National Library of Canada

Acquisitions and Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada

Direction des acquisitions et des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Canada

APPROACHES FOR HANDLING SOFTWARE VERSION MISMATCH IN DATA COMMUNICATION NETWORKS

By
Hartek Singh Minhas

A thesis submitted to
the School of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

Master of Computer Science

Department of Computer Science - University of Ottawa
(Ottawa-Carleton Institute for Computer Science)

Ottawa, Ontario
CANADA

ISBN 0-612-00615-8

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

ACKNOWLEDGEMENT

I would like to express my deepest appreciation and sincere thanks to my thesis supervisor Dr. R. L. Probert of the Department of Computer Science for his valuable advice, suggestions and encouragement.

ABSTRACT

Most existing networks, whether homogeneous or heterogeneous networks, coexist and interoperate relatively well. Various software implementations of OSI standards and CCITT recommendations are available in such networks and interoperability between them is taken for granted even though successful interoperability is not always completely achievable. In this thesis, a distinct implementation of a software, protocol or interface standard is called a "version". There are versions of application software (X.400'84, X.400'88), versions of communication software (X.25'80, X.25'84, X.25'88), versions of operating systems (UNIX 3.0, UNIX 3.5, VMS 3.0, VMS 4.0, DOS 4.0) and so on.

This thesis studies restrictive interoperability between distinct versions of implementations based on standards or recommendations. Successful interoperability between two communicating versions on different networks depends upon:

- how accurately the versions were derived from the standard, and
- how similar were the selection of supported options in the distinct versions, and
- how familiar were the respective network managers about the functional differences between the two versions before provisioning and the start of commercial operation.

Each network has a complex architecture to support numerous applications installed on a broad range of different hardware platforms. Applications are designed to be accessed by local users or public network subsribers. Attempts made by users/subscribers to access a service from a remote application in order to carry out an end-to-end

function over one or more than one distinct network may fail because of:

- functional limitations of the protocol stacks beneath the target application, or
- functional limitations in the services or options offered by an intermediary network.

In this thesis, we relate possible causes and give possible methods for prevention and recovery within two primary development and operational mechanisms:

1. The standards development process.
2. Network management methods.

It is shown how version mismatch problem systematically becomes a by-product of standard development process. Version mismatch, unknowingly and unintentionally becomes an integral and inseparable part of many proposed software standards. One of the crucial tasks of the standard development process is to produce improved versions of standards. On the other hand, one of tasks imposed on network managers is to manage multiple versions based on a single standard.

One of the primary objectives of network management systems(NMS) is to detect and resolve problems in the networks. The most crucial information to resolve a problem is the exact location of the module where the problem has occurred. This area of resolving problems, in the literature, is divided into two functions: fault isolation and resolution. Most of the known NMS's are lacking the capability of handling version mismatch problems. In this thesis, detailed analysis to resolve version mismatch problem has been provided. We have included the description of the problem as well as a more detailed discussion of the charectristics of a version mismatch

problem. This analysis will help distinguish version mismatch problem from the problems which may not be considered as version mismatch problem. Briefly, version mismatch problems can occur where two versions of software are communicating with each other. Also we have described systematic ways to resolve version mismatch problem.

The Main contributions of the thesis

1. A precise definition of the Version Mismatch problem encountered in the lower three layers of packet switching network.

2. A partial characterization of root causes of the Version Mismatch problem.

3. Improvements are proposed to the standard development process to take into account the Version Mismatch (VM) problem. These improvements involve the definition of the Core of the Protocol based on identifying common operational priorities and tagging functions as mandatory, optional, prohibited or not applicable.

4. Suggestions for improvements to current network management procedures to take into account VM management and control, namely

- version incompatibility management procedures and
- an integrated network management architecture.

Examples are presented to illustrate the potential value of these improvements as well as an indication of which improvements have been at least partially achieved in the evolving OSI standards and non-OSI standards.

TABLE OF CONTENTS

Chapter One

Chapter 2

The Version Mismatch Problem in Current Standards Evolution and Network Management Practices

Chapter 3

A New Process for Prioritizing Functions during Standard Development Cycle

Chapter 4

Network Management (NM) Architecture and Version Mismatch (VM) Problem

Chapter 5

Conclusions and Assessments of Proposed Process

Figures

CHAPTER 1

1.1 Introduction and Background

In this decade, there has been a tremendous increase in the development and deployment of computer networks. Computers are using a vast number of communication protocols implementations based on standards. There are many other computers not using specific standards but making use of non-standard specifications. A large variety of applications are being accessed via these networks. Applications have been developed to support many clerical and managerial functions such as word processing, spreadsheet and electronic mail. Applications are used not only within the network, but increasingly also provide electronic connection to both suppliers and customers [CARU90]. This explosion in the demand for networking the information resources for the use of public information companies as well as the demand by the corporations has made inter-communication across networks of great importance.

A computer network is composed of a number of local as well as remote data processing resources available to most network users. Communication protocols e.g. X.25, X.75, network architectures e.g. OSI, SNA, TCP/IP, and application services e.g. FAX delivery through X.400 are bound to increase in number as well as in complexity with the expansion of networks. All this can lead to serious problems of network management and administration of networks.

In modern computer communication networks, a computer can communicate with two other computers through two different independent protocol stacks. Similar protocol stacks may use various versions of software implementations. Two identical implementations can support many different options. For example: a X.400 implementation can

support many different customer selectable options which can vary from customer to customer. Variations like these add another important dimension to network management, namely the management of software versions so as to understand, to predict and to control the impact of version mismatch on the operation of the network.

Communication software is developed by various software development companies based on national or international standards, and should be developed according to a formal process. In every protocol specification, certain functions are mandatory and certain other functions are optional. Optional functions are developed at the discretion of customer marketing representatives. Certain functions are not implemented because of their lack of importance to the overall functionality [MEPE86] or lack of perceived market demand. As a result of such decisions, mutually incompatible software can be installed on nodes in the same network. Communication attempts may be made through such nodes resulting in a potential nightmare for network managers We give rigorous definition and example of such mismatches in Chapter 2.

Complete and accurate communication between nodes demands complete communication between their protocol entities [ISO1]. If the functionality of the entities is not compatible (definition in Chapter 3), the communication may break down. This may cause the overall functionality of the network as well as certain crucial services to degrade dramatically. This problem of unexpected functional mismatches during the operation of the network is one of the issues dealt with regularly during network management.

The intervention of a network manager can be caused by a system alarm or as a result of a call from the end user. They

tend to intervene to alleviate the problem by partially disabling the culprit function or by completely disabling the function or service so that both affected parties can reestablish communication with each other without sacrificing the performance of the system[HELD89]. They attempt to correct the problem without making significant changes in the user environment.

Communication service providers are designed as layers of software modules each performing a specific set of functions. Grouping of the functions for a layer is accomplished by establishing the common purpose of each layer.

Layered protocol structure is designed to establish and maintain the data transmission parameters such as window size and throughput class so as to permit accurate and effective communication. For each class of communications such as dial-up, dedicated, out-dial, connectionless or connection oriented, a choice of services and protocols are allocated by the network managers at the time when the contract is signed between the subscriber and the network utility. Services and protocols conform to CCITT Recommendations and ISO Standards.

It is important to remember that standards are continuously evolving to provide improved functions and to address current problems. When a new version of the standard specification containing new services becomes available, the networks are required to be upgraded to support those new services. This type of unplanned upgrades in software can trigger ripple of changes in the functionality of existing services. To avoid regression problems, it would be useful to provide network managers with detailed documentation of such upgrades.

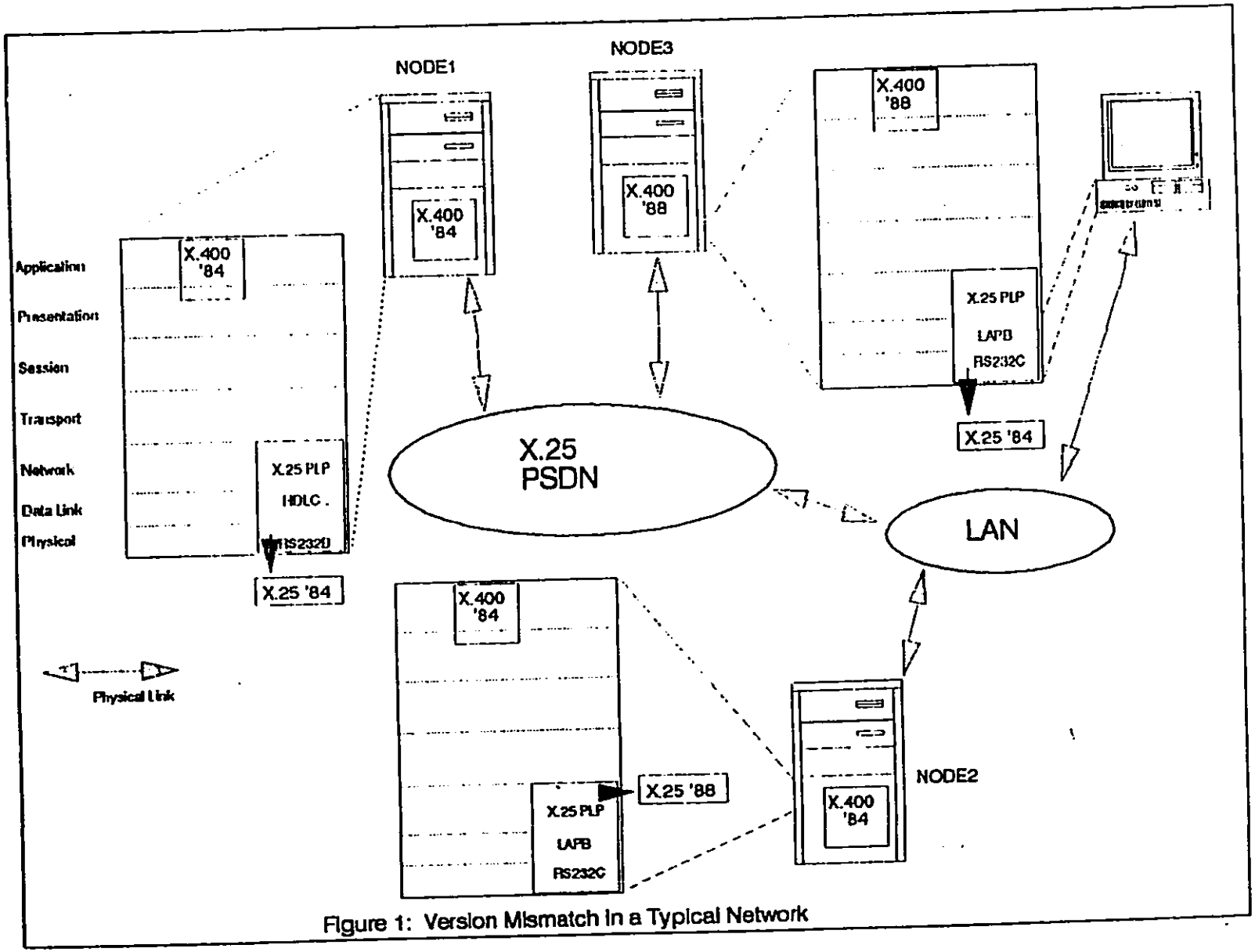


Figure 1: Version Mismatch in a Typical Network

1.2 Motivation

Consider the configuration in **Figure 1**. This network consists of three nodes - NODE1, NODE2 and NODE3 - designed to support an electronic messaging system based on CCITTs X.400 Recommendation (Message Handling System).

