CHAPTER V
IDENTIFICATION &
CLASSIFICATION OF
KPA'S IN NETWORK
MANAGEMENT
Identification and Classification of Key Process Area's in Network Management

The purpose of this chapter is to identify and classify potential management functions. To find management functions, the network's design process will be considered. Within such process the designer elaborates the network's primary functions. As will be demonstrated in this chapter, this elaboration introduces a number of management problems. To solve such problems, management functions must be introduced.

There are several models that describe the design process. In this chapter the step-wise design model will be considered. This model is easy to understand and allows the identification and classification of management functions in a structured way. Three classes of management functions will be introduced, each class belonging to a particular phase of the design process.

The structure of this chapter is as follows.

- Section 5.1 provides an introduction to step-wise design. This section recognizes three important phases: the architectural phase, the implementation phase and the realization phase.

- Section 5.2 discusses management functions that can already be identified during the architectural phase. Since the outcome of this phase will be a service specification, the term 'service management' will be introduced to denote these functions.

- Section 5.3 discusses management functions that can be identified during the implementation phase. To denote these functions, the term 'protocol management' will be introduced.

- Section 5.4 discusses management functions that can be identified during the realization phase. The outcome of this phase are individual network systems, such as switches and terminals. In the telecommunication world these systems are commonly called 'network elements'. The term 'element management' will therefore be introduced to denote these management functions.
• Section 5.5 shows that the distinction between service, protocol and element management may not only be useful to classify management functions during the design phase, but may also be used to structure the various management operations during the operational phase. As will be shown, such structured approach may be beneficial for fault detection purposes.

• Section 5.6 provides the conclusions.

5.1 Introduction to step-wise design

The starting point of a step-wise design process is the definition of user requirements. In most cases these requirements are written in a natural language without any particular structure. According to software Engineers "user requirements may be incomplete and inarticulate and insofar they are recorded they may be vague, imprecise, ambiguous, redundant and sometimes even contradictory" [Pressman].

![Diagram of Step-Wise Design Process]

Figure. 5.1 : -Step-Wise Design Process
The user requirements are transformed via a number of steps into a realization (Figure 5.1). There is no general rule stating how many steps are needed: that decision is made by the designer and depends amongst others upon the complexity of the design.

5.1.1 Phases in a step-wise design

Although the exact number of design steps varies from case to case, several theories [Pressman R.S. :" Software Engineering A Practitioners Approach", McGraw Hill publishing] propose to distinguish between the following phases (Figure 5.2):

- Architectural phase: this phase starts from the user requirements and results into the intermediate specification called the 'architecture'.

- Implementation phase: this phase starts from the architecture and results into one or more implementations. At this level, the internal structure of the system is defined. The fact that a single architecture can be implemented in different ways, is shown in the figure by means of multiple arrows. A good example that multiple implementations can be derived from a single architecture, is given by Digital Equipment. This company has developed many implementations (e.g. 11/03, 11/04, 11/34, 11/60, 11/70) from the single PDP-11 minicomputer architecture.

- Realization phase: this phase starts from the implementation and results into one or more realizations. At this level, functions are mapped upon hardware and software components. There are many possibilities to perform these mappings: functions may for instance be implemented in TTL logic, VLSI chips, micro controllers etc.

The realization phase is the last phase of the design process; after the realization is ready the operational phase may be started.
Figure. 5.2: - Phases in the Design Process
5.1.2 Step-wise design and distributed systems

The theory that was described in the previous subsection can be applied to the specific case of distributed systems design (Figure 5.3) [Pressman R.S. Software Engineering A Practitioner's Approach", McGraw Hill publishing]. In such case the phases are usually described as follows:

- Architectural phase: this phase starts from the user requirements and results into the intermediate specification called the 'service'. The service defines the common behavior of users and provider; it hides the internal structure of the provider from the user. This implies that, at architectural level, the provider is considered as a black-box with a number of interaction points.

