138.0.3\}$

The calculations of the various sub quality factors for both candidates are present in Appendix C. The resulting quality factor Q_{device} of US board 1 as defined by formula 5.122 is:

$Q_{device} = 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 = 1$

The resulting quality factor Q_{device} of US board 3 is:

$Q_{device} = 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 0.75 \cdot 1 = 0.75$

Therefore, Dec 3 decides that US board 1 should take over the failing US board 2. This information is passed on to Act 3 and Act 3 starts with the preparation of the actual take over action. When finished, it starts the actual take over action after which it notifies Inf 3 to clear the local information of US board 2. Furthermore, it notifies Inf 3 that the situation concerning the static information of head-end 3 has changed. This results in the broadcast of *delta_information* messages with this static information. After the take over action, Act 3 waits for a certain time interval after which it ends the take over activating procedure. This also ends the alarm state of the redundancy system of head-end 3.

This example shows that the redundancy system is able to cope with intra head-end failures too. It is even possible to expand this functionality in such way that instead of only using boards present in the same head-end as input for the take over decision function, also boards present in the other head-ends of the redundancy network can be used as input. However, in this case failing boards can no longer be treated independently. In this case, the fact that DS areas and US areas (and thus DS channels and US channels) are linked has to be used in the calculations.

6.6 Removing a head-end from the redundancy network

After the failing head-end 1 and the failing US board 2 of head-end 3, there are only two head-ends still running and participating in the redundancy network. After a while, head-end 1 is back online and participating in the redundancy network again. The detection ring is ordered as it used to be ordered, thus in the following way: head-end 1 (192.138.0.24), head-end 2 (192.138.0.13) and head-end 3 (192.138.0.38).

Suppose that the operator wants to shut down head-end 3 for maintenance purposes. The procedure to shut down a device has been described in section 5.7.3. An example of this procedure is presented below. Note that it is an example of the proper way to shutdown a device.

Figure 6.4 shows the shutdown procedure of head-end 3. The figure starts when Det 1 sends an *alive* message to Det 2. Immediately hereafter it sends an *ack_alive* message to Det 3. Then, the operator initiates the shutdown procedure of head-end 3. This triggers Inf 3 to generate *delta_information* messages (1). These messages contain all Dec and Act information concerning head-end 3 and the information is presented as 'to be removed'. When done, these messages are broadcasted (3). Both Inf 1 and Inf 2 receive these messages and both process the Dec information messages (7). Furthermore, Inf 2 also processes the Act information message since it concerns head-end 3 and this is the head-end it precedes in the detection ring (7).

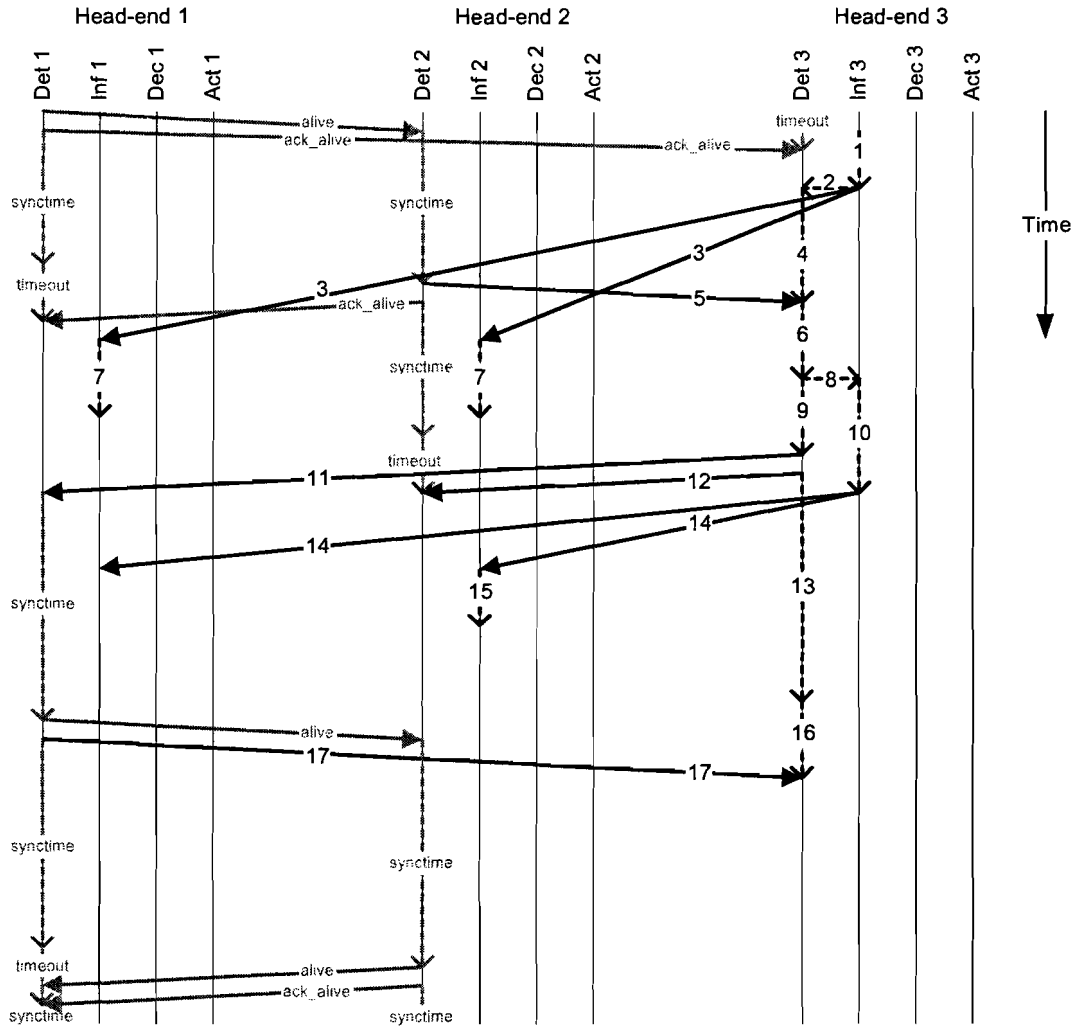

Figure 6.4. Flow chart (in time) of the redundancy network in case head-end 3 enters its shutdown state.

After broadcasting the *delta_information* messages, Inf 3 notifies Det 3 to wait for an *alive* message in order to continue the shutdown procedure (2). Det 3 waits for the time interval (4) and then it receives an *alive* message from Det 2 (5). This message is first processed and head-end 3 is removed from the detection ring (6). After the processing, Det 3 informs Inf 3 that the detection ring has changed (8). This triggers Inf 3 to generate a *detection_ring_change* message (10). This message is needed to provide head-end 2 with the Act information of head-end 1 because head-end 2 precedes head-end 1 in the new detection ring. When done, the *detection_ring_change* message is broadcasted (14). It is received by Inf 1 and Inf 2 but only Inf 2 processes the message because the IP address of the message is set to its IP address (15).