NODE1 is installed with two protocol entities, an implementation based on CCITT Recommendation X.25'84 to provide networking infrastructure and an application to provide services based on CCITT Recommendation X.400'84.

NODE2 is installed with an implementation based on CCITT Recommendation X.25'88 to provide networking infrastructure. It is supporting the application software based on CCITT Recommendation X.400'84.

NODE3 is installed with an implementation based on CCITT Recommendation X.25'84 to provide networking infrastructure and supporting the application software based on CCITT Recommendation X.400'88.

Service subscribers may try to access X.400 services through Personal Computers. They can send messages to other remote users connected to nodes other than their own. When these nodes try to communicate with each other, the communication can become extremely restrictive. The reason is that X.400'84 provides fewer services than X.400'88. Also many services in X.400'88 are improvements of the X.400'84 specification. The users can therefore face problems in sending messages composed by using X.400'88 services to the users on X.400'84 nodes. To further complicate the problem, these applications are supported by two different versions of X.25(1984 and 1988) which means that any protocol function required to support X.400 based

electronic services must be present in both X.25'84 and X.25'88 implementations. Such scenarios are very likely in modern networks where users are encouraged to establish interconnections through non-identical versions of communication protocols.

It is possible to have different versions of software installed at other layers such as Session Layer and Presentation Layer as shown in **Figure 2**. This adds complexity to a already complex problem.

Before proceeding further, we will try to establish differences between problems encountered during network operation which are solely produced due to multiple versions of software.

Many problems are encountered during the management of networks installed with different versions of implementations. Such problems as discussed in detail in Chapter 4. Many of them can be fixed if detailed knowledge between the versions is available to network managers.

There are many issues related to troubleshooting networks which can be resolved or avoided altogether if the networks installed with large population of mixed versions of communication protocols can be identified. Similarly issues concerning potential backward compatibility of newer versions of software can be predicted.

Issues related to prioritizing of the implementation of functions necessary for conformance to international or national protocol standards can be controlled and understood. It is the tendency of the software developers to first design and implement mandatory functions and important services and to leave the optional functions and less desired services to the end of the

development cycle or sometimes leave them completely unimplemented. This can lead to creations of versions which are not interoperable.

In this thesis, we will give examples of this type of version mismatch using X.400 and X.25. X.25 protocol has been selected because of the following characteristics:

1. It can be used in a wide variety of networks and configurations, including point-to-point (DTE - DTE) communications.

2. It is a mature public standard.

3. Its use is rapidly expanding and will likely migrate into future ISDN evolution.

4. It has evolved into a reliable and widely available system.

To cover version mismatches in applications, CCITT's X.400 series of recommendations are selected. This is the most growing and extensively used application for which comprehensive set of CCITT recommendations are available.

1.3 Contribution

In this thesis, an effort is made to provide initial partial characterizing attributes leading to the Version Mismatch problem. A new approach is proposed to expand the list of existing rules which tag functions as mandatory or optional.

The problems are to be investigated from two points of views involved in terms of standard development and in terms of network management.

In Chapter 4, attempt is made to document the complexity of the problems faced by network managers who support

networks populated with multiple versions of communication protocols or applications.

In Chapter 3, a new approach is put forward which will help standard development groups to define functions such that the importance of the functions is not tied to operational aspects alone but also other factors which will later help to alleviate the problems and issues created by version mismatch.

We propose a new approach to Network Management (NM) where impacts of version mismatch can be predicted based on the availability and non-availability of functions. This approach is used in the non-OSI world in the latest version of SNMP, but is not discussed here. See [STAL93] for more details.

1.4 Organization of the Thesis

The remainder of the thesis is structured as follows. Chapter 2 presents a literature survey of work related to the current standards development process and the current network management process. As well we discuss the relationship of these processes to version mismatch. Relevant concepts in the area of data communication in relationship to protocols, layering, functions, and standards are also discussed

In Chapter 3, the version mismatch (VM) issue is dealt with closely in relationship to the standards evolution process. A new approach is recommended to be incorporated into the existing standards evolution process.

In Chapter 4, we focus on network management practices as impacted by version mismatch; that is, management of multi-version networks when information is available to operators before the software installation as well as during network operation.

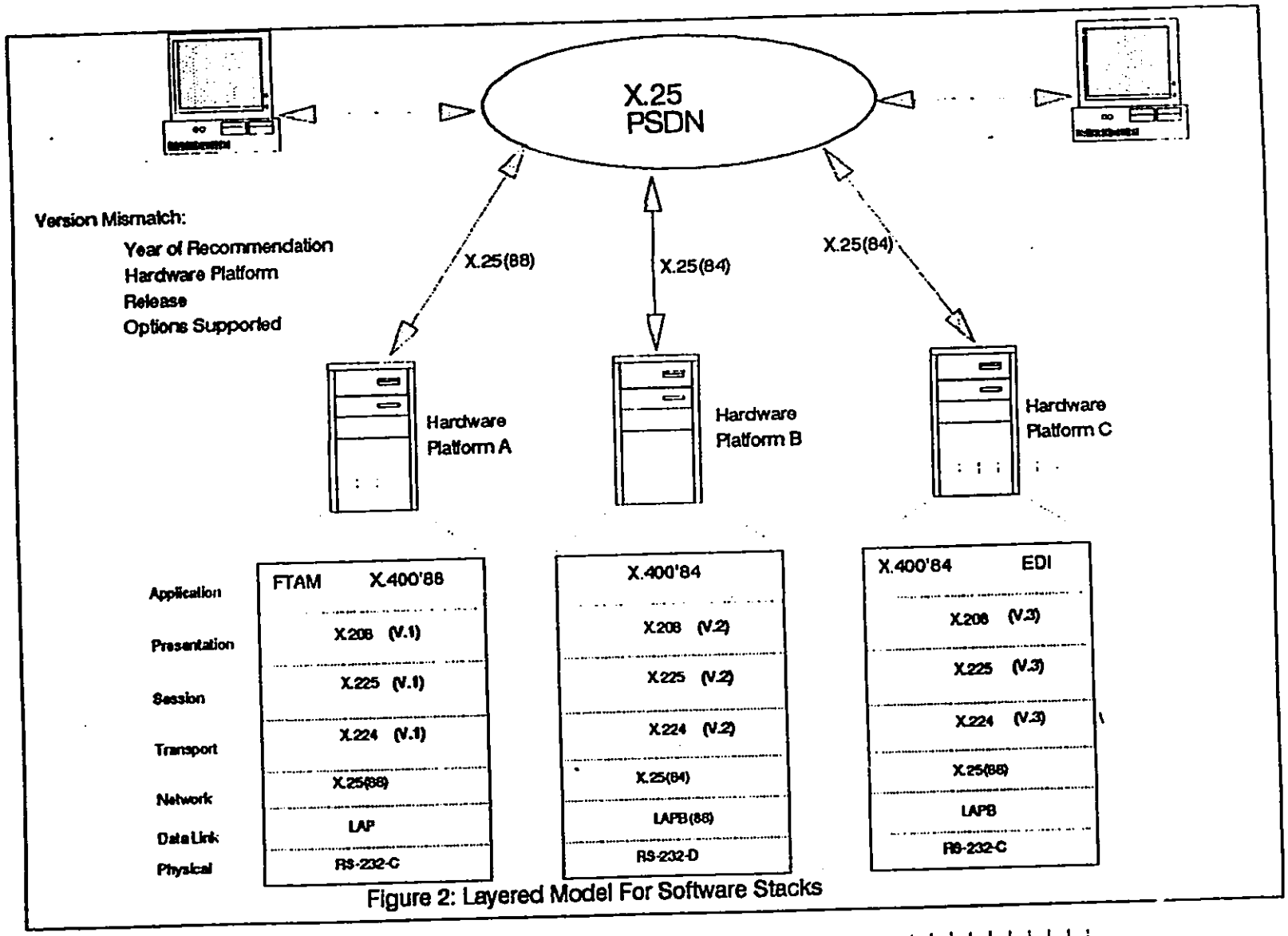


Figure 2: Layered Model For Software Stacks

Chapter 5 provides some conclusions, advantages and disadvantages of these newly proposed approaches as compared to the current ones. Finally, suggestions are made for future research.

Chapter 2

The Version Mismatch Problem in Current Standard Evolution and Network Management Practices

2. Introduction

Access to and exchange of the most up-to-date information at all times is vital for information managers. Exchange of information must be error-free, reliable and consistent. A physical channel between computers is necessary but not sufficient for achieving reliable, secure and accurate information transfer.

Such business requirements as these force separate organizations into joint efforts to design and develop national and international standards for dealing with communications software. The standards development process will be discussed in detail in the following sections to demonstrate ways in which this process actually contributes to the problems of static and dynamic version mismatch.

Any Open System software designed for transmitting messages and information must adhere to a common set of rules which is defined in a document or set of documents denoted **communication standards** or sometimes, **protocol standards**.

Protocol implementations are designed and developed by various corporations or organizations based on protocol standards. Some implementations are denoted "proprietary" implementations

because their distribution and use are strictly controlled by private companies.

The international community develops standards under the umbrella of the International Standards Organization(ISO) for the standardization. In late 70's, the U.S. Department of Defence put together an organization which was solely responsible for communication protocol software standards and networking architectures. Since then, DARPA has produced well accepted standards - Internet Suite of Protocols.

2.1 Functions and Services: Key Ingredients of Specification Standards

The exchange of information between two nodes requires the involvement of three basic categories of communication functions[MELE86]. These functions can be grouped together depending upon the communication or interpretation state at which the services of these functions are used. Functions could be responsible for providing the following three capabilities:

- Data transfer from one node to another.
- Identification of the communication relationship (originator, responder).
- Interpretation of the information.

Data transfer phase involves protocol handshaking between the two parties and is required at all levels of communication. For example, before one node switches itself in a waiting state to receive data, it must notify about its current state to the other node. After receiving and correctly interpreting the notification, the other node starts transferring data. This is the beginning of the execution

of the functions responsible for the third phase.

During second capability data is received, checked for errors. Data is corrected if it is correctable and interpreted to establish the purpose of data's arrival or simply passed on to a user process.

Any communication between two systems involves three phases such as these:

1. Connection establishment
2. Data Transfer
3. Data Verification and Acknowledgement

The functions and services together can be described in terms of objects, each associated with a specific task. A node can also be considered as an object. We will discuss the management of node and its manageable attributes in detail in later sections on network management.

2.1.1 Node Objects

The management of a network includes the management of a number of software-related objects as well as hardware-related objects. Our discussion here is limited to management of software-related objects.

OBJECTS are:

- Nodes
- Protocol Stacks
- Protocol Entities
- Functions and associated services

The software version mismatch problem can affect at least first three out of the four above mentioned network objects. Network managers try to control these objects by acquiring some

knowledge and understanding of the software functions included in the design of these objects. From acquired knowledge, they predict the future behaviors of the objects and their interactions. But at the same time, it is quite difficult for the network manager to predict the effect of functions not included in the software design. This situation is discussed in considerable detail in Chapter 4.

Nodes

An important function of any node is to manage the attached resources. The nodes are capable of providing services or capabilities such as: node access; communication link establishment; keeping record of the routing information; reliable data transfer; keeping statistics; and keeping accounting records. Providing such functions and services is a requirement for most of nodes.