![Diagram of design phases]

Figure 5.3: Phases in the design of distributed systems
• Implementation phase: this phase starts from the service specification and results into protocol specifications. At this level, the internal structure of the distributed system is defined. This can be done in a number of 'horizontal and vertical structuring steps'.

• Realization phase: this phase starts from the protocol specifications and results into one or more realizations. At this level, protocol functions are implemented in hard- and software. We may expect that different manufacturers follow different paths towards their realizations. From a single protocol definition different realizations are therefore possible.

5.2 Management issues in the architectural phase

To support the primary functions that are defined during the architectural phase, it may be necessary to introduce some management functions. To give an idea of such management functions, Subsection 5.2.1 provides some examples. After an understanding of these management functions has been obtained, Subsection 5.2.2 discusses some general characteristics. To distinguish these management functions from other management functions, Subsection 5.2.3 defines the concept of service management. The section concludes with some general remarks concerning complexity and standardization of service management (Subsection 5.2.4).

5.2.1 Examples

Management functions may be introduced in the architectural phase to initialize and modify service characteristics, but also to obtain information from the service provider.

Initialization

Services, as defined by standardization organizations, generally describe the possibility to exchange user data between SAPs. The way in which individual SAP addresses should be assigned, is usually not described by these service definitions. Such initialization functions can be considered as management functions.

Modification

Service definitions usually concentrate on primary functions and ignore the question of how to connect and remove users from the provider. That question is usually considered to be a management problem.
There are several reasons why users should be connected or removed from the network. Connection to the network may be demanded in case new users appear. Removal from the network may be necessary in case users refuse payment of their bills or show some other form of misbehaviour (e.g. hackers).

In the case of networks for mobile communications, it is also conceivable that users request temporary removal of themselves. Such requests may be a matter of money: even in case users do not participate in connections, they may be charged. This is because networks for mobile communications exchange user specific signalling information, even in case no connections are made.

Obtaining information

Management functions may, for example, also be introduced to monitor the QoS. QoS figures can be useful to determine whether sufficient capacity is available to connect new users or to determine that QoS has dropped below the negotiated level, in which case billing may be adapted.

5.2.2 General characteristics

The examples in the previous subsection showed that service definitions usually include descriptions of the primary functions, but lack descriptions of management functions. Since primary functions specify only parts of the provider's behaviour, management functions should be added to complete the description of this behaviour (Figure 5.4).

The management functions that can be identified during the architectural phase, have a direct relationship to the user requirements. This can be understood if one considers the problems that lead to the introduction of these management functions:

![Diagram](image)

**Figure 5.4: Complete description of the provider's behaviour**
Networks are usually not designed as dedicated systems that satisfy the requirements of one specific set of users, but as general purpose systems that have the potential to support different sets of users. Before a network can be used by a certain set of users, it must be initialized. The purpose of such initialization is to adjust the network's behaviour to the requirements of a particular set of users. The term 'provisioning' is sometimes used for such initialization. Tasks that may be performed during the initialization phase are: the connection of users to the network, the assignment of SAP addresses, the definition of group addresses, the establishment of closed user groups etc.

User requirements usually change in time. Tasks that may be performed to cope with these changing requirements are: the connection of new users to the network, the removal of existing users from the network, the modification of group addresses and closed user groups, the modification of cost parameters etc.

Users and provider may not always behave in the way that has been negotiated. Functions must therefore be added to ease the detection of potential misbehaviour. Examples of such functions are: monitoring of activities that take place at a certain SAP, request QoS figures, perform reachability tests etc.

It may be noted that these three reasons correspond to what has been discussed in the subsection on 'why management' (Subsection 1.2) under the headings: cost reduction, flexibility and faults.

In the architectural phase no knowledge is needed of the provider's internal structure. This implies that also for the identification of management functions the provider may be considered as a black-box. The identification of management functions will for instance not be effected by the choice whether the provider should internally use the OSI ConnectionLess Network Protocol (CLNP) or the Internet Protocol (IP).