Meanwhile, Det 3 is waiting to continue the detection ring with the new detection message (9). When the time interval *synctime* is finished, it sends the new *alive* message to Det 1 (11). Immediately hereafter it sends an *ack_alive* message to Det 2 (12). Then it starts waiting for the new time interval *synctime*[1] (13) and *timeout* (16). If everything is alright with head-end 1, it receives the *ack_alive* message (17) within time interval *timeout*. This message ends the shutdown state of the redundancy system of head-end 3. The operator now is allowed to switch off head-end 3.

[1] In the new case there are just two head-ends present in the detection ring. Therefore, the value of *synctime* changes.

7 CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

In chapter 1 the two main objectives of the graduation project were defined. The first main objective was to investigate if it is possible to improve the availability of the cable modem head-end system as it is developed by BarcoNet Eindhoven B.V. The second main objective was to investigate how this improved availability could be realized.

In chapter 2 the term 'improved availability' has been further specified. This resulted in two main design objectives. The first design objective was that the availability of the complete cable modem head-end system should be as high as possible from the user point of view. The second design objective was that the realization of this improved availability should be as simple as possible. These two main design objectives were used as the targets of three possible solutions as presented in chapter 3. However, none of these solutions satisfied both main design objectives. Therefore, these solutions were combined into one overall solution: the redundancy system. In chapter 4 the two main design objectives were translated into a design requirements specification for this redundancy system.

As stated in chapter 4, the design requirements specification of the redundancy system could later on be used as a checklist to see if the redundancy system as it has been described in chapter 5, satisfies the initial design objectives.

The first main design objective was that the availability of the complete cable modem head-end system should be as high as possible from the user point of view. Without the redundancy system implemented, the availability from the user point of view, is mainly defined by the availability of the head-end that serves that user. In fact, this head-end is a single point of failure between the user and the Internet. This means that if the head-end would fail in providing its service to the user, this user would have no other option than to wait until the failure has been repaired. In practice, this means that first the operator has to notice the failing head-end. Then, maintenance personal has to travel to the problem location. This would take several minutes to hours worst case, depending on the geographical location. After they have arrived, they have to repair the failure. Depending on the seriousness of the failure, this could also take several minutes to hours worst case. Therefore, the total time the user has to wait until he gets back the service, can be as much as several hours depending on various factors (influenced by humans).

With the redundancy system implemented, the availability from the user point of view is no longer defined by the availability of a single head-end. The availability now is defined by a cluster of head-ends. Theoretically, only one of these head-ends should be available to have the complete cluster available from the user point of view. In practice, this availability of the cluster is limited due to the configuration of the individual head-ends. By adjusting the configuration of the cluster, an operator influences the availability from the user point of view. Whenever a head-end fails, the redundancy system takes care of the users connected to this head-end. They will still suffer from packet loss and perhaps even broken connections, only they will get back their service within a few minutes at most. The maintenance personal still has to repair the failing head-end, only since this head-end no longer serves any user this can have a much lower priority than without the redundancy system implemented. Furthermore, the redundancy system does not require complete redundant head-ends. Therefore, the operator is free to add extra capacity without decreasing the availability as is the case with N+1 solutions.

Concerning the first main design objective, the conclusion can be drawn that the availability from the user point of view has been improved from hours downtime to minutes downtime every time a head-end fails. Furthermore, the pressure on the operator in case of a failing head-end has seriously decreased. Finally, in section 5.8 a method has been described to use this same redundancy system for failing boards of the head-end as well. Included in the redundancy system this would further

increase the availability from the user point of view and therefore, further decrease the pressure on the operator in case of a failing board.

The second main design objective was that the improved availability would have to be realized as simple as possible. The redundancy system as described in chapter 5, is divided into four functions. Each function alone performs some very simple tasks of the redundancy system. These functions can be programmed and tested independently. Together, they form a more complex system that is able to automatically react in case a part of the cluster of head-ends fails.

First it was thought that the redundancy system for inter head-end redundancy should also be translated into a similar redundancy system for intra head-end redundancy. The reason for this was that the cluster of head-ends could be seen as a symmetrical distributed multiprocessor environment. When zooming in on a single head-end, again a symmetrical distributed multiprocessor environment can be seen. Two similar redundancy systems would increase the simplicity of the total solution. However, as stated in section 5.8.6 it turned out that for a CableDock 200 this would not be the best option. Another method was presented that actually is an expansion of the redundancy system for inter head-end redundancy. Although the resulting redundancy system does not work exactly the same for inter head-end redundancy and intra head-end redundancy, it is just one system.

Concerning the second main design objective, the conclusion can be drawn that the realization of the improved availability has resulted in a simple redundancy system. The ultimate target of having one redundancy system that works for both inter head-end redundancy and intra head-end redundancy in the exact same way, is not reached. However, this is due to practical reasons; it has been proved that it is possible to use the redundancy system for both forms of redundancy.

Finally, the main conclusion is that the redundancy system as it has been defined in this report, provides an improved availability of the CableDock 200 using a simple implementation.

7.2 Recommendations

The redundancy system as it has been described in chapter 5, provides the basis for the actual implementation in software. Before programming the software, first the individual functions of the redundancy system should be simulated and tested. These tests then should be expanded to test the interaction of the functions with the same functions of another redundancy system. Finally, the complete redundancy system should be simulated and tested in an environment with other redundancy systems. These tests will provide data about the actual time it takes before a failing device has been detected, the decision has been made which device should take over the failing device and the actual take over action has be performed. These tests will also provide a more detailed insight in the network traffic on the management network that is generated by the four redundancy functions.

With the help of these simulations the individual functions can be optimized. For example, in section 5.4 it turned out that almost all of the information sharing message types defined, are similar messages. Only the reasons for sending a message change. Perhaps these messages can be merged into one message.

The take over decision function as it has been described in section 5.5 provides a method to decide which candidate is the best candidate to actually take over a failing device. The calculations used to make this decision can be further optimized. This would improve the speed of the take over decision calculations. The calculations are based upon the failing device. They could also be based upon the DS and US areas served by that failing device. This would result in other and perhaps faster calculations. Finally, the calculations could also be expanded. For example, now they do not include a sub quality factor due to restrictions of the possible combinations of output ports of the switch that can be interconnected with one input port.

The redundancy system provides an automatic way to take over failing devices. However, it does not provide an automatic way for the failing device after it has been repaired, to get back all connections it served before the failure. This could be implemented with the help of other software, like load balancing software or re-provisioning software. It could also be a part of the redundancy system. This would require that the redundancy system keeps a history of all actions that have been taken.