Protocol Stacks

For the sake of the management of the network, it is extremely important for both communicating nodes to have similar protocol stacks when trying to communicate with each other. If the communication software packages are not designed and stacked by using the same coding conventions, many problems will occur during the management of protocol stacks.

For each type of communication requirement, selection of services, protocols and protocol stacks should be accurate to achieve secure and reliable data transmission . The detailed user and network requirements of the services available, based on implementation of the functions in protocols conforming to CCITT Recommendations and/or ISO Standards, should also be outlined. Provisions are usually made for negotiation between communicating end-systems and intervening communication networks.

Service Layer

The lower layer of the ISO's OSI [ISO1] seven layer model - physical, data link, network and transport - are responsible for reliable transfer of isolated units of data from one system to another over the intervening subnetwork. The selection of the services and the protocols for the upper layers is based on the requirements of the different applications.

Using the layered structure, each layer is controlled by formal functional specification, both from peer-to-peer "protocol" perspective and with a defined "service" interface between layers. Service is provided to the layer above and the peer-to-peer protocol is used in a communications with the same layer of the open system at the far end of the connection. Each successive layer is built on the next lower layer, and layers are linked to provide ease of implementation for future modifications.

In a particular implementation, the services of a lower layer protocol might be directly accessible only by protocols in the next higher or lower layer, as it is clearly defined in OSI Reference Model. On the other hand, through the usual interprocess communication facilities and subroutine calls of the product, layers are designed and implemented so that they can be accessible by other protocols residing at a higher layer.

Protocol Implementations

In data communication networks, the node resident software establishes a communication link with other corresponding nodes before nodes try to exchange information with each other and then in turn deliver to the users. If the functionality of communication software on the first node is limited as compared to the functionality of the second node, the communication session between the two would not be as stable and reliable as desired. Depending upon the place of the function in the protocol stack, this kind of software functionality limitations could cause a number of network management and network operation related issues during the communication session.

To understand this issue of location of the software limitation during interoperability between the nodes in the network or across the networks, one has to define and separate functionality into groups based on the nature of the functions and their location in the implementation architecture. The implementation architecture could be either layered architecture or non-layered architecture. Implementation groups are moving towards layered architecture from non-layered architecture because of many advantages over others such as: diagnostic capability, interoperability, performance, and being less trouble some during upgrading and high modularity.

Network Architecture

In this thesis the term network architecture will be used for the underlying structure of communication control, the provision of network services in networks. This term has been used extensively in computer communication network related literature [BRAD88, BELL88, CARU90, FERN89] with slightly differing meaning. Often a set of communication protocols - protocol stacks - and the form in which they are coupled in the network can be regarded as network architecture.

Architecture in itself covers many structures of network, namely:

- Topological Structure
- Network Operations Structure
- Layered Structure

The definition of network architecture is an agreement with many existing network architectures. Many types of architecture like ARPANET are available while the architecture for packet switching networks are explained in various ISO, CCITT and IFIP documents. Many vendor provided networks are available like SNA of IBM and DNA of DEC.

The seven-layer architecture of the OSI-Reference Model is the most significant and widely used architecture which identifies and groups communication functions as well as facilitates to achieve the coupling of applications with communication functions. ISO's OSI layered stack was a product of international effort and in the process has inherited many international problems. Despite this drawback, the OSI Reference Model gained universal acceptance as an appropriate computer systems architecture.

Another organization responsible for the development of protocol specifications is CCITT which is an abbreviation for International Telephone and Telegraph Consultative Committee. The specifications produced by this organization are known as **Recommendations**. The recommendations produced by CCITT are more extensively used as compared to ISO. In this thesis, the word "standard" and "recommendations" are used interchangeably and refer to same concept.

ISO's OSI Reference Model is a several-layered model. Each layer is designed to perform certain functions and provide certain services to the layers above it and below it including the successful transfer of information.

Each layer of the OSI Reference Model is made of entities which communicate with their peers according to a predefined protocol. Both corresponding entities on each side have to use similar coding conventions, data structures, timers, etc. to communicate. This demand makes these complex layers very difficult to specify and much more difficult to implement.

2.2 Interconnection Between Nodes and Issues of Version Mismatch

Before the nodes establish an interconnection, corresponding functions at different levels in each node's software need to be identified and interactions between them need to be defined [MELE86]. If the software versions running on the nodes are the same, then the effects of each function in the protocol are similar in both communicating nodes. The services associated with a function can be pre-specified according to the task which may be influenced by user requirements.

Establishing interconnection between nodes not running similar version of software is complicated in consideration of the fact that complete effect of a function in one node may not exist in the node to be interconnected.

The approaches to interconnect two nodes with different software versions must take functional differences into account. Software implementations and, hence, software versions can differ in many aspects and often at different levels. A simple example of a minor version mismatch is given below.

EXAMPLE 1

At the packet level, one implementation version could support a packet size option of 1024 bytes and another could be limited to a packet size of 256 bytes. The limitation in the second version could be because of node resident hardware restrictions or just because of software design decisions. Such nodes can establish communication links with each other by nagotiating to avoid transfer of packets larger than 256 bytes in size. The successful transfer will be achieved only if packet size is restricted to 128 or 256 and not larger.

Hence, the capability of the first node to send or receive 1024 bytes of packets is superfluous. This is a simple example of version mismatch The magnitude and impact of the VM problems can vary from function to function less and service to service. The type of this version mismatch is denoted as **version mismatch with respect to static option selection.**

To establish interconnection between nodes not equipped with same versions of software can be avoided by establishing formal guidelines acceptable to both communication parties. Those guidelines also needs enforcement by network operators.

But first, let us define and establish why and when communication software are considered - statically or dynamically - incompatible to each other and whether the incompatibility is only limited to certain function or part of the functions.

2.2.1 Static Software Version Mismatch

Once installed on node, communication software has to be compatible to the internal hardware mechanisms. Parameters such as memory limits, CPU processing speed, clock cycles, and granularity of the clock need proper configuration. If any similar options do not comply with the hardware requirements, the result would be the provision of faulty software for the customers. The worst possibility is that the node would crash during the operation. Such environment can also create many unexpected mode including faulty call set-up modes or faulty data transfer mode.

EXAMPLE 2

EIA RS-232-C interface software implemented according to a standard is capable of supporting many data transmission speeds such as 300, 1200, 2400, 4800, and 9600 bits per second. In the real world, for each dedicated circuit, utility assignees the speed to the circuit at the time when the contract is signed between utility and the service subscriber. This parameter remains in effect throughout the duration of the contract or if the user pays to upgrade or downgrade circuit speed. Only network operators can make such upgrades. Usually it requires replacement or adjustment of hardware at modems and DTE's. Sometimes, in older equipment, options are stored in Read Only Memory (ROM) chips. Administer and control of DTE's is performed through dip switches or straps.

If a pad does not support a particular speed as requested by the subscribers

after he upgrades his DTE, he would face a version mismatch between his environment(DTE) and PAD to which he is physically connected(DCE). **This type of mismatch is denoted as version mismatch with respect to static options.**

EXAMPLE 3

Let us consider Figure 4, a real scenario. HUNT group is an X.25 user facility which can be bought by the user at the subscription time.. With this facility, a remote user is able to access the first available DTE out of a large predefined group of DTE's. The DTE's can be selected from more than one node. If a new X.25 version of software is installed and the version does not support the hunt group capability as other nodes do. Then consistency of the service would break down. **This version mismatch is caused by a new software release.**

2.2.2 Dynamic software Version Mismatch

The software ready to be installed at a node should have its dynamically selectable options compatible to the standard and also to the hardware constraints. The software running at a particular node can not provide DCE access baud rate to the users if their hardware can not support that speed, even if software implementation is capable of supporting that speed and even if that speed is recommended in international protocol specification. The process of upgrading any communication software is constrained by the hardware functionality.

Dynamic software compatibility is the issue which is one of the major headaches for network managers. The network subscribers have options to select parameters to satisfy their dynamic needs. Attempting to select an option which is not supported by communication software or hardware can become tedious exercise both for the user and for the network.

EXAMPLE 4

X.3 and X.25 communication protocols provide a mechanism to change and select parameters to fit user environment on the fly. In other words, users can change these parameters before establishing a call or even in the middle of data transfer. An example of such a dynamic parameter is the capability to enable or disable flow control(X/ON or X/OFF) sent either from the network or from the terminal. This parameter controls the flow of data traffic. In case the buffer are full, X/OFF is sent. These changeable or selectable dynamic parameters remain in effect as long as call is not dropped. **This is mismatch with respect to dynamic options selection.**

EXAMPLE 5

The most extensively used user parameters of X.25 that are dynamic in nature are:

1. **packet size** and
2. **window size.**

The **packet size** indicates the maximum packet size that the receiver can receive and sender can send. 512 bytes is commonly used for packet size which allows a substantial quantity of user data per packet. This is especially useful for applications such as file transfer or data collection for statistical purposes. The **window size** indicates the number of outstanding unacknowledged packets allowed for a communication link. Increasing window size will decrease the number of control packets on the network and provide the sender with a shorter waiting time before transmission. There may be two releases of which one supports window size of 8 and other window size of 128. **This mismatch can be denoted as version mismatch with respect to static options selected.**

2.3 Version Mismatch Due to Evolution of Standards

The complexity of functions is compounded by the fact that large networks grow, evolve and are modified throughout their lifetime. This evolution is due in part to the discovery of design errors, in part to technical enhancements, and in part to the evolutionary changes in requirements imposed by the users who are using services offered by the networks.

2.3.1 Vendor's Viewpoint

Information systems user or vendor must always deal with two worlds: that of standard products and that of proprietary products. This is an oversimplification, since even the standards arena may have multiple components; formal international standards (represented primarily, but not exclusively, by ISO & CCITT and de facto standard created primarily by market acceptance of initially proprietary products. Vendors of information processing products are affected in several ways. On one hand their customers want flexibility, in terms of supplier selection. On the other hand, major vendors must supplement the functional area covered by standards with proprietary products in order to provide complete solutions[WARD89]

Protocols as well as their underlying individual functions become part of national or international standards after a private network or a vendor has successfully offered a service based on a specific function which was accidentally or intentionally not included in the final formal body of the standards. Network vendors have proposed and achieved success in including a function into the standards. The prime example of this kind is when IBM's SDLC was adopted by CCITT and then CCITT changed its name to HDLC (High Level Data Link Layer) keeping almost all its functionality intact which at the end became a large and integral part of the crucial second layer of X.25 protocol specification(excluding LAPB).