As a consequence, the management functions that may be identified during the architectural phase can not be used to manipulate individual components within the provider, such as for example routing tables.

5.2.3 Definition of service management

To distinguish the management functions that are identified in the architectural phase from other management functions, the term 'service management' is introduced.
Definition: Service management is that part of management that is responsible for the proper interaction between users and provider. The object of service management is the service provider. Service management functions can be initiated by service users and allow the initialization, modification and observation of the provider's behaviour.

The OSI and Internet management architectures have not defined the service management concept, even though some of their MIBs include objects that can be used to manage the interactions between user and service provider.

As opposed to OSI and Internet, the TMN management architecture did define service management (see Section 3.4). Still there are certain differences between service management as defined in this theses, and service management as defined by TMN. These differences come from the fact that this thesis uses, in accordance to the OSI Reference Model, the term 'service' in a specific sense. As opposed to TMN who uses this term in a general sense, this thesis does for example not always regard applications and information servers as services.

5.2.4 Concluding remarks

The fact that service management considers the service provider as a blackbox implies that service management should be easier to comprehend than the management functions that can be identified during the implementation and realization phase.

Because service management can be defined independent of a particular implementation or realization, the possibility exists to develop service management standards.

This section concludes with a remark concerning a potential misinterpretation of the term 'service management'. Consider the example of an implementor who decides to add functions within the provider that monitor the provided QoS for the purpose of optimizing the routing tables. If these monitoring functions are performed invisible for the service users (e.g. there is no possibility for users to manipulate these functions) these functions should (according to the definition) not be considered as service management functions.

5.3 Management issues in the implementation phase

The aim of the implementation phase is to develop one or more protocol definitions. These definitions usually describe the primary functions that must be performed by
protocol entities. To support these functions, it will generally be necessary to add management functions.

To illustrate which management functions may be added, Subsection 5.3.1 starts with some examples. Characteristics of these management functions will be presented in Subsection 5.3.2. To distinguish these management functions from other management functions, Subsection 5.3.3 defines the concept of protocol management. Subsection 5.3.4 discusses the relation between protocol management and service management.

5.3.1 Examples

The management functions that can be identified during the implementation phase, are needed to support the primary protocol functions. In this subsection some primary functions of a hypothetical network layer protocol will be considered to identify examples of management functions.

Assume the network layer protocol performs the following primary functions:

- Forwarding. In routers the forwarding function determines the outgoing link over which packets should be forwarded. The implementation of this function can

![Diagram of Forwarding within a network layer entity](image)

Figure 5.5: Forwarding within a network layer entity
make use of forwarding tables (Figure 5.5). The destination address which is contained in each packet, serves as input to this table. The table is searched, until an entry is found that matches this destination address. This entry associates the destination address with an outgoing link.

Protocol definitions usually do not specify how to initialize and maintain the forwarding table. Nevertheless initialization and maintenance of this table is necessary to allow the network to become and remain operational. How this should be done, is often considered to be a management problem.

• Flow control. A good example of a flow control mechanism, is the window mechanism. This mechanism uses a 'maximum window size' parameter, which must be set by management. How this should be done, is usually not specified by the protocol definition.

• Congestion control. In case of congestion, it may be a good idea to drop certain packets. When and which packets should be dropped, is usually not specified by the protocol definition. It is the responsibility of management to solve this question.

• Segmentation and reassembly. The segmentation function may be activated in case the size of a network layer PDU exceeds the maximum that is supported over an outgoing link. The implementation of this function may be based upon a table that contains for each outgoing link the maximum size that is supported. Initialization and maintenance of this table may be considered as a management function; protocol definitions usually do not specify this function.

• Error correction. A possible error correction mechanism is retransmission. Examples of variables that may be used by this mechanism, are the time-out variable and the variable that indicates the maximum number of retransmission attempts. Setting these variables is usually considered to be a management responsibility.