Another expansion of the redundancy system would be to make it capable of spreading the load of a failing head-end over multiple head-ends. Now the system only decides which head-end is the best candidate to take over a failing head-end. This results in a head-end that handles the load of two head-ends. Spreading the load of the failing head-end over multiple head-ends would result in a head-ends that handle just somewhat more than the load they normally handle. This also could be implemented by using load balancing software after the take over action.

The HFC matrix switches are commercially available. However, they turned out to be very expensive. The reason for this is that they are very fast (faster than required for this purpose) and they can switch the complete spectrum of 5 MHz to 1 GHz. Since US traffic does not use the complete spectrum (only 5 MHz – 65 MHz) and the output DS frequency (before the up-converter) is always the same IF frequency, these switches are over-dimensioned for this purpose. Since these switches form an indispensable part of the redundancy system, they need some further attention.

Finally, as stated in section 5.8 the intra head-end redundancy system could be realized with the help of the inter head-end redundancy system. To implement this, the hardware watchdogs that are present on the DS and US boards should be used to guarantee a correct working of the board. Using this, a board that is able to pass on the status of the watchdog to the System Controller, is still working correctly. In case it cannot be passed, the System Controller could take preventive actions. It could already start with the take over decision procedure and prepare the actual take over action. This would result in a smoother take over action in case the board is failing after all.

This principle of guaranteeing a correct working can be expanded to the complete head-end. Whenever a head-end receives an *alive* message, it should only react if it is absolutely sure that everything within the head-end is working correctly. Otherwise, it is not allowed to react resulting in a take over action of another head-end. This also gives a smoother take over action, for it is possible that the users are serviced (at a degraded service level) until just before the take over action. This could even be expanded with the possibility for the redundancy system of a head-end to broadcast emergency messages to notify the other redundancy systems that it should be taken over.

There are many things concerning the redundancy system that can be adjusted to provide more or a better functionality. However, with every adjustment one should always keep in mind what the benefits and costs of such an adjustment are. For example, suppose that the redundancy system and thus the availability can be improved in such way that the take over procedure per user and per failing device goes from 4 minutes to 3 minutes. However, if the costs of this improved take over action would be the broadcasting of twice as much information the whole time, one should seriously wonder if this should be implemented. The redundancy system as defined in this report, provides only the basic things needed to improve the availability without making the system more and too complex.

8 References

[1] *'Digital Video Broadcasting (DVB); DVB interaction channel for Cable TV distribution systems (CATV)'*, ETS 300 800, DVB-RCC version 2, ETSI, November 2000.

[2] *'Technical Reference Manual CableDock® 200 r1.1'*, BarcoNet Eindhoven B.V., 2001.

[3] *'Telecommunications Act'*, Ministry of Transport, Public Works and Water Management, 1998.

[4] *'Rapportage Internetuitkoppeling'*, Onafhankelijke Post en Telecommunicatie Autoriteit (OPTA) and Nederlandse Mededingingsautoriteit (NMa), 2000.

[5] *'Beleidsnotitie Nationaal Trusted Third Parties (TTP) Project'*, Ministry of Transport, Public Works and Water Management and Ministry of Economic Business, March 1999.

[6] *'Sixth Report on the Implementation of the Telecommunications Regulatory Package'*, COM (2000) 814, Commission of the European Communities, December 2000.

[7] *'Network Aspects (NA); Quality of service indicators for Open Network Provision (ONP) of voice telephony and Integrated Services Digital Network (ISDN)'*, ETR 138, second edition, ETSI, December 1997.

[8] *'The Quality of Voice Telephony Services and Related Consumer Protection Issues; Final Report; Volume 1; Executive Summary, Synthesis and Recommendations'*, study for the European Commission, SagaTel, January 2000.

[9] *'The Quality of Voice Telephony Services and Related Consumer Protection Issues; Final Report; Volume 2; Country Monographs'*, study for the European Commission, SagaTel, January 2000.

[10] Vet, J: *'CableDock® 200 – Commercial Requirements Specification'*, version 1.1, internal document of the CableDock® 200 of BarcoNet Eindhoven B.V., 2000.

[11] Beuk, L: *'CableDock® 200 – Functional Requirements Specification'*, internal document of the CableDock® 200 of BarcoNet Eindhoven B.V., 2000.

[12] Koolen, G.J.K.M.: *'CableDock® 200 – Architectural Design Document'*, internal document of the CableDock® 200 of BarcoNet Eindhoven B.V., 2000.

[13] *'PacketCable™ 1.0 Architecture Framework Technical Report'*, PKT-TR-ARCH-V01-991201, Cable Television Laboratories Inc., December 1999.

[14] *'PacketCable™ 1.0 Security Specification'*, PKT-SP-DVBSEC-D01-000801, Cable Television Laboratories Inc., December 1999.

[15] *'Providing Open Architecture High Availability Solutions r1.0',*
High Availability Forum, http://developer.intel.com/platforms/applied/eiacomm/haforum.htm,
last checked on 4-12-2001.

[16] Koolen, A.:
'CableDock® 200 - ADD – Software Upstream CD200',
internal document of the CableDock® 200 of BarcoNet Eindhoven B.V., 2001.

[17] Lewis, E.E.:
'Introduction to Reliability Engineering',
2nd edition, New York: Wiley, 1996.

[18] Failure Mode and Effect Analysis (FMEA),
FIC FMEA Information Centre®, http://www.fmeainfocentre.com, last checked on 4-12-2001.

[19] Quintech™ Electronics and Communications Inc.,
http://www.qecinc.com.

[20] La Hei, D., Bruls, M. and Maarl, A. van der:
'CableDock® 200 - DDD Traffic Data Handler – System Controller CD200',
internal document of the CableDock® 200 of BarcoNet Eindhoven B.V., 2000.

[21] Stevens, W.R.:
'TCP/IP Illustrated, Volume 1: The Protocols',
17th printing, Addison Wesley Longman, Inc., 2000.

Appendix A: Acronyms and definitions

Table A.1. Acronyms used throughout the report.