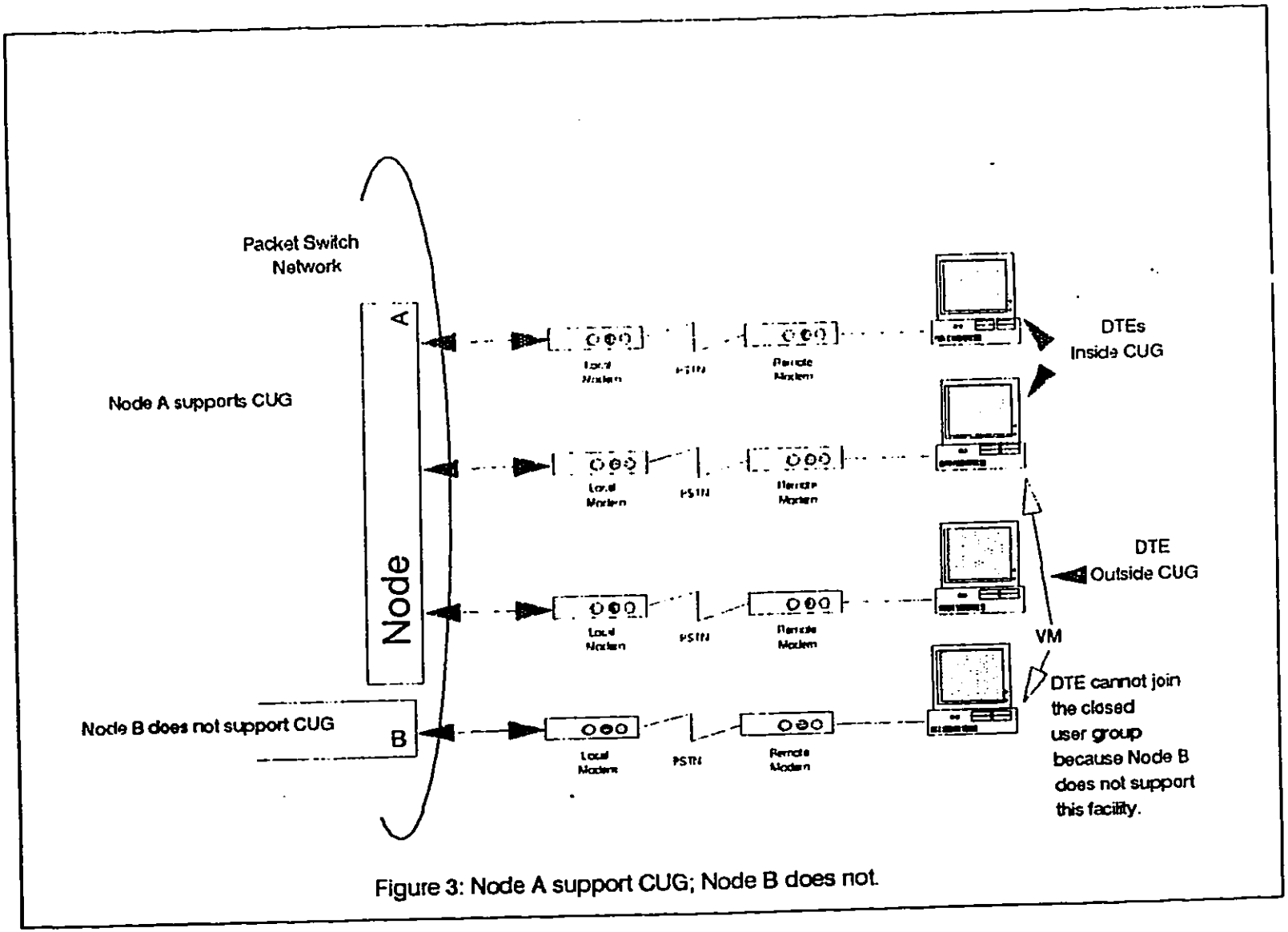


Figure 3: Node A support CUG; Node B does not.

2.3.2 Network User's Viewpoint

Another reason for a function's merging into a standard is the result of the demand by network and application users. They continuously upgrade the networks to meet the demands. This reason is also a major thrust behind the evolution of standards.

Examples of user facilities which have been incorporated into evolving standards to support network user's requirements are CUG, HUNT Group and PDS. Before discussing the effect this has in relation to version mismatch, we briefly describe each of these optional facilities.

EXAMPLE 6

Closed User Group (CUG) is an optional facility offered to subscribers in X.25.

In this feature, a group of DTE's can be defined in X.25 public packet switching network environment such that group members can communicate with each other but any other DTE outside the group will be barred from communicating with the members of group (Figure 3).

EXAMPLE 7

HUNT group is an option requested at subscription time by X.25 service subscribers. A group of DTE's can be assigned as a HUNT Group in order of priority. Once a remote DTE tries to access that group, it will be connected to the first available DTE from that HUNT group(**Figure 4**). In other words, the remote DCE will continue to hunt until it find next available DTE from the group.

Closed User Group (CUG) functionality and Hunt Group Selection functionality became part of the X.25 specification because public utilities and their network user population have requested to have these particular functions included into the CCITT X.25 Recommendation.

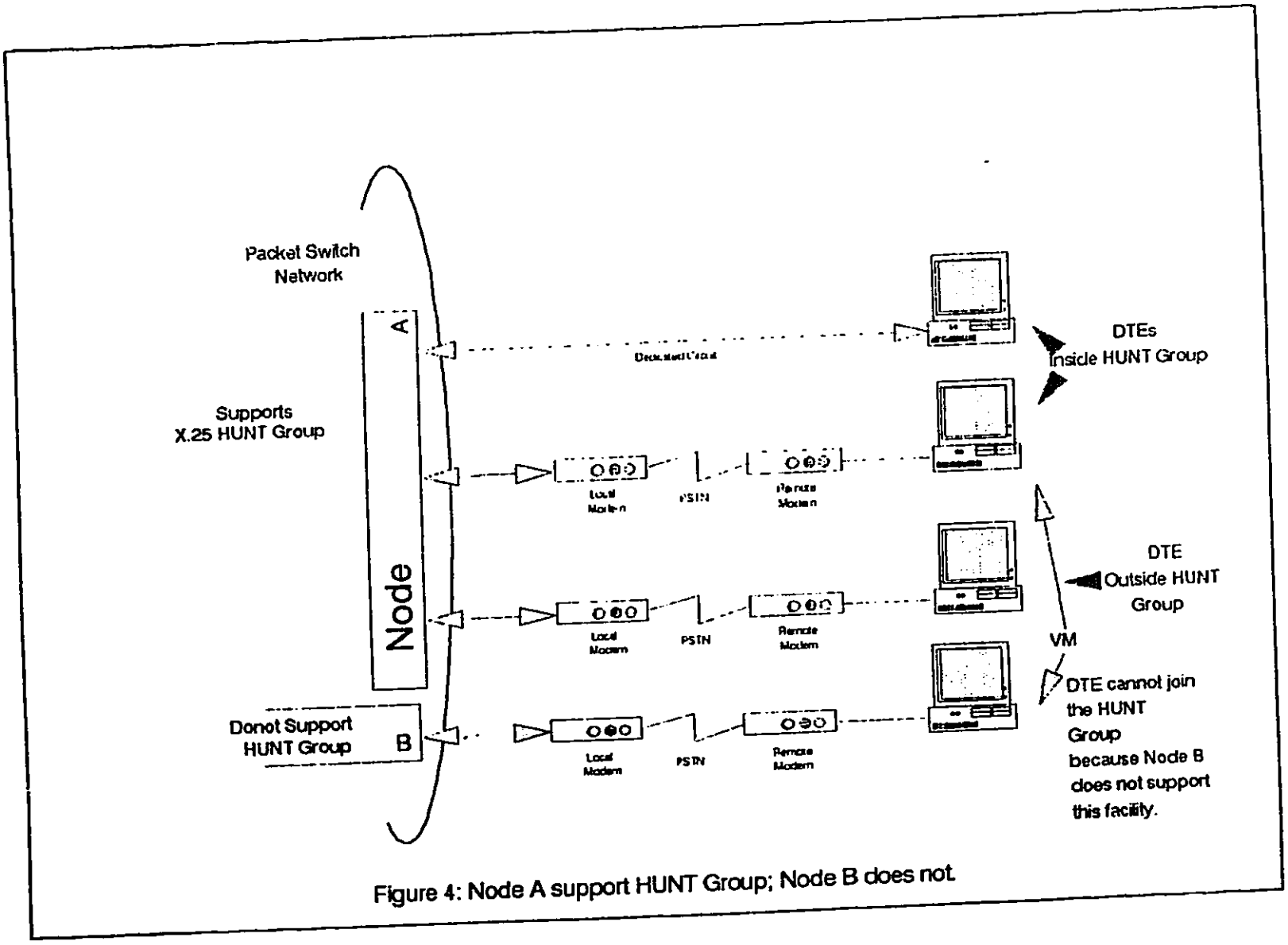


Figure 4: Node A support HUNT Group; Node B does not

To understand the version mismatch problem, let us look at following two examples. In **Figure 3**, **Node A** is capable to support CUG facility to its service subscribers. Two DTE's ("DTE's inside CUG" as shown in figure) have subscribed for this service and the third DTE ("DTE outside CUG", as shown in figure) chose to stay out of this group. But, **Node B** is not capable to offer this services. Any DTE from this node, even if it wants to join the above CUG cannot do that because of the lack of proper version of software on NODE B. This is a situation of version mismatch between **Node A** and **Node B** Software. **Figure 4** shows breakdown in HUNT group service between two nodes where one node (NODE A) support hunt group and second one (NODE B) does not. This causes a version mismatch problem as illustrated in the **Figure 3 and 4.**

EXAMPLE 8

The Physical Delivery Service (PDS) allows electronic mail based on X.400 specification to be moved over packet switching networks and printed on remote pre-defined printers. Physical Delivery Service was not defined in X.400-1984 specification. PDS service has been successfully implemented and tested by many organizations in the late eighties including Canada Post Corporation and this service will be offered publicly in the early nineties and has been accepted and adopted by CCITT PDS service in the X.400(1988) standard. In this example, the network users have succeeded in convincing standard development bodies to include a service oriented feature into the standard.

This services will only be available to customers who have accounts X.400(88) system. The customers who are on systems other than CPC's system will not be capable to use PDS services if their local host systems do not support such service. **This type of mismatch can be denoted as version mismatch with respect to year of recommendation of the specification of the top layer of the protocol stack.**

2.3.3 Arbitrary Technological Constraints

The primary function of communication protocols is to transport data generated by an application usually sitting on the top layer of the protocol stack [ISO1]. When lower layer protocols evolve, the application supported by them must evolve with the protocol. The magnitude of the change in the application could be minor when at times, it requires changing one or two literal values, or it could be major when it requires changing the data structures defining the headers of the protocol envelope.

EXAMPLE 9

The X.400 Message Handling System is a specification developed by the CCITT [KaBa88, SiDo88]. This recommendation defines the Message Handling Services which provides the architecture for exchanging messages on a store-and-forward basis. It also defines application environment for the delivery of many different types of messages such as text, facsimiles, graphics, faxes and sometimes different types of character sets. This recommendation defines X.25 as lower layer architecture. This means that the networks not currently supporting X.25 protocol suites but planning to offer X.400 services to its users can be left in the cold.

Most educational institutions are connected through TCP/IP backbone network architecture. It would be impossible for them to support X.400 without making crucial changes in terms of human resources and equipment to support their networks. The best option for them would be to install an X.400 application customized for TCP/IP as an architecture for lower two layers. For a moment consider a situation when X.400 application is installed on TCP/IP lower layer stack. It is possible that in future, X.400 series of standards will evolve into a broader recommendation which would end up including TCP/IP as one of the lower layer stacks to accommodate TCP/IP

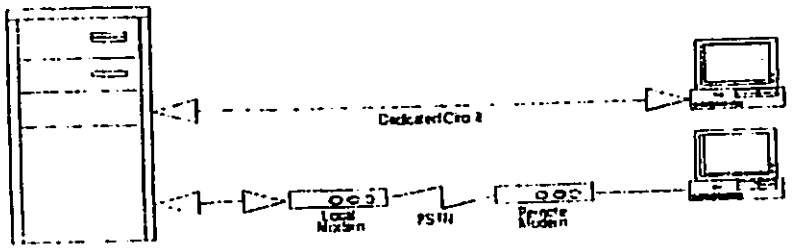
networking world. Then the user could access the application by following the TCP/IP route as well as the X.25 route. This recommended solution will provide more coverage as far as usability and coverage of this standard is concerned.