The examples demonstrated the need to extend primary protocol functions with functions to initialize and modify variables. It may also be necessary, however, to introduce management functions to obtain the values of protocol variables. Example: Users may request that packets be transferred over the links that show the lowest error rates. In this case management, thus the functions that maintain the forwarding table, must be able
to obtain error values for each possible link.

5.3.2 General characteristics

The management functions that are added to support the primary protocol functions have the following general characteristics:

- They can be identified as part of the design process of the primary protocol functions, thus during the implementation phase.

- They affect the operation of protocol entities (this is a consequence of the fact that the primary protocol functions are performed by protocol entities). In fact, the management functions initialize, modify and observe information within these entities.

- The management functions belong to the same protocol layer as the primary protocol functions. This is a logical consequence of the fact that for the definition of these management functions detailed knowledge of the primary protocol functions is needed. Such knowledge is not available at higher or lower protocol layers.

- The management functions can be described in a way that abstracts from possible realizations. It is therefore possible to standardize these management functions.

As opposed to service management, protocol management deals with problems that relate to the internal structure of the provider. These problems may be much more complex than the problems service management is faced with.

5.3.3 Definition of protocol management

To distinguish the management functions that can be identified in the implementation phase from other management functions, the term 'protocol management' is introduced.

Definition: Protocol management is that part of management that is responsible for the proper operation of a particular protocol. Protocol management functions support the primary protocol functions. Protocol management functions allow the initialization, modification and observation of entities within a protocol layer.

Most of the current management literature is on protocol management. Protocol management is defined by OSI, TMN and Internet.
5.3.4 Relation with service management

The implementation relation that exists between service and protocol functions is shown in Figure 5.3. An interesting question is whether a similar implementation relation exists between service management and protocol management: should protocol management functions be regarded as the implementation of service management functions or not?

To answer this question, two more general questions will be considered:

• How can primary service functions be implemented?

• How can service management functions be implemented?

Implementation of primary service functions

Primary service functions may be mapped upon primary protocol functions, but also upon protocol management functions. This can be explained by means of the following example.

Consider the primary service function that describes how to exchange user data between SAPs. During the implementation of this function, forwarding functions may be

![Diagram](image-url)

Figure 5.6: Implementation of Primary Service functions
introduced. Such forwarding functions can be considered as primary protocol functions. As explained earlier, forwarding functions may make use of forwarding tables. To initialize and maintain these tables, protocol management functions must be added too.

The implementation of primary service functions thus involves primary protocol functions as well as protocol management functions.

Implementation of service management functions

Service management functions can also be implemented in different ways. To demonstrate this, two examples will be given:

• In the first example, the service management functions will be mapped upon primary protocol functions.

• In the second example, the same service management functions will be implemented in a different way and mapped upon protocol management functions.

Figure 5.7: Service Providers

The service management function that will be used for both examples, is modification of SAP addresses. Consider the case of a service provider with three SAPs: a, b and c (Figure 5.7). Assume the address b should be changed into x.

First example:

Suppose the service provider can be decomposed into three protocol entities (one per SAP) plus an underlying service provider with broadcast capabilities (Figure 5.8). This structure allows for a straightforward implementation of the service management function: before b changes the SAP address, it broadcasts a special Check PDU to verify the uniqueness of the new address. If the other entities do not raise objections, b carries out the actual address change and notifies the other entities of this change via (for instance) a Commit PDU.
With this kind of implementation, the protocol function that takes care of the address change can be developed in isolation from other protocol functions. It will therefore be pointless to associate this address change function to any of the primary protocol functions; the function that changes the SAP address is not needed to support any of the primary protocol functions. This implies that the function that changes the SAP address does not satisfy the requirements of protocol management. The function should therefore be regarded as a primary protocol function.