Acronym	Description
ARP	Address Resolving Protocol
BER	Bit Error Rate
CATV	Community Antenna TeleVision / Cable TV
CD200	CableDock 200
CMTS	Cable Modem Termination System (DOCSIS Head-end)
CoS	Class of Service
CRS	Commercial Requirements
DOCSIS	Data Over Cable Service Interface Specification
DS	Downstream
DVB	Digital Video Broadcasting
ETSI	European Telecommunications Standards Institute
EU	European Union
FMEA	Failure Mode and Effect Analysis
FPGA	Field Programmable Gate Array
FRS	Functional Requirements
HA	High Availability
HFC	Hybrid Fiber Coax
HW	Hardware
IF	Intermediate Frequency
INA	Interactive Network Adapter (DVB Head-end)
ITU	International Telecommunication Union
MAC	Media Access Control
MTBF	Mean Time Between Failure
MTTF	Mean Time To Failure
MTTR	Mean Time To Repair
NIU	Network Interface Unit
NRA	National Regulatory Authority
OPTA	Onafhankelijke Post en Telecommunicatie Autoriteit
PSTN	Public Switched Telephone Network
RF	Radio Frequency
RIP	Routing Information Protocol
S/N	Signal To Noise Ratio
SC	System Controller of the CableDock 200
SNMP	Simple Network Message Protocol
SPF	Single Point of Failure
SW	Software
US	Upstream
VoIP	Voice over IP

Availability
The percentage of the time a user is able to use a system and its service with a certain service level. Uptime can be seen as an extra restriction of availability. This is because uptime requires a system to be fully available. In other words, with the normal (highest) service level. High Availability also is an extra restriction of Availability.

Graceful Degradation
Graceful Degradation can be described as the degraded level of service (higher than 0) that occurs when a system is not functioning in an optimal way. For example, a system can not only be up or down but also up but running at a slower speed or up but not supplying all of its services. When one is able to turn system failures that contribute to downtime into graceful degradation, one is able to extent the uptime of the system because now a user still is able to use (parts of) the system. In High Availability systems and in redundant systems this is a very common and preferable technique.

High Availability
High Availability (HA) [15] is defined as a characteristic of a system that is robust with respect to runtime events associated with failures and upgrades of hardware and software. Complete fault-tolerance, in which a system cannot have a Single Point of Failure that will potentially bring down the system, requires redundant hardware. HA, on the other hand, must provide flexibility that allows for runtime upgrades of hardware and software, as well as runtime debugging. HA systems are also charged with providing redundancy in software to prevent certain types of failures. A HA system provides extreme usability, functionality and flexibility without the added costs and limitations associated with a complete fault-tolerant system.

Inter Head-end Redundancy
Redundancy realized by head-ends that are able to take over each other.

Intra Head-end Redundancy
Redundancy within a single head-end. This means that parts of a single head-end are able to take over other (failing) parts of the same head-end.

MTBF
Mean Time Between Failures is the mean time expected between failures and is measured in hours [17]. It is a statistical value and is meant to be the mean over a long period of time and large number of units. For constant failure rate systems, MTBF is the inverse of the failure rate. Technically MTBF should be used only in reference to repairable items, while MTTF should be used for non-repairable items, but MTBF is commonly used for both repairable and non-repairable items.

MTTF
Mean Time To Failure is the mean time expected to the first failure of a piece of equipment [17]. It is a statistical value and is meant to be the mean over a long period of time and large number of units. For constant failure rate systems, MTTF is the inverse of the failure rate.

MTTR
Mean Time to Repair is defined to be total amount of time spent performing all corrective maintenance repairs divided by the total number of those repairs [17].

Redundancy
The provision of multiple interchangeable components to perform a single function in order to cope with failures and errors. Redundancy normally applies primarily to hardware. For example, one might install two or even three systems to do the same job. There are several ways these could be used. They could all be active all time thus giving extra performance through parallel processing as well as extra availability because a failing system can be taken over by the other systems. One could be active and the others simply monitoring its activity so as to be ready to take over if it failed (warm standby). Also, the spare hardware components could be kept turned off and only switched on when needed (cold standby).

Redundancy Functions
The different functions that perform the different tasks of the redundancy system. There are four different redundancy functions: the detection function, the information sharing function, the take over decision function and the take over activating function.

Redundancy Network
The total network of devices used for redundancy. For intra head-end redundancy these devices are the boards of a single head-end. They are interconnected by CompactPCI bus. For inter head-end redundancy these devices are the head-ends themselves. They are interconnected by the Ethernet management network.

Redundancy Messages
The messages used by the different redundancy systems to interact. There are three types of redundancy messages: detection messages, information sharing messages and take over activating messages. Note that for inter head-end redundancy these messages are IP over Ethernet based messages. For intra head-end redundancy we use the CompactPCI specification for signaling between different boards.

Redundancy States
The different states in which a redundancy system can be. These states are: the initialization state, the running state and the shutdown state. It is important to notice these states have nothing to do with the normal states of the CableDock 200.

Redundancy System
All software on a single device needed to participate in the redundancy network. Since there are two types of redundancy networks, there are two types of redundancy systems: an intra head-end redundancy system (the software on a single board) and an inter head-end redundancy system (the software on a single head-end).

Single Point of Failure
A Single Point of Failure (SPF) is used to describe an item that, if failed, would cause a failure of the complete system.

Uptime
The time during which a functional unit is fully operational. This time is measured from the functional unit point of view. HA systems are commonly associated with systems that have an uptime of '3 or more nines' meaning 99.9% uptime or better. In normal PSTN telecommunication systems an uptime of '5 nines' or 99.999% uptime is a normal feature.