Such a mismatch can be denoted as version mismatch due two distinct vendor architectures - one where X.400 is stacked on TCP/IP and other where X.400 is stacked on X.25. The complexity mismatch will further be complicated if the different year of recommendation are also included in discussions.

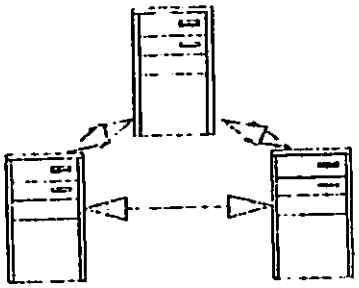
2.4 Current Network Management Practices

The purpose of network management is to provide a single set of tools for managing all network resources. A network can either be distributed or centralized and can constitute small subnetworks. Such subnetworks can be derived from different vendor-proposed architectures while the networking software is also supplied by them. To achieve such a network management task one should design a network management system acceptable and suitable to subnetworks. Management function and managed objects should be consistent across the network. The management information transfer protocols should be consistent, and last but not least, all the nodes and peripherals should be using consistent communication software as well as application software. Anybody involved in real life operation networks can easily suggest that it is too much to ask from currently available network management oriented-architectures and software.

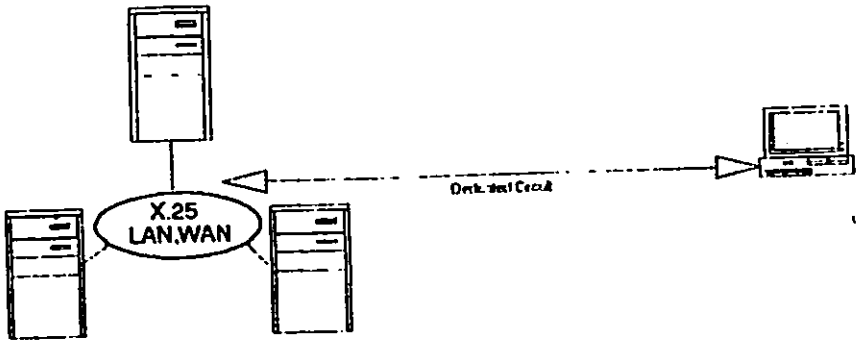
The management of consistent networks is much easier as compared to networks which are populated with different software and hardware provided by all kind of vendors. Almost all the work in the network management area is associated with networks which are built based on a specific architecture [BELL87, CARU90, EmMa90, FARU80, JoMU90, KELL89].



VM is simply not an issue (a)



VM is an Issue (b)



VM is a Complex Issue (c)

Figure 5: Types of Networks

2.4.1 Networks inside Networks: Types of Networks

The idea of managing complex computer networks is not new: it is almost as old as the idea of networking the distributed or physically scattered computing resources like PC's, workstations, micro-computers, mini-computers and mainframes, printers and plotters. International standard development bodies have proposed a few number of complex network management architectures and protocols[FERN89, EmMa90, KELL87].

Many local area networks are equipped with their own network management tools capable of performing network management through only a particular set of architectures and software/hardware platforms.

Digital's DECnet is providing an ethernet based office automation architecture. It has a set of products designed to manage its own environment. This particular product is not built to manage any other local area network's extensive customization. In essence, there is not a single product on the market which can manage all possible LAN configuration topologies and architectures.

Before we start to discuss in detail network management and issues related to the version mismatch problem, it is important to define some common types of networks available to network managers.

2.4.1.1 Terminal Access Network

A terminal access network consists of a collection of terminals and printers sharing a micro, mini, or main frame computer (**Figure 5.a**).

In this network the terminals are directly connected to the host. There are many network management related tools and products to manage a bank of terminals connected to a host.

The examples of such networks are many.

1. A large pool of terminals accessing a University computer.

2. A large number of Point of Sale terminals (a typical network of a departmental store) hooked to a central computer.

Such networks are immune to problems caused by software version mismatch. The main reason is that there is only one node -a single host in this case- and information generated due to network management remains local to this node. Any upgrade to the existing versions of an applications or terminal server software will cause changes to only one node and questions of interconnectivity between nodes does not arise.

2.4.1.2 Computer to Computer Network

A computer to computer network is a collection of computers exchanging information with each other while using a standard communication protocol (**Figure 5.b**). One such network is weather prediction computer collecting information from many computers spread over vast geographical area. Computer-to-computer network architectures are extremely vulnerable to a version mismatch problem. Each node must have a copy of the same communication software to achieve reliable and error-free communication.

2.4.1.3 Packet Switching Network

Many large multinational organizations need to make different computer systems communicate with each other while the computers are thousand of miles apart from each other. One way is to connect them by dedicated leased lines. This method is expensive but used extensively

There are two major drawbacks in such set up.

1. Leased lines are especially expensive in Canada.
2. It only provides closed communication.

That is, to switch public packet switching networks.

There are many public and private packet switching networks currently in operation (**Figure 5.c**). Two such networks are DATAPAC, a Canadian packet switching network and INFOSWITCH, CNCP's packet switching network.

PSN networks are very complex in terms of network management. The version mismatch issue is of paramount importance in such networks as we discussed in previous chapters.

2.4.1.4 Local Area Network

A local Area Network is distinguished from other networks by the area it serves. More or less, local area networks were devised to handle the needs of a local community. In its most general sense, it is probably not larger than 10 KM across. Although, most networks vendors limit the local area network performance to a few hundred meters. These are called support networks and they support communications in a terminal access network or a computer-to-computer network. Such networks are provided by data processing equipment providers and network management tools are available and can be used successfully.

2.4.2 Summary of Version Mismatch

We have discussed many real-life examples dealing with issue of version mismatch as is seen by users and network managers. From these examples, some common version mismatch attributes has been emerged. Another important point emerged which can be used to support the definition of version mismatch problem is the derivation of common factors involved during the creation of version mismatch problem. Version mismatch can exist because of the following reasons:

- Mismatch due to year of recommendations (EXAMPLE 8).
- Mismatch between proposed architectures (EXAMPLE 9)
- Mismatch due to system software release date(EXAMPLE 3 and EXAMPLE 5)
- Mismatch due to options supported by the system (EXAMPLE 1, EXAMPLE 2, EXAMPLE 6, and EXAMPLE 7).

2.5 Version Mismatch as a Problem:

Figure 6 illustrates a history of X.25 recommendation over the period of twelve years from 1976 to 1988[BuPu89]. This paper [BuPu89] discusses the evolution of X.25 and step-by-step inclusion of many services and functions in the formal standard definition. Also this work has pointed out that certain services were classified as **Essential**. CCITT stressed that those Essential services must be provided by all PSPDN's as stated in Recommendation X.2. In this paper, an crucial statement is made which can be considered as the basis of this thesis: **"Currently, a minority of PSPDNs have fully implemented the 1984 version of X.25. However almost all networks have provided schedules to do so; it is only matter of time before X.25 1984 is universally available"**. So there was a time when some PSPDNs implemented those essential services and other did not. The question is that how did the mixed population of network subscribers and network managers reacted to situation where some of them were able to access a group of services and others were not. Also in this paper, a conclusion is made that X.25 can be viewed as the most universally used protocol due to the upward compatibility. The version mismatch problem as is seen through many examples separates the issue of upward compatibility from the interoprability between two distinct version of software. Two version may be upward compatible but may not support all the functions and services when accessed from a third remote implementation version. That is the broad issue we are discussing in this thesis as related to network management and standards evolution.

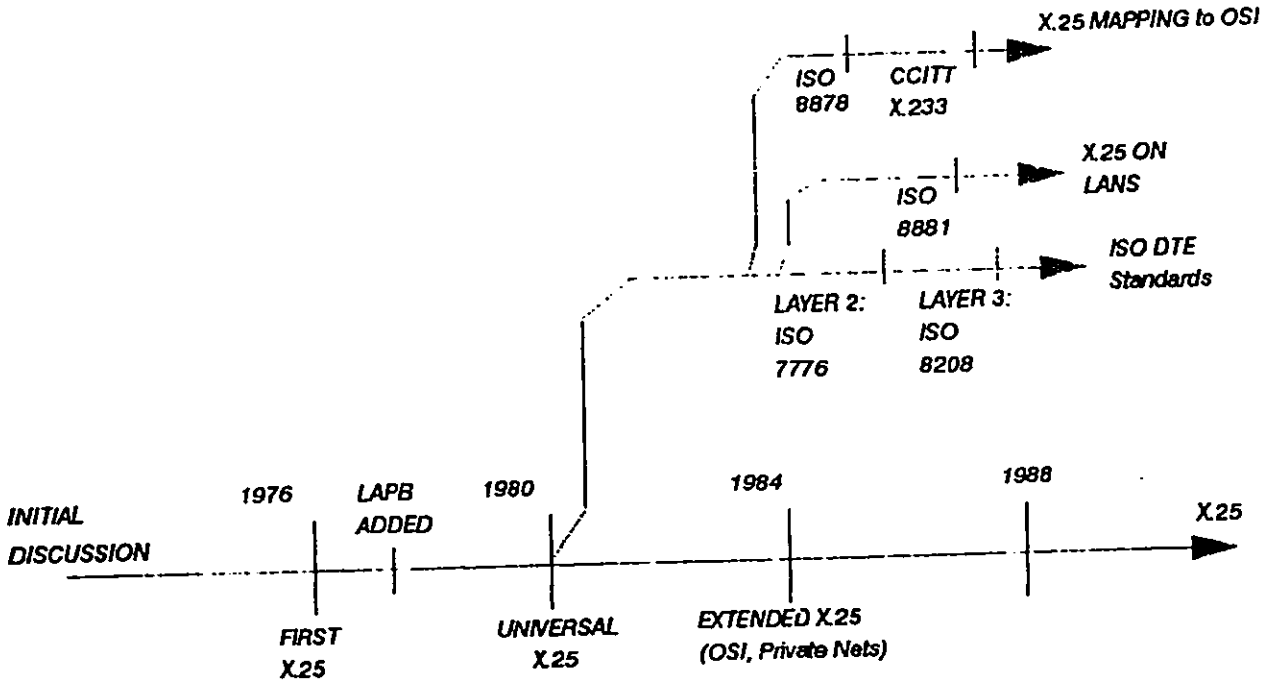


Figure 6: Timeline for X.25 and related standards(BuPu89)

Four formal recommendations were released by CCITT. Horizontal axis shows the number of times X.25 recommendation has been published during the last decade.

Vertical axis shows the possibilities of distinct software implementations which may be released based on those recommendations. If we consider all vendors involved in developing X.25 based software all over the world and selling it to different networks then this number will be combinatorially explosive. Clearly, this results in implementation versions mismatches both within networks and across networks.

In the following sections, We now precisely define the version mismatch problem based on the discussion we have had so far.