Second example:

Suppose the service provider can be decomposed into three End Systems (ES) plus one Intermediate System (IS). Each ES is connected to the IS via a point-to-point link. The IS performs a forwarding function and uses a forwarding table. In case the address b is changed into x, the forwarding table should be adjusted. As discussed previously, modification of the forwarding table may be considered as a protocol management function.

In this case, the implementation of the service management function involves the addition of protocol management functions.
Conclusion:

The two examples above indicated that service management functions can be implemented as primary protocol functions, as well as protocol management functions (Figure 5.6). Thus there is not necessarily a one to one relationship between service management and protocol management functions.

Figure 5.9: Implementation involves modification of the forwarding table

Figure 5.10: Implementation of service management functions
5.4 Management issues in the realization phase

To allow observation, initialization and modification of the components that realize individual network systems, management functions may be needed. To give an idea of the realization components that should be managed, Subsection 5.4.1 starts with some examples. The examples are followed in Subsection 5.4.2 by a discussion of some general characteristics of these management functions. In Subsection 5.4.3 the concept of element management will be defined.

5.4.1 Examples

Individual network systems can be realized in hardware and/or in software. Both realization forms introduce their own management problems. Hardware implementations require management functions to observe, initialize and modify hardware modules.

Items that are observed in many hardware implementations, are power lines. The idea behind this is that changes in power consumption may indicate failure of hardware components. Since hardware failures are more likely with increasing temperatures, several hubs already include functions that read the current temperature and inform the manager in case this temperature exceeds a pre-defined level.

A common initialization task is to allocate interrupt vectors and I/O addresses.

Modifications may involve replacement of hardware boards, but also the update of firmware. Flash memories are nowadays widely used to accommodate such firmware changes.

Software management also involves observation, initialization and modification. In case network software is implemented on top of a multi-processing operating system (e.g. UNIX), the following items may be observed: process status, process priority, CPU time, foreground memory, swap space etc.

Some of these items require some initial assignment and may also be modified after the system has become operational. Examples of these are priority and memory use.

5.4.2 General characteristics

A primary protocol may be realized by different manufacturers in different ways. Realizations are thus vendor specific. This implies that also management of these realizations will be vendor specific. As a result, standardization should be restricted
to some common parts, such as how to structure and transfer management information. The decision which items can be managed (e.g. the contents of the MIB), can only be taken by the individual manufacturers.

Agreement upon such MIBs is unlikely, since the differences in these MIBs allow manufacturers to distinguish their products and thus obtain a higher share of the market.

It is interesting to note that the IETF has recognized the need to develop enterprise specific MIBs. For this purpose, the IETF has included a dedicated branch in the naming tree. A special name for this form of management has not been defined by the IETF, however.

5.4.3 Definition of element management

To distinguish the management functions that can be identified during the realization phase from other management functions, the term 'element management' will be used.

Definition: Element management is that part of management that is responsible for the proper operation of the realization components that are used within individual network systems. Element management functions include the initialization, observation and modification of these physical components.

The concept of element management has also been defined by TMN; the ISO and Internet management architectures did not define this concept. In TMN, the concept is called: 'network element management'.

5.4.4 Relation with protocol management

Analogous to service and protocol management, there need not necessarily be a one to one relationship between protocol and element management. Primary as well as protocol management functions may therefore be realized as primary, but also as element management components.

5.5 Relevance for the operational phase

In the previous sections, three classes of management functions were identified: service, protocol and element management. Each of these classes relates to one of the three stages of the step-wise design process. In this section it will be demonstrated that the distinction between service, protocol and element management may not only be relevant during the design phase, but that this distinction may in some cases also
be useful during the operational phase. In this section it is assumed that a layered architecture has been used for the design of the network.

To explain how service, protocol and element management can be applied in the operational phase, a structured approach to fault detection will be presented. It should be noted that this approach is not intended to replace traditional fault management approaches, but to supplement traditional approaches; it is for instance still desirable to monitor the operation of the crucial network components on a regular basis.