Appendix B: FMEA analysis of an US board

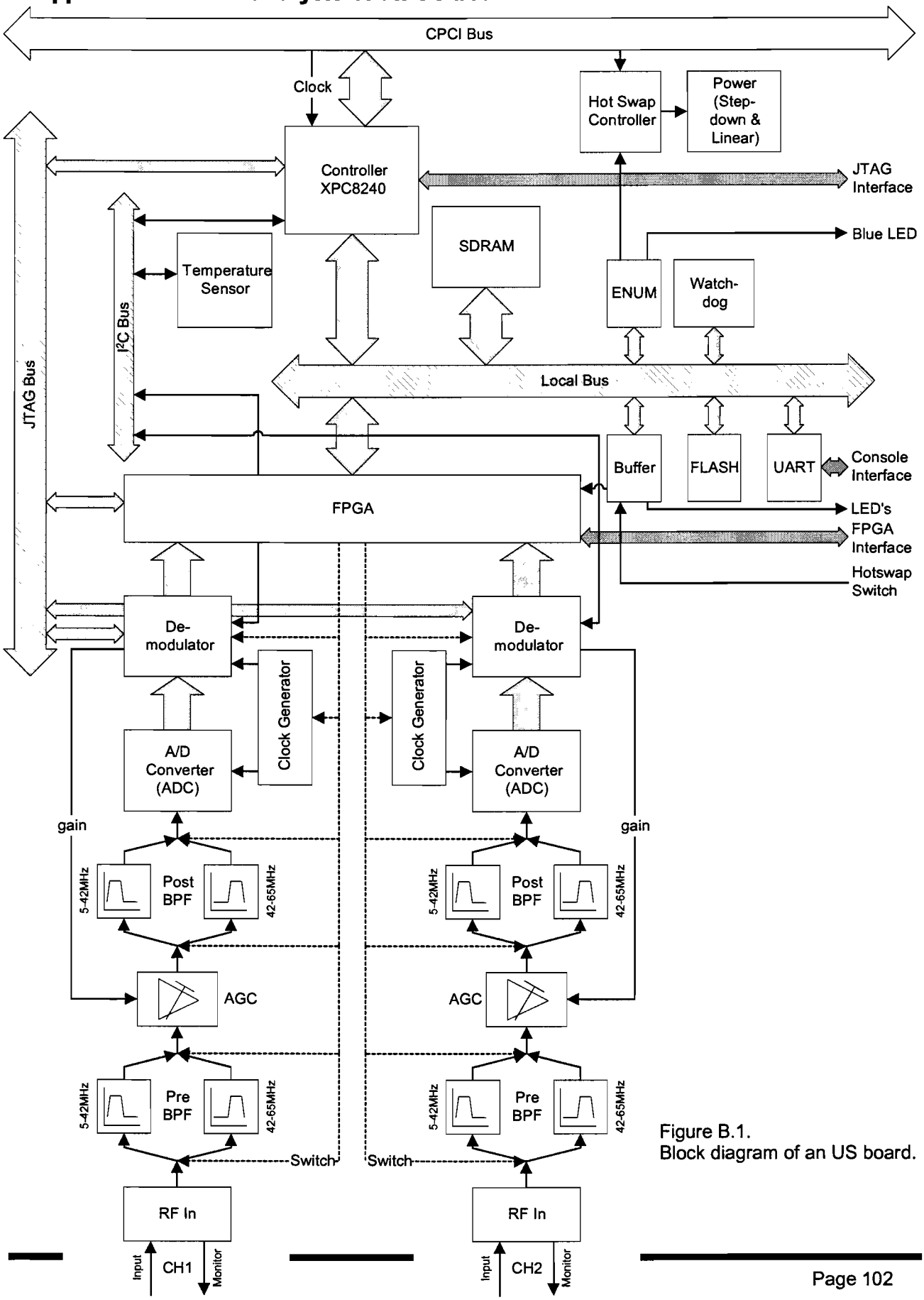

Figure B.1.
Block diagram of an US board.

Failure Mode and Effect Analysis (FMEA) [18] is a method to examine the effects of failures on the proper working of the product. Normally this analysis is used to examine the manufacturing process of the product. However, this method has been adopted for the (single) points of failure analysis. Normally this analysis is based upon three parameters: possibility, impact and detectability. For the sake of this analysis these parameters were slightly adjusted, resulting in the following parameters and the values of these parameters that describe their relative importance:

- Possibility. What is the possibility that an error occurs? This parameters is expressed by the following values:
  5 = a very high possibility. Most times this is a failure due to design errors.
  4 = a high possibility.
  3 = a normal possibility.
  2 = a small possibility. Under normal circumstances this failure will not occur.
  1 = a very small possibility. The possibility this failure will occur is almost nil.
- Impact. If the failure occurs, what is its impact on the proper working of the system? Note that the user point of view is used for the impact.
  5 = a very high impact. The complete head-end cannot function anymore. Therefore, this is a SPF. This failure affects all users connected to this head-end.
  4 = a high impact. The same as 5 only now the failure only affects a couple of users or just one user. Most times this is a failure on a single DS/US board or DS/US channel.
  3 = a normal impact.
  2 = a small impact. The failure has only influence on the long-term proper working of the system. In time it has to be replaced, but for now the system functions normally.
  1 = a very small impact. The failure does not have any influence on the proper working of the system.
- Reparability. How hard is it to repair the failure with the available hardware and software? Note that with hardware most times the hardware should be replaced and in practice this means that a new DS/US board has to be installed. This parameter does not say anything on the time it takes to repair the failure.
  5 = unreparable. In order to repair this failure, the complete head-end has to go down and the failing part has to be replaced.
  4 = unreparable. The failing part has to be replaced, only now this only troubles a part of the system.
  3 = repairable. A reset of the complete head-end will repair the failure.
  2 = repairable. Only a part of the head-end or just the failing part has to be reset.
  1 = very easy to repair. The available software and hardware can easily take over the failing part. Most times this is the case with redundant hardware and software.

With the use of Figure B.1 and the design documentation of the US board of the CableDock 200 [16] the different possible failures can be pointed out. Furthermore, values can be assigned to the FMEA parameters. The results are presented in table B.1.

The FMEA procedure requires sorting table B.1 by the product of the FMEA parameters in increasing order. This part of the analysis has been skipped because with the help of table B.1 already the point can be made why increasing the availability on module level is not a good option after all. This is explained in section 3.2.3.

Table B.1. Results of the FMEA analysis of an US board of the CableDock 200.

No.	Failure	Type	Reason	Result	Possi-bility	Impact	Repair-ability
1	RF input connector	HW	Bumped, connecting	No input	2	4	4
2	RF input bad connection	HW	Bad wires	Noise on input	2	3	4
3	RF monitor connector	HW	Bumped, connecting	No output	2	1	4
4	RF monitor bad connection	HW	Bad wires	Noise on output	2	1	4
5	RF in	HW	Wrong switch signal	Wrong pre bandpass filter	1	4	2 or 4
6	Pre bandpass filter	HW	Blown	To much noise	1	2	4
7	Amplifier (AGC)	HW	Blown	Signal level errors	1	3	4
8	Post bandpass filter	HW	Blown	Wrong S/N for ADC	1	2	4
9	Pre and Post bandpass filter	HW	Blown	To much noise	1	4	4
10	A/D converter (ADC)	HW	Blown	High BER	1	3 or 4	4
11	Clock generator	HW		Wrong sample frequency	1	4	4
12	Demodulator	HW	Design errors	No packets for FPGA	2	4	2
13	FPGA	HW	Timing errors	Packet loss	1	3	2
14	FPGA interface	HW	Bad header	No external programming	2	1	4
15	I^2C bus halts	HW	Demodulator	Cannot use bus	1	3	2
16	JTAG bus halts	HW		Cannot use bus	1	1	2 or 4
17	Local bus halts	HW		Cannot use bus	1	5	2 or 4
18	SDRAM	HW		Possible SW problems and data buffer unusable	1	5	2 or 4
19	Controller XPC8240	HW		Central processing	1	5	4
20	FLASH	HW		Bootloader out of order	1	1	4
21	Buffer	HW	Blown	No led output and hot swap switch unusable	1	1	4
22	UART (console) interface	HW	Blown / Bad connection	No direct communication	1	1	4
23	ENUM	HW	Blown	No blue led and hot swap	1	1	4
24	Watchdog	HW	Blown	No automatic failure recovery	1	1	4
25	Hot swap controller	HW	Blown	No power when hot swapping	1	5	4
26	Hot swap switch	HW		No hot swap possible	1	1	4
27	Power step-down converter	HW		2.5 V not available for FPGA and Controller	1	5	4
28	Power linear regulator	HW		1.8 V not available for FPGA and Controller	1	5	4
29	LED's	HW	Blown	No visual control	1	1	4
30	JTAG interface	HW	Bad header	No direct interface with Controller	1	1	4
31	Temperature sensor	HW		No temperature information	1	2	4
32	Control queue fills up	SW	Buffer not cleared	Possible buffer overflow	2	4	2
33	Data queue fills up	SW	Buffer not cleared	Possible buffer overflow	2	4	2
34	E-buffer[1] SC unavailable	SW	SC failure	Buffer overflow and packet loss	1	4	3
35	SW that disables interrupts halts	SW	No interrupts available	FPGA halts and packet loss	1	4	2
36	3 ms heartbeat mismatch	SW		Out-of-sync	1	4	2
37	Master DS board down	HW or SW		No 3 ms heartbeat	1	5	3 or 5
38	OSE kernel	SW		SW hangs	1	4	3