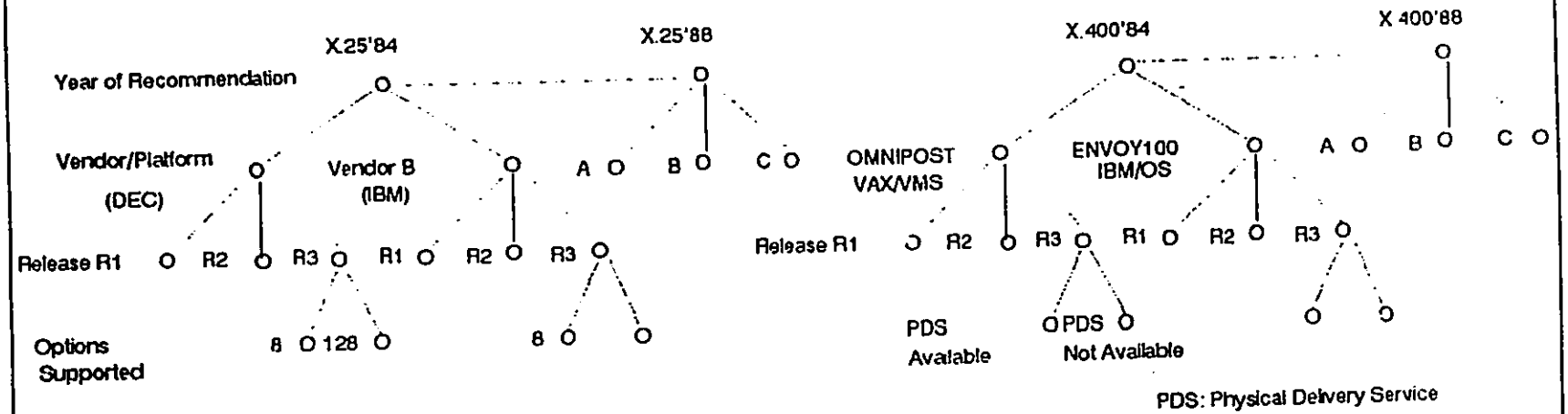
2.5.1 Definition of Version Mismatch

A Version Mismatch exists between a pair of implementations of a protocol which differ with respect to one or more types (called levels) of attributes. In this thesis, we consider four levels of attributes of version mismatch, namely:

- Year of Recommendation
- Vendor proposed architecture
- Software release number
- Supported options

From Section 2.1 to Section 2.4, we have covered many real life situations and examples related to network operations and version mismatch. Based on that discussion, we will briefly summarize the version mismatch levels in terms of attributes of version mismatch.

VERSION GROWTH TREES Figure 7 - a



EXAMPLES OF IMPACT ON RESPECTIVE GROUPS Figure 7 - b

	USER		NETWORK MANAGEMENT		STANDARD DEVELOPMENT	
	X.25	X.400	X.25	X.400	X.25	X.400
Year	High	High	High	High	High	Extremely High
Vendor/Platform	Low	Low	High	High	Low	Very Low
Release	High	Extremely High	Extremely High	Extremely High	Low	Very Low
Options Supported	Extremely High	Extremely High	Extremely High	Extremely High	Low	Low

Figure 7

Level 1 Mismatch There may be two versions of software which are direct derivatives of two independently and separately published recommendations.e.g. an implementation based on X.25'84 and an implementation based on X.25'88.

Level 2 Mismatch Communication software developed by two hardware vendors - Company A (DEC) and Company B (IBM). For example: CCITT's X.25'84 recommendation is implemented by numerous vendors for their distinct hardware architectures.

Level 3 Mismatch Companies may produce more than one release of software for various target platforms. For Example, vendors may need to support many releases of communication software (X.25) concurrently.

Level 4 Mismatch Two distinct implementations of X.25 may support different static user selectable options such as window size modulo 8 and modulo 128.

Such levels of mismatches can be illustrated using a tree called a VERSION GROWTH TREE (**Figure 7.a**). We have drawn two version growth trees one each for X.25 and X.400.

Each tree consists of a set of elements called nodes. Each recommendation or standard can be represented by one tree network. In the tree, the first layer of nodes represents recommendation version, the second layer represent vendor and their respective hardware platforms, third layer represents a release corresponding to a software platform and the fourth layer represents distinct options or groups of options.

To find a distinct version of software, one has to follow a path from the first layer of node to the last layer of nodes. A path is a sequence of adjacent nodes which is four and the length of

the path is the number of arcs it contains which is three in this case.

In other words first layer nodes are associated with only one attribute (year of recommendation), second layer nodes are associated with two attributes (year of recommendation and vendor), third layer node have three attributes

(year of recommendation, vendor and release number) and fourth layer nodes have four attributes. Hence by following the path from top level (Year of Recommendation) to bottom level (Options supported) will represent a distinct version of software.

As we have seen that there can be four attributes which can contribute for the creation of versions: Year of recommendation, vendor/platform, release number and options supported, which can contribute in the creation of a distinct version.

Another dimension to the above discussion which we have not considered is **level of impact** of each attribute as well as the impact of all four put together on people responsible to alleviate VM problems. Individual version mismatch attribute may have different impact on respective bodies such as users, network managers and standard developers while they deal with version mismatch related problems. It is extremely difficult to quantify the level of impact but we can provide relative terms to differentiate between serious attributes and non-serious ones.

We know subscribers access X.400 applications through X.25 network. Version attributes affect their environment drastically. We will discuss some of those attributes as seen by the different groups.

1 "Year of recommendation" of X.25 and X.400 is highly critical to the subscribers. The number of services usually available to them varies with the year of implementation[SiDo88].

2. The platform on which X.400 application is running is not important to users. For example, few subscribers of ENVOY100 are interested in the vendor and the make of hardware platform over which the application is installed. Also they may not be interested in the hardware used to built X.25 network. There may be many upgrades performed on X.25 packet switching network hardware without any user involvement.

3. On the other hand, users are always impacted by the release of the application and communication software. Each time ENVOY100 applications is upgraded, attempts are made to educate users before hand so that they can adjust their environment according.

4. Which options supported by the application and communication implementations are also critical to the subscribers environment. The options supported by the network will limit and control the capability of the services provided to the users.

This analysis has been summarized and illustrated in Figure 7b where first two columns show relative importance of version mismatch attributes for the network users. The next two pairs of columns depict the relative importance of version mismatch attributes to network management and standard development groups.

Chapter 3

A New Process for Prioritizing Functions during Standards Development Cycle

3.1 Introduction

The desired use of applications installed at different nodes can be achieved only if the nodes can freely communicate with each other. This depends entirely on the stability and quality of the communication software. Both implementations application software and communication software must be based a nationally or internationally acceptable standards to achieve global connectivity. Also, implementations must be consistent in terms of the number of functions they can support and number of services they can provide.

Before installing a new release on nodes, network manager should know the functional capabilities of the implementations. He should know exactly the number of missing functions and services. In practice, they rely heavily for this information over software developers. Based on the information they receive from software developers, managers often provide list of services the network will offer to the subscribers. While in operation, they find services completely missing. Also certain services not performing consistently and correctly. This inconsistency may be caused by the lack of understanding of the specification standard by either network manager or software designer.

The specifications are usually written in natural language. The process of writing the specification in natural language have their own drawbacks.

There are cased when the functions and their corresponding descriptive specifications are extremely general and other times they are extremely precise.

The primary task of the software developers is to identify functions in the specification and then try to implement them based on certain priorities established by the design team or based on customer requirements. Implementation of a function is derived from the specification standard which is in turn part of the national/international standard.

Standards such as X.25, X.3, X.28, X.29 and X.75 have been implemented by many independent vendors. There may be an implementation of a function which can differ from other implementations. A problem may result because of different interpretations in the specification written in natural language. The interoperability between such implementations where the differences are caused by interpretations is an issue which is much more complex than the version mismatch issue and it will not be discussed any further in this thesis.

Function Categories

Functions and their associated services in specification standards are categorized based on following areas.

1. Mandatory functional requirements;
2. Optional functional requirements;
3. Acceptable/unacceptable limitations on parameter values.

We will focus on variations in these requirements to define the version mismatch problem.

3.1.1 Mandatory functional requirements

Mandatory functional requirements are explained in the specifications of the function without the implementation of which the availability of desired mandatory services are not possible.

The result of such a function could be a direct service to the user or it could be an operational benefit for the network.

For Example

The link layer procedure in X.25 Recommendation contain two window mechanisms: Modulo 8 in LAPB is a mandatory function.

3.1.2 Optional functional requirements

Optional functional requirements are related to a function whose implementation may not be necessary. Implementation of a optional functions are left to the discretion of the implementation team. A final implementation may not lose its operational power if any of the optional functions is not implemented.

For example

Modulo 128 in LAPB is an optional function.

3.1.3 Parameter limitations

Each function is designed to perform a set of operations and these operations can be controlled externally or internally by selecting parameters values. These parameter values can control the effect and result of the function's output. Usually a limited number of parameter values are acceptable in a function. The function's implementation can be designed to accept a wide range of parameter values. Implementation of a function would not accommodate any parameter value. It can be designed to reject unacceptable parameter values up front while sending alarm signals.

3.2 Definition and selection of functions

There are many ways communication functions can be grouped together based upon their usefulness, necessity, and advantage. A **complete set of functions** includes all the functions defined in the standard.

Functionality of a function consists of large number of variations used to achieve all the results outlined in the boundary of a function. Standards and Recommendations are full of such definitions and all the functions outlined in a standard can be called a complete set of functions. In practice, not a single implement implements a complete set of functions in their final product. Also, Standards never recommends implementing each and every function specified in a standard.

On the other hand, software developers choose functions before the start of implementation based on the decisions made by the product manager. Such decisions are the direct result of a large number of variables such as financial situation, available time, end-user demand, and the last but not least target node constraints. Almost all the time, software developers have these obligations: support a healthy communication environment and compete and survive economically. The above process of choosing functions before implementations gives birth to a limited set of functions. A limited set of functions can be defined as a group of functions which is treated as limiting factor on implementation group. The final product of such implementation effort would provide a reliable and error-free communication link to the target users as long as all the nodes are using a copy of that implementation.

3.2.1 Functions and Options

The common thrust behind the process of the addition of new functions or the alteration of functionality of existing functions had been discussed. The approach to add, delete, or change functionality of a function will be discussed from another perspective, that is, from **the degree of importance** of function at the time of its introduction. Introduction of new function into a standard can result because of any of the following reasons stated in question forms:

1. Why is the implementation of a function important and for whom?
2. Who is effected by the availability/non-availability of a function?
3. Is the provision of a function is the responsibility of node, network or country in which the network operating?
4. Who is using the services of a function: the user, the network or both?

Definition

Function(f): A function can be defined as any realizable, codeable set of rules outlined in specification as one unit which can be taken out or added into the system as a whole without any change in the functionality of the remaining product.

Thrust factor behind Functions:

Implementation of a function may be the result of the following thrust factors.

Inclusion of a function in the specification is not only decided by the standard development group but also due to the active support in favour of the function by network owners and network subscribers.

Decisions taken during the design, development and implementation phases to include or exclude a function are again different . One reason may be that a

particular function is needed because it provides a service better than a competing network.

The third and most important factor is because a user group is willing to pay for its initial cost or a network is willing to pay because it is required by the network to have that particular function implemented to achieve interoperability with other networks successfully.