Figure 5.11: Structured approach to fault management

This structured approach to fault detection can be illustrated as follows:

1) Use service management to find the protocol layer that causes the problem.

2) Use protocol management to find the (sub)system within that layer that causes the problem.

3) Use element management to find the realization component within that system that causes the problem (Figure 5.11).
Step 1: use service management

According to the structured approach, the manager should first trace the protocol layer that causes the problem. This can be done by checking the behaviour of each of the underlying service providers in succession. Such checking can be performed without knowledge of the provider's internal structure; it can therefore be based upon service management. If the outcome of the check shows that the provider performs well, the problem must be caused by a higher layer protocol.

The provider that is checked first (see note below), may be the one that is located immediately below the user's application. The outcome of this check may be:

- The provider performs well. Lower level service providers need therefore not be investigated and the problem must be caused by the protocol layer immediately above this provider. Step 1 can be closed.

- The provider does not perform well. The problem is thus located within this service provider. Step 1 repeats, but this time the next lower service provider will be checked.

Figure 5.12: Find protocol layer that causes problems

Figure 5.12 shows the successive checking of underlying service providers. The loop stops after a service provider is found that performs well. The protocol that uses this underlying provider must be the one that caused the problem and will be investigated in step 2.
Note: step 1 starts with the service provider immediately below the user's application, and not the lowest service provider. This can be understood by realizing that additional knowledge may be required to determine the lowest (e.g. physical and data link) service providers that are used by the application. Obtaining this knowledge may not be easy, since it requires insight in the network's topology and an understanding of the algorithm that is used to forward user data. The approach that requires the least amount of knowledge is therefore the one that start from the highest level service provider, thus the one located immediately below the application.

Step 2: use protocol management

After the protocol layer that causes the problem has been determined, the specific entity that causes the problem should be determined. In case the protocol is a point-to-point protocol, only two entities must be checked. In case of a network layer protocol, all entities between source and destination must be checked. To find these entities, it may be necessary to consult the forwarding tables or use possible 'route recording' functions.

Step 3: use element management

After the entity has been found that causes the problem, the components that realize this entity must be checked. To perform such check, knowledge of the specific realization must be available.

Example

Consider the case of an e-mail problem. Figure 5.13 shows the successive steps that can be performed by the manager to solve such problem. The figure assumes that the problem is caused by a malfunctioning component within one of the network routers.
Notification of e-mail Problems Between Two Users

**Step 1**
Check Underlying Transport Service Provider

**Provider Causes**

**Step 2**
Check Underlying Network Service Provider

**Provider Causes**

**Step 3**
Determine & Check Underlying Data Link Service Providers

**Providers Are**

**Network Buyer Causes**

**OK**

**Problems**

**Step 2**
Determine And Check Relevant Network Layer Entities

**Malfunctioning Router**

**Determined**

**Step 3**
Determine And Check Components That Realize This

**Malfunctioning Component**

**Determined**

**Correct Error**

Figure 5.13: Example of fault management
OSI and Internet management do not recognize the idea of service management. As a consequence, these approaches do not support step 1 of our structured approach. Instead, the current OSI and Internet MIBs confront managers immediately with all details of the protocols (see also Subsection 4.4.2).

5.6 Conclusions

This chapter showed that management problems could be identified within all stages of the step-wise design process. This conclusion is interesting, since the idea to define management functions at service level has not been recognized by the OSI and Internet management groups.

In this chapter, three classes of management functions were identified: service, protocol and element management. The distinction between these classes allowed us to improve our understanding of (Figure 5.14):

- Which management issues arise at which stage of the design.
- Which management issues are visible to the network users, and which issues require knowledge of the provider's internal structure.
- Which management issues are vendor specific.
- Which management issues may, and which management issues may not be standardized.

<table>
<thead>
<tr>
<th>Design step</th>
<th>Visible to user</th>
<th>Vendor</th>
<th>Standardization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Protocol</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Element</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Figure 5.14: Differences between service, protocol and element management

Conclusions