[1] See also references [2], [12] and [20].

Appendix C: Calculation of the sub quality factors

The calculations concerning head-end 2 (192.138.0.13)
Dec 3 starts with the definition of the input collections concerning head-end 2:

$DS_c = \{1,2\}$

$US_c = \{2.1,2.2\}$

Concerning switch 192.138.0.2 the following collections are defined:

$DS_{sw_c} = \{1\}$

$DS_{sw_f} = \{1\}$

$US_{sw_c} = \{2.2\}$

$US_{swf} = \{1.1,1.2,1.3\}$

Concerning switch 192.138.0.3 the following collections are defined:

$DS_{sw_c} = \{1,2\}$

$DS_{sw_f} = \{1\}$

$US_{sw_c} = \{2.1,2.2\}$

$US_{sw_f} = \{1.1,1.2,1.3\}$

Concerning the load, the following inputs are defined:

$load_f = 25+3+3+4 = 35$

$load_c = 0+20+2+2 = 24$

Now Dec 3 starts with the calculations of the sub quality factors. First the sub quality factor Q_{device_load}:

$$nload_c = \frac{24+35}{100} = 0.59 \qquad \Rightarrow Q_{device\_load} = 1$$

The sub quality factor Q_{DS_ch}:

$$Qch\_DS_{192.138.0.2} = \frac{1}{1} = 1 \qquad and \qquad Qch\_DS_{192.138.0.3} = \frac{2}{1} = 2 \xrightarrow{2>1} 1$$

$$\Rightarrow Q_{DS\_ch} = \frac{1+1}{2} = 1$$

The sub quality factor Q_{US_ch}:

$$Qch\_US_{192.138.0.2} = \frac{1}{3} \qquad and \qquad Qch\_US_{192.138.0.3} = \frac{2}{3}$$

$$\Rightarrow Q_{US\_ch} = \frac{1/3+2/3}{2} = 0.5$$

The sub quality factor Q_{DS_sw}:

$IN\_F_{192.138.0.2} = \{4\}$ $\Rightarrow OUT\_F_{192.138.0.2} = \{22\}$

$IN\_C_{192.138.0.2} = \{9\}$ $\Rightarrow OUT\_C_{192.138.0.2} = \{21,22,23,24,25,26,27,28,29,30,31,32\}$

$\Rightarrow OUT_{192.138.0.2} = \{22\}$ $\Rightarrow Qsw\_DS_{192.138.0.2} = \frac{1}{1} = 1$

$IN\_F_{192.138.0.3} = \{4\}$ $\Rightarrow OUT\_F_{192.138.0.3} = \{\}$

$IN\_C_{192.138.0.3} = \{6,7\}$ $\Rightarrow OUT\_C_{192.138.0.3} = \{21,22,23,24,25,26,27,28,29,30,31,32\}$

$\Rightarrow OUT_{192.138.0.3} = \{\ \}$ $\Rightarrow Qsw\_DS_{192.138.0.3} \xrightarrow{N(OUT\_F_{192.138.0.3})=0} 1$

$\Rightarrow Q_{DS\_sw} = \frac{1+1}{2} = 1$

The sub quality factor Q_{US_sw}:

$IN\_F_{192.138.0.2} = \{1,2,3\}$ $\Rightarrow OUT\_F_{192.138.0.2} = \{21,23,24\}$

$IN\_C_{192.138.0.2} = \{8\}$ $\Rightarrow OUT\_C_{192.138.0.2} = \{21,22,23,24,25,26,27,28,29,30,31,32\}$

$\Rightarrow OUT_{192.138.0.2} = \{21,23,24\}$ $\Rightarrow Qsw\_US_{192.138.0.2} = \frac{3}{3} = 1$

$IN\_F_{192.138.0.3} = \{1,2,3\}$ $\Rightarrow OUT\_F_{192.138.0.3} = \{\ \}$

$IN\_C_{192.138.0.3} = \{5,8\}$ $\Rightarrow OUT\_C_{192.138.0.3} = \{21,22,23,24,25,26,27,28,29,30,31,32\}$

$\Rightarrow OUT_{192.138.0.3} = \{\ \}$ $\Rightarrow Qsw\_US_{192.138.0.3} \xrightarrow{N(OUT\_F_{192.138.0.3})=0} 1$

$\Rightarrow Q_{US\_sw} = \frac{1+1}{2} = 1$

The sub quality factor Q_{DS_load}:

$load\_C_{192.138.0.2} = 0$

$mload\_C_{192.138.0.2} = 40$

$load\_F_{192.138.0.2} = 25$

$\Rightarrow nload\_C_{192.138.0.2} = \frac{25+0}{40} = 0.625$ $\Rightarrow Qload\_DS_{192.138.0.2} \xrightarrow{1/0.625>1} 1$

$load\_C_{192.138.0.3} = 0 + 20 = 20$

$mload\_C_{192.138.0.3} = 40 + 40 = 80$

$load\_F_{192.138.0.3} = 25$

$\Rightarrow nload\_C_{192.138.0.3} = \frac{25+20}{80} = 0.563$ $\Rightarrow Qload\_DS_{192.138.0.2} \xrightarrow{1/0.563>1} 1$

$\Rightarrow Q_{DS\_load} = 1 \cdot 1 = 1$

The sub quality factor Q_{US_load}:

$load\_C_{192.138.0.2} = 2$

$mload\_C_{192.138.0.2} = 6$

$load\_F_{192.138.0.2} = 3 + 3 + 4 = 10$

$\Rightarrow nload\_C_{192.138.0.2} = \frac{2+10}{6} = 2$ $\Rightarrow Qload\_US_{192.138.0.2} = \frac{1}{2}$