Each specification document describes a collection of numerous functions and their associated services. One major goal of protocol specification is to provide a basis for implementation. Some specifications methods facilitate this goal more than other. On the other hand, such descriptions must be easy to understand and precise. The use of natural languages gives the illusion of being easily understood, but leads to lengthy and informal specification which often contain ambiguities. Such ambiguities are difficult to check for completeness or correctness.

The selection of function for final implementation is not easy. Most of the stake holders influence selection process. Consider a function selected by network owner. The reason for this selection could be that the service provided by the function is solely beneficial to the performance of the network. The owner of the function is willing to pay all the cost required for the its implementation. Situations like this always crop -up and always contribute in the process of the selection of functions.

There are two other forces influence the selection of the functions: network subscribers and implementation groups. There may be a function in the specification tagged as "optional" but in great demand in the subscribers

circle. Such functions often become part of the final implementation. There may also be a function needed by both network owners and network subscribers as it is described without making a significant change to hardware platform used during implementation. The required hardware upgrade may not be available easily because of the technological limitations. Organizations may be faced with a choice between spending major resources to develop such platform or wait for couple of years and let the technology develop on its own.

3.2.2 Binary functionality

A project manager who is responsible for implementing specification standard has a choice weather to implement a function if the effort to achieve a particular goal fits the economic and technical resource boundaries of the company. The availability of the function to the network customer then depends upon the software development company's goals and intentions.

While introducing a binary function and its associated service to the network customers, network managers can take the following crucial steps:

- Upgrade required network equipment
- Acquire technical knowledge about the function and service

Network Managers have a choice of saying "yes" or "no" to users when asked about the service availability.

Marketing people can plan ahead before service availability time and educate the users about the new coming feature and the implications on the hardware or on other physical resources, if any.

A function explained in the protocol specification standard which can clearly

be separated from other functions falls into the special list made by the software designers. Such functions can be implemented as separate objects. Such function are always implemented in their entirety. These functions and their functionality can not be separated into sub-parts before implementation. There is no intermediate state between the two extremes: complete availability or complete unavailability. We will name such functions as **binary function.**

Networks are vulnerable from the quick introduction of binary functions. Their may have been a function which was intentionally omitted during one release but made available in a succesive release. Once such release is installed, there may be a well-informed user out there who is anxious to use the new available service just for the exploration, or to improve his productivity, or to enhance his environment. That can lead to a situation where user can completely disconnect himself from remote system.

Table 1 indicates list of X.25 (network layer) binary functions.

X.25 Binary Functions

1. Incoming call barred
2. Outgoing call barred
3. Reverse Charging acceptance
4. Incoming call barred within CUG
5. Outgoing call barred within CUG
6. Bilateral Closed User Group
7. Bilateral CUG with outgoing access
8. D-bit modification
10. Fast select acceptance
11. Call redirection
12. Call deflection

13. NUI subscription
14. NUI override
15. Local charging prevention
16 Incoming call barred
17. Outgoing call barred

TABLE 1: Binary Functions in the X.25 Standard.

3.2.3 Non-binary functionality

A packet switching network may want to upgrade a remote DTE- access baud rate from 2400 baud to 9600 baud. It sounds simple but it requires months of planning by network managers and includes real life testing, user education, network support team education, execution of related field trials, upgrading of documentation and equipment customization of modems, modem racks and switching equipment. Similar functions where baseline implementation exists and only new options required to be added or deleted are named as non-binary functions in this discussion.

In other words, functions which can be separated into pieces before implementation can be named as **non-binary functions**.

The complete implementation of a function requires the implementation of a sub-componen t of that function. User usually has a choice to negotiate any one of the small functionalities. Such negotiation can take place at the beginning of the call set-up or during the process of data communication. Networks usually end up supporting few such options for the users.

It is considered as a problem if no options relating to a function are supported by an implementation. If all the options of a function are not

supported, the magnitude of the problem will be not be great.

Examples:

Options are built around functions and users are allowed to choose options to satisfy the specific demand before the service contract is signed between the user and service provider. Subscribers may be authorized to select an option on the fly at the beginning of the establishment of a communication session.

Before utilizing the X.25 protocol service, the negotiation to select maximum packet size to transfer or receive data is common practice. There is a price to pay by the network for this user-selectable option. Networks may or may not offer such services which let users choose a packet size from a pre-specified list. Users are not allowed to select a packet size recommended in the specifications standard but most remain within the boundary of available parameters offered by the implementation.

A function specified in specification standard which can not be separated from the functionality of another is a non-binary function. Such functions can not be implemented as separate objects but in a collection of more than one smaller functions. Project managers responsible for implementing the protocol specification has a choice for implementing the function in part or in whole if the total effort to achieve his goal is economically and technically feasible. Such functions are implemented in parts. Small number of them are available in their entirety. (e.g. no network supports every standard-specified baud rate at any given time).

Let us have look at situation when a larger packet size is requested and then

used by the sender. The packet containing large bytes of data would require the node to reserve large parameters for error correction routines large work areas for error correction; large storage areas is required in the RAM to store the packet momentarily. Provision of large packet size puts stress on node's dynamic resources. Usually network dictates maximum packet size availability.

Parameter values negotiated during the communication session or before the communication session are extremely important from the customer's point of view. After successful assignment, the user is then restricted to design his environment to accommodate those parameters and grow within their boundaries. On the other hand, negotiated values responsible for stressing the node throughput or overall handling of traffic are usually sensitive issues for the network operators and managers.

Table 2 indicates the functions, the values of which the user can negotiate with the network operating organizations.

X.25 (network layer) Non-binary functionality

1. Degree of flow control

2. Level of throughput

3. Type of charging information

4. Supported size of packet

5. Supported window size

6. On-line facility registration

Table 2: X.25 Non-binary functions.

3.4 Core of the specification standard

Protocol specifications are made of a set of functions responsible for complete and accurate communication between two communicating parties. These functions are specified in sequence in the specification standard to maintain inter-dependency and functionality overlap.

It would be interesting to put some effort in establishing boundaries around group of functions depending upon the importance from both user's and network-owner's points of view. Sometimes in the standards, functions are flagged as mandatory not because of technical reasons but because of political or economic reasons.

The life cycle of a protocol specification constitutes a number of stages from the time an initial concept is developed to a point when specification is presented in draft form. Draft standards are discussed dealing with a function by function at a time. In case conflict occurs and a decision is not reached concerning the validity of a function or a procedure, voting takes place. Standard development is a rigorous process. Even if this process is long, less technical and more political, it has successfully produced many widely acceptable standards and recommendations.

What is the core of a specification?

Definition:

The core of a specification standard is defined as the set of functions without the implementation of which the protocol

could not perform at the desired level of performance and satisfaction.

Performance:

As the use of computer networks increases because of new applications, the reliability and performance of the network becomes more critical. As the complexity increases, their reliability becomes more difficult to control. We can deal with issue of controlling the performance of the network by establishing performance related milestones outlining the expected performance criteria which must be met. The following are some of the performance parameters: throughput, response time, and window size. These are the basic parameters of communication systems and should be considered before the development of the protocol.

Satisfaction:

Network subscribers are considered the lifeline of successful networks. Network growth in terms of number of subscribers tells a lot about the network. Success of any network hinges on the number of satisfied subscribers in terms of quality of services and quality of the support they receive from network. The quality of the service is extremely delicate issue for them. Information must receive to and from them reliably.

Where to define a core:

Specification standard describes the function or protocol. Protocol descriptions play a key role in all stages of protocol design. The document is used by software designers as reference to understand the proposed functionality of each function, services or module. In addition to all the relative information provided in these documents, the standard development organizations must add information defining the core of the specification which once implemented transforms into core of the protocol.

The core of the protocol is not often straight forward to determine from the specification document. To determine the core of a protocol from a specification standard requires complete understanding of the users requirements, network requirements, inclinations of implementation teams as well as the nature of the organization responsible for developing standards.

Discussions so far has not provided an acceptable solution in terms of defining the core. The most reasonable place for the core is in the specification standard because such a documents is solely developed by groups representing all possible interest areas. Standards and recommendations provides extremely generic framework to accommodate all possible requirements. It is important to determine whether a specification provides enough description and whether there are descriptions which can be ignored.

It is clear from the discussion we have had so far that there are a large number of constraints which should be considered before establishing a boundary around the core of the protocol.

Properties of the core:

The core of the protocol may never be changed or modified during the evolution of a standard. If attempt is made to change the core of the protocol the result would the creation of a new "proprietary" protocol implementation. The new corresponding implementation may not inter-operate with a previous implementation.

The core of the protocol should be defined such that there is infrastructure which would "always" support the use of protocol and services without any change in the user environment once initially established with the agreement between the user and the network.

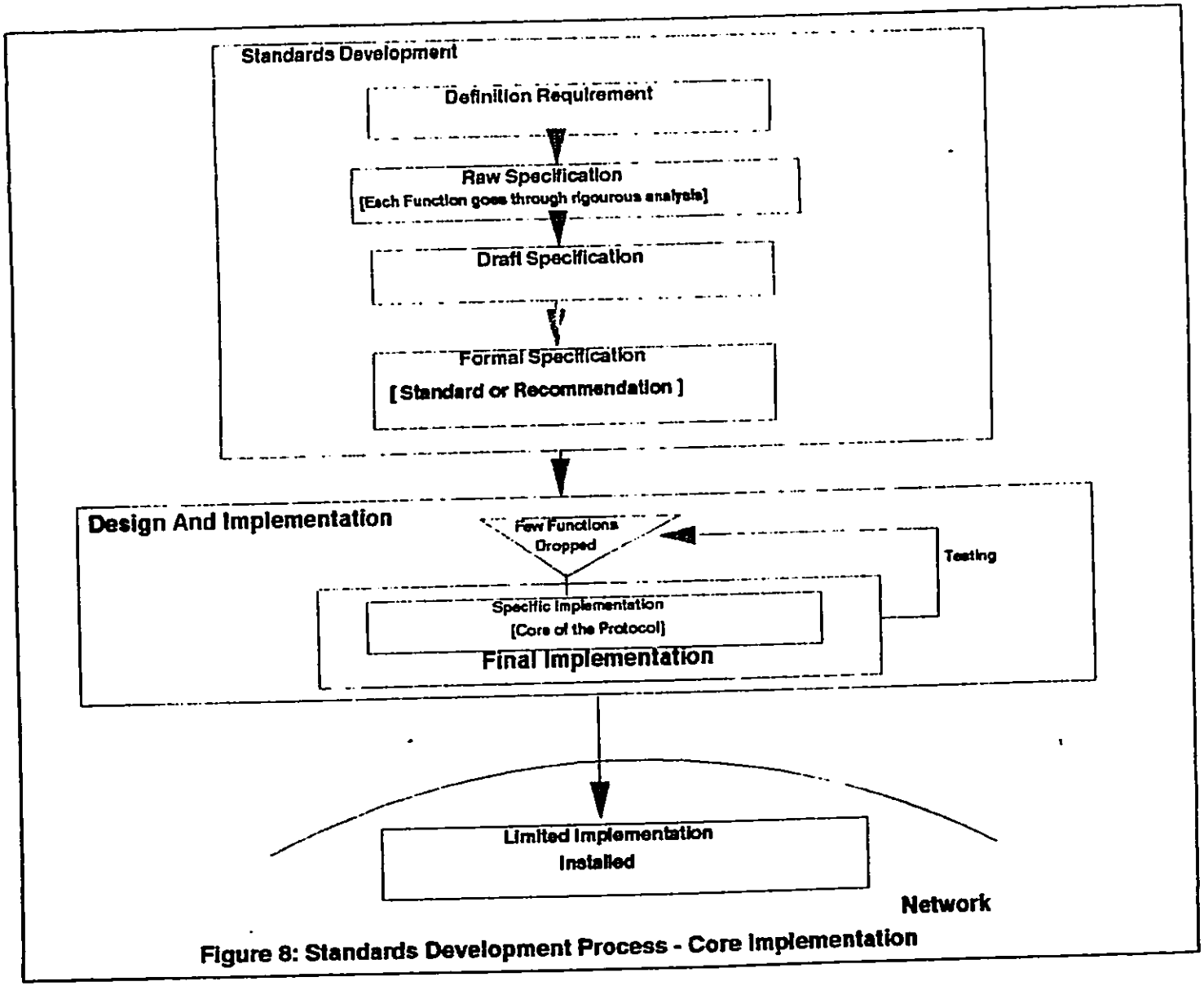


Figure 8: Standards Development Process - Core Implementation

If a situation arises and existing protocol implementation needs a new function to be incorporated, it should be done in such a way that a current user would continue using services without making any change to its existing environment. He/She may be willing on his/her own to upgrade his environment to receive the benefits of the new available service. But subscribers should always have the option of falling back to the core of the protocol and receive the core services.