$load\_C_{192.138.0.3} = 2 + 2 = 4$

$mload\_C_{192.138.0.3} = 6 + 6 = 12$

$load\_F_{192.138.0.3} = 3 + 3 + 4 = 10$

$\Rightarrow nload\_C_{192.138.0.3} = \frac{4+10}{12} = 1.17$ $\Rightarrow Qload\_US_{192.138.0.2} = \frac{1}{1.17} = 0.857$

$\Rightarrow Q_{US\_load} = 0.5 \cdot 0.857 = 0.429$

The sub quality factor Q_{DS_CXN}:

$cxn\_C_{192.138.0.2} = 0$

$mcxn\_C_{192.138.0.2} = 2000$

$cxn\_F_{192.138.0.2} = 1000$

$\Rightarrow ncxn\_C_{192.138.0.2} = \frac{0+1000}{2000} = \frac{1}{2}$ $\Rightarrow Qcxn\_DS_{192.138.0.2} \xrightarrow{2/1>1} 1$

$cxn\_C_{192.138.0.3} = 0 + 1400 = 1400$

$mcxn\_C_{192.138.0.3} = 2000 + 2000 = 4000$

$cxn\_F_{192.138.0.3} = 1000$

$\Rightarrow ncxn\_C_{192.138.0.3} = \frac{1400+1000}{4000} = \frac{6}{10}$ $\Rightarrow Qcxn\_DS_{192.138.0.2} \xrightarrow{10/6>1} 1$

$\Rightarrow Q_{DS\_CXN} = 1 \cdot 1 = 1$

The sub quality factor Q_{US_CXN}:

$cxn\_C_{192.138.0.2} = 800$

$mcxn\_C_{192.138.0.2} = 2000$

$cxn\_F_{192.138.0.2} = 300 + 300 + 400 = 1000$

$\Rightarrow ncxn\_C_{192.138.0.2} = \frac{800+1000}{2000} = \frac{9}{10}$ $\Rightarrow Qcxn\_US_{192.138.0.2} \xrightarrow{10/9>1} 1$

$cxn\_C_{192.138.0.3} = 600 + 800 = 1400$

$mcxn\_C_{192.138.0.3} = 2000 + 2000 = 4000$

$cxn\_F_{192.138.0.3} = 300 + 300 + 400 = 1000$

$\Rightarrow ncxn\_C_{192.138.0.3} = \frac{1400+1000}{4000} = \frac{6}{10}$ $\Rightarrow Qcxn\_US_{192.138.0.2} \xrightarrow{10/6>1} 1$

$\Rightarrow Q_{US\_CXN} = 1 \cdot 1 = 1$

The sub quality factor $Q_{DS\_frequency}$:

$F_{192.138.0.2} = \{810\}$

$DS\_F_{810} = \{1\}$ *and* $DS\_C_{810} = \{1\}$

$\Rightarrow Q_{810} = \frac{1}{2 \cdot 1} + 0.5 = 1$ $\Rightarrow Qf\_DS_{192.138.0.2} = 1$

$F_{192.138.0.3} = \{810\}$

$DS\_F_{810} = \{1\}$ *and* $DS\_C_{810} = \{1,2\}$

$\Rightarrow Q_{810} = \frac{2}{2 \cdot 1} + 0.5 = 1.5 \xrightarrow{1.5>1} 1$ $\Rightarrow Qf\_DS_{192.138.0.3} = 1$

$\Rightarrow Q_{DS\_frequency} = 1 \cdot 1 = 1$

The sub quality factor $Q_{US\_frequency}$:

$F_{192.138.0.2} = \{13,16\}$

$US\_F_{13} = \{1.2,1.3\}$ *and* $US\_C_{13} = \{2.2\}$

$\Rightarrow Q_{13} = \frac{1}{2 \cdot 2} + 0.5 = 0.75$

$US\_F_{16} = \{1.1\}$ *and* $US\_C_{16} = \{\}$

$\Rightarrow Q_{16} = \frac{0}{2 \cdot 1} + 0.5 = 0.5$ $\Rightarrow Qf\_US_{192.138.0.2} = 0.75 \cdot 0.5 = 0.375$

$F_{192.138.0.3} = \{13,16\}$

$US\_F_{13} = \{1.2,1.3\}$ and $US\_C_{13} = \{2.2\}$

$\Rightarrow Q_{13} = \frac{1}{2 \cdot 1} + 0.5 = 0.75$

$US\_F_{16} = \{1.1\}$ and $US\_C_{16} = \{2.1\}$

$\Rightarrow Q_{16} = \frac{1}{2 \cdot 1} + 0.5 = 0.75$ $\Rightarrow Qf\_US_{192.138.0.3} = 0.75 \cdot 0.75 = 0.563$

$\Rightarrow Q_{US\_frequency} = \sqrt{0.375 \cdot 0.563} = 0.459$

The sub quality factor $Q_{DS\_modulation}$:

$MT_{192.138.0.2} = \{QAM64\}$

$DS\_F_{QAM64} = \{1\}$ and $DS\_C_{QAM64} = \{1\}$

$\Rightarrow Q_{QAM64} = \frac{1}{2 \cdot 1} + 0.5 = 1$ $\Rightarrow Qmt\_DS_{192.138.0.2} = 1$

$MT_{192.138.0.3} = \{QAM64\}$

$DS\_F_{QAM64} = \{1\}$ and $DS\_C_{QAM64} = \{1,2\}$

$\Rightarrow Q_{QAM64} = \frac{2}{2 \cdot 1} + 0.5 = 1.5 \xrightarrow{1.5>1} 1$ $\Rightarrow Qmt\_DS_{192.138.0.3} = 1$

$\Rightarrow Q_{DS\_modulation} = 1 \cdot 1 = 1$

The sub quality factor $Q_{US\_modulation}$:

$MT_{192.138.0.2} = \{QPSK\}$

$US\_F_{QPSK} = \{1.1,1.2,1.3\}$ and $US\_C_{QPSK} = \{2.2\}$

$\Rightarrow Q_{QPSK} = \frac{1}{3}$ $\Rightarrow Qmt\_US_{192.138.0.2} = \frac{1/3}{1} = 0.333$

$MT_{192.138.0.3} = \{QPSK\}$

$US\_F_{QPSK} = \{1.1,1.2,1.3\}$ and $US\_C_{QPSK} = \{2.1,2.2\}$

$\Rightarrow Q_{QPSK} = \frac{2}{3}$ $\Rightarrow Qmt\_US_{192.138.0.3} = \frac{2/3}{1} = 0.667$

$\Rightarrow Q_{US\_modulation} = \frac{0.333 + 0.667}{2} = 0.5$

The sub quality factor Q_{DS_symbol}:

$SR_{192.138.0.2} = \{8\}$

$DS\_F_{8} = \{1\}$ and $DS\_C_{8} = \{1\}$

$\Rightarrow Q_{8} = \frac{1}{1} = 1$ $\Rightarrow Qsr\_DS_{192.138.0.2} = 1$

$SR_{192.138.0.3} = \{8\}$

$DS\_F_{8} = \{1\}$ and $DS\_C_{8} = \{1,2\}$

$\Rightarrow Q_{8} = \frac{2}{1} = 2 \xrightarrow{2>1} 1$ $\Rightarrow Qsr\_DS_{192.138.0.3} = 1$

$\Rightarrow Q_{DS\_symbol} = \frac{1+1}{2} = 1$

The sub quality factor Q_{US_symbol}:

$SR_{192.138.0.2} = \{6\}$

$US\_F_6 = \{1.1, 1.2, 1.3\}$ and $US\_C_6 = \{2.2\}$

$\Rightarrow Q_6 = \frac{1}{4 \cdot 3} + 0.75 = 0.833$ $\Rightarrow Qsr\_US_{192.138.0.2} = 0.833$

$SR_{192.138.0.3} = \{6\}$

$US\_F_6 = \{1.1, 1.2, 1.3\}$ and $US\_C_6 = \{2.1, 2.2\}$

$\Rightarrow Q_6 = \frac{2}{4 \cdot 3} + 0.75 = 0.917$ $\Rightarrow Qsr\_US_{192.138.0.3} = 0.917$

$\Rightarrow Q_{US\_symbol} = 0.833 \cdot 0.917 = 0.764$

The sub quality factor Q_{DS_power}:

$PW_{192.138.0.2} = \{90\}$

$DS\_F_{90} = \{1\}$ and $DS\_C_{90} = \{1\}$

$\Rightarrow Q_{90} = \frac{1}{1} = 1$ $\Rightarrow Qpw\_DS_{192.138.0.2} = \frac{1}{1} = 1$

$PW_{192.138.0.3} = \{90\}$

$DS\_F_{90} = \{1\}$ and $DS\_C_{90} = \{1, 2\}$

$\Rightarrow Q_{90} = \frac{2}{1} = 2 \xrightarrow{2>1} 1$ $\Rightarrow Qpw\_DS_{192.138.0.3} = \frac{1}{1} = 1$

$\Rightarrow Q_{DS\_power} = \frac{1+1}{2} = 1$

The sub quality factor Q_{US_delay}:

$RTD_{192.138.0.2} = \{0\}$

$US\_F_0 = \{1.1, 1.2, 1.3\}$ and $US\_C_0 = \{2.2\}$

$\Rightarrow Q_0 = \frac{1}{3} = 0.333$ $\Rightarrow Qrtd\_US_{192.138.0.2} = \frac{0.333}{1} = 0.333$

$RTD_{192.138.0.3} = \{0\}$

$US\_F_0 = \{1.1, 1.2, 1.3\}$ and $US\_C_0 = \{2.1, 2.2\}$

$\Rightarrow Q_0 = \frac{2}{3} = 0.667$ $\Rightarrow Qrtd\_US_{192.138.0.3} = \frac{0.667}{1} = 0.667$

$\Rightarrow Q_{US\_delay} = \frac{0.333 + 0.667}{2} = 0.5$

The calculations concerning head-end 3 (192.138.0.38)

Dec 3 starts with the definition of the input collections concerning head-end 3:

$DS_c = \{1\}$

$US_c = \{1.1, 1.2, 1.3\}$

Concerning switch 192.138.0.2 the following collections are defined:

$DS_{sw_c} = \{1\}$

$DS_{sw_f} = \{1\}$

$US_{sw_c} = \{1.1, 1.2\}$

$US_{sw_f} = \{1.1, 1.2, 1.3\}$

Concerning switch 192.138.0.3 the following collections are defined:

$DS_{sw_c} = \{1\}$

$DS_{sw_f} = \{1\}$

$US_{sw_c} = \{1.1,1.2,1.3\}$

$US_{sw_f} = \{1.1,1.2,1.3\}$

Concerning the load, the following inputs are defined:

$load_f = 25+3+3+4 = 35$

$load_c = 35+1+2+4 = 42$

Now Dec 3 starts with the calculations of the sub quality factors. Since these calculations are the same as for head-end 2, only the results are showed. The resulting sub quality factors of head-end 3 are:

$Q_{device\_load} = 1$

$Q_{DS\_ch} = 1$	*and*	$Q_{US\_ch} = 0.833$
$Q_{DS\_sw} = 1$	*and*	$Q_{US\_sw} = 1$
$Q_{DS\_load} = 0.444$	*and*	$Q_{US\_load} = 0.923$
$Q_{DS\_CXN} = 0.246$	*and*	$Q_{US\_CXN} = 1$
$Q_{DS\_frequency} = 0.25$	*and*	$Q_{US\_frequency} = 0.25$
$Q_{DS\_modulation} = 1$	*and*	$Q_{US\_modulation} = 0.833$
$Q_{DS\_symbol} = 1$	*and*	$Q_{US\_symbol} = 0.917$

$Q_{DS\_power} = 1$

$Q_{US\_delay} = 0.833$

The calculations concerning US board 1.

Dec 3 starts with the definition of the input collections concerning US board 1. Since the failing device is an US board, only the calculations concerning US channels have to be done:

$US_c = \{1.1\}$

Concerning switch 192.138.0.3 the following collections are defined:

$US_{sw_c} = \{1.1\}$

$US_{sw_f} = \{1.2\}$

Now Dec 3 starts with the calculations of the sub quality factors. Since these calculations are similar to the calculations for head-end 2 as presented earlier this appendix, only the results are showed. The resulting sub quality factors of US board 1 are:

$Q_{US\_ch} = 1$	*and*	$Q_{US\_sw} = 1$
$Q_{US\_load} = 1$	*and*	$Q_{US\_CXN} = 1$
$Q_{US\_frequency} = 1$	*and*	$Q_{US\_modulation} = 1$
$Q_{US\_symbol} = 1$	*and*	$Q_{US\_delay} = 1$

The calculations concerning US board 3.
Dec 3 starts with the definition of the input collections concerning US board 3. Again, since the failing device is an US board, only the calculations concerning US channels have to be done:

$US_c = \{1.3\}$

Concerning switch 192.138.0.3 the following collections are defined:

$US_{sw_c} = \{1.3\}$

$US_{sw_f} = \{1.2\}$

The calculations are similar to the ones already presented. Therefore, only the results are showed. These resulting sub quality factors of US board 3 are:

$Q_{US\_ch} = 1$	*and*	$Q_{US\_sw} = 1$
$Q_{US\_load} = 1$	*and*	$Q_{US\_CXN} = 1$
$Q_{US\_frequency} = 1$	*and*	$Q_{US\_modulation} = 1$
$Q_{US\_symbol} = 0.75$	*and*	$Q_{US\_delay} = 1$