Core of the protocol include a limited set of communication functions which can effectively provide communication link establishment between two communicating partners and then, in turn, provide a communication medium to transfer data reliably over that medium.

How to define the core:

The core can be defined formally using a natural language which can become integral part of the specification standard. Functions can be tagged using this new approach for the core can be defined by using similar techniques as currently used to tag mandatory and optional functions. **Figure 8** (STANDARD DEVELOPMENT PROCESS -CORE IMPLEMENTATION) illustrates the sequence in which specification standards are produced and software products are implemented. This figure also presents the view about the implementation of the core of the protocol as it is derived from the specification.

Another formal procedure can be finite state machine representation of protocol specification. This can be achieved by drawing all states and transitions in the Finite State Machine model where the core is distinctly represented. Inside the core, making changes to existing state, deleting an existing state, or adding a new state

results into the change in the required functionality of the protocol.

What is beyond the core of the protocol?: Once the core is defined, the next step is to define the variables which are user controllable, alterable and selectable to make the protocol flexible enough and powerful enough to accommodate a finite set of user requirements, and at the same time, the subscriber can use the limited set of network provided services successfully.

3.3.1 Core of the protocol: an example

3.3.1.1 The physical layer

The bottom layer of the OSI [OSI1] stack, called the physical layer, includes the electrical interface for the driver and receiver circuits and the mechanical connector equipment. There are many standards available for the physical layer. Let us name a few.

- EIA RS-232-C/ RS-232-D
- ISO standard 2110
- CCITT Recommendation V.35
- EIA-423-A, EIA-449
- X.21 bis interfaces as described in CCITT Standard X.25.

Let us discuss some of the basic functions from the most accepted standard used for this layer which is EIA's RS-232-C/RS-232-D because of its simplicity and completeness. In the physical layer implementation, the options are to be provided for 1200, 2400, 4800 and 9600 bits per second. The baud rates of 4800, 19200, 48000, 56000 or 64000 are not available on all networks. When the 56,000 bits-per second-or-more-rate is required, the CCITT V.35 full-duplex interface is used. Other parameters of this layer include full-duplex transmission,

simplex transmission, continuous carrier, parity checking and parity setting.

3.3.2.1 Core of RS-232 standard - a physical layer protocol:

There are many applications where the RS-232 standard is used and will require the use of Pin 2, 3 and 7 to realize the successful transmission. Without going into detailed analysis of the design and purpose of each pin in this standard, let us define a minimum pin configuration required to achieve the desired result of serial transmission. It is known that the standard provides many user selectable and setable options (parity selection: odd, even, none) and many redundant pins are provided for secondary equipment as well as to support asynchronous and synchronous communication. The minimum possible pin configuration required for this standard is explained in Appendix 1.

3.3.2. Data link layer

Before discussing the associated functions and services involved in this layer, we will first study following text quoted from CCITT's X.25 Specification.

Section 2.1.5 of document AP IX-49-E states," **For those networks that choose to support both the basic and extended LAPB sequence numbering services, the choice of either basic mode (modulo 8) or extended mode (modulo 128) may be made at subscription time.**"

Section 2.1.6 of document AP IX-49-E states," **In case of those networks that support both LAPB procedures and LAP procedures, the DCE will maintain an internal mode variable B, which it will set as follows:**

- to 1, upon acceptance of an SABM/SABME (modulo 8/ modulo 128) command from the DTE, or upon issuance of the SABM/SABME command by the DCE;

- to 0,."

The specification standard - LAP and LAPB- provides solutions and guidelines for possible applications based on individual user as well as network requirements. Under such circumstances the specification can be transformed into many valid but different implementations. The dark side is that communication within networks loaded with single vendor provided implementations will be much smoother as compared to communication across networks when the requirements differ drastically from network to network. The reason is that within a network it is expected that all nodes run similar and consistent versions of software.

Suppose a network supports LAP protocol with modulo 8 and another, newer network is supporting LAPB both at modulo 8 and modulo 128. The user on the second network can subscribe to the modulo 128 option but he would not be allowed to communicate with first network while using 128 option. On the other hand, network would suggest to falling back to LAP and modulo 8 which again depends on whether the LAP is available in the second network. If the LAP is not available in the second network, the communication link across two nodes will be impossible to establish.

3.3.2.1 Data link layer protocols:

There are many acceptable specification standards designed to support this layer.

- Link access Procedures (LAP)
- Link Access Procedures - Balanced (LAPB)
- LAPD
- High-Level Data Link Layer (HDLC)
- Synchronous Data Link Control (SDLC)

Introduction:

The most important function of the data link layer is to transfer information reliably across the physical links. Its functions include synchronization, flow control and error control. This layer is also responsible for providing services to Network Layer such as the transfer of information to the remote receiving entities.

Communication at this level can also be achieved by exchanging information without waiting for the receiver to send acknowledgement. Flow control and error correction mechanism which is an extremely important requirement of the modern networks are not used. Transfer of data with flow control and error-correction functions are not reliable. To tackle this reliability problem, the information is transferred only after receiving the acknowledgement from the peer receiving entity. Almost all the protocols standards named above provide discretion for similar functions.

Protocol functional description:

Communication is achieved by exchanging pre-defined bit patterns known as frames. Bit patterns represent distinct commands and responses used between two communicating partners to establish control and information transfer.

Frames consists of several components such as:

flags, information about the start and end of the frame which controls, identify different control frames; CRC, the error checking code; Information,

1. Information transfer function

This function is responsible for transferring information inserted into frames. The function of the information

command is to transfer across a data link frame containing data.

2 Ready to receive function

This - receive ready - frame is used by DTE or DCE to indicate it is ready to receive the information frame.

3 Not ready to receive function

The opposite to 2.1 is when DCE or DTE sends a frame to indicate it is not ready to receive the information frame.

The above three commands/responses are the basic set of functions required to transfer data across the network. Think about the situation when one of the information frames is lost or received incorrectly or transmitted incorrectly.

4. Frame recovery function

This type of frame, the reject command/response, is used by the DTE to request from the sender retransmission of "n" number of frames previously transmitted unsuccessfully by the sender.

The fourth command/response frame adds functionality to achieve error-recovery mechanism.

To make the operation at the data link layer transparent to other unrecoverable situations by the above four frames, one needs to add more functions to accommodate undefined states.

5. Frame reject function

The FRMR frame response is used by DCE or DTE to report an unrecoverable error condition. This can occur when the transmitted frame is valid but one or more fields is undefined.

6. Disconnect function response

The function DM frame is a response to report a status where the DCE or DTE is behaving as if it is disconnected from the link, logically, not physically.

7. Disconnect function command

The function DISC command is the command used by DTE or DCE to terminate the mode it is presently in and suspending operation.

3.3.2.1.1 Binary functions and non-binary functions:

In order to apply what we have discussed in relationship to binary and non-binary functions, let us look at commonly defined functions described in all the related specification standard available for this layer. The next step would be to investigate how many of them can be categorized as binary and non-binary functions.

The nature of the final implementation varies from product to product if personal discretion is applied by the designer or the implementor. To make the standard acceptable and to make the implementations interoperable, standard must cover all expectable functions and also each function must be extrapolated to its limits so that it will fit in a given environment.

First of all let us visit the functions of the data link layer again and study which ones of them are binary functions and which ones are non-binary functions.

- Information transfer function

Based on the definition of the binary functionality, this function is a **binary function.**

- Ready to receive function

Binary function

- Not ready to receive function

Binary function

- Frame recovery function

Non-binary function

- Frame reject function

Non-binary function

- Disconnect function response

Binary function

- Disconnected function command

Binary function

The implementation of binary functions is more desirable in the software because of its modular nature. Standards should tend to convert non-binary functions into partial binary functions. The rest of the functionality can remain optional.

3.3.2.1.2 Core of data link layer protocol

The Link Access Procedures (LAP and LAPB) are described as link level procedures required for data interchange between a DCE and DTE over a single physical circuit (LAP and LAPB) or optionally over multiple physical circuits(LAPB). LAPB modulo 8 is the basic recommended service and is available in all networks. LAPB modulo 128 is recognized as optional, the subscription time selectable, but the extended sequence numbering service is not available in all networks.

The single link procedures (SLP) use the principle and terminology of High-level Data Link Control (HDLC) procedures specified by the ISO [ISO2,ISO3].

Synchronous Data Link Control (SDLC) can be classified as a subset of HDLC using the unbalanced, normal response mode class of procedures.

The data link layer specified in X.25 Standard is capable of supporting simplex, half duplex and full-duplex transmission configured point-to-point or multipoint facilities. It allows access through dial-up or dedicated facilities.

Dial-up and dedicated facilities are used to establish long distance communication.

The core of the data link layer can be defined with LAPB as defined in the X.25 specification document. In particular, the single link access procedures are the most suitable representation of this layer. Modulo 8 is the most acceptable and recommended parameter. The window for unacknowledged packets is to be optional between the first and the seventh frames. Data will be structured in 8-bit octets by the user. 256 user

data unit length should be implemented. The core must support half-duplex and dedicated facility.

Chapter 4

Network Management Architectures and the Version Mismatch Problem

4.1 Brief overview of types of network mangement architectures

Modern enterprise networks are composed of collection numerous variety of devices, systems, platforms, and network architectures. Enterprise Network may be a collection of local area networks, wide area networks, voice networks and mobile communication networks. These networks have matured over last few decades to satisfy different user requirements. As these systems grow they support multiple protocols, such as OSI, TCP/IP, SNA and DECnet. As a consequence, network managers find it increasingly complex to manage such networks. To perform their daily functions, they use available network management software products. In recent years, research on Integrated Network Management has made great progress, especially in the area of standards development.

The organizations developing Integrated Network Management standards can be divided into two prime groups: ***OSI-Based*** and ***Non-OSI-based***. In this thesis, we will focus on OSI-based NM standards. The major alternative NM standard is SNMP [STAL93], but it is not discussed here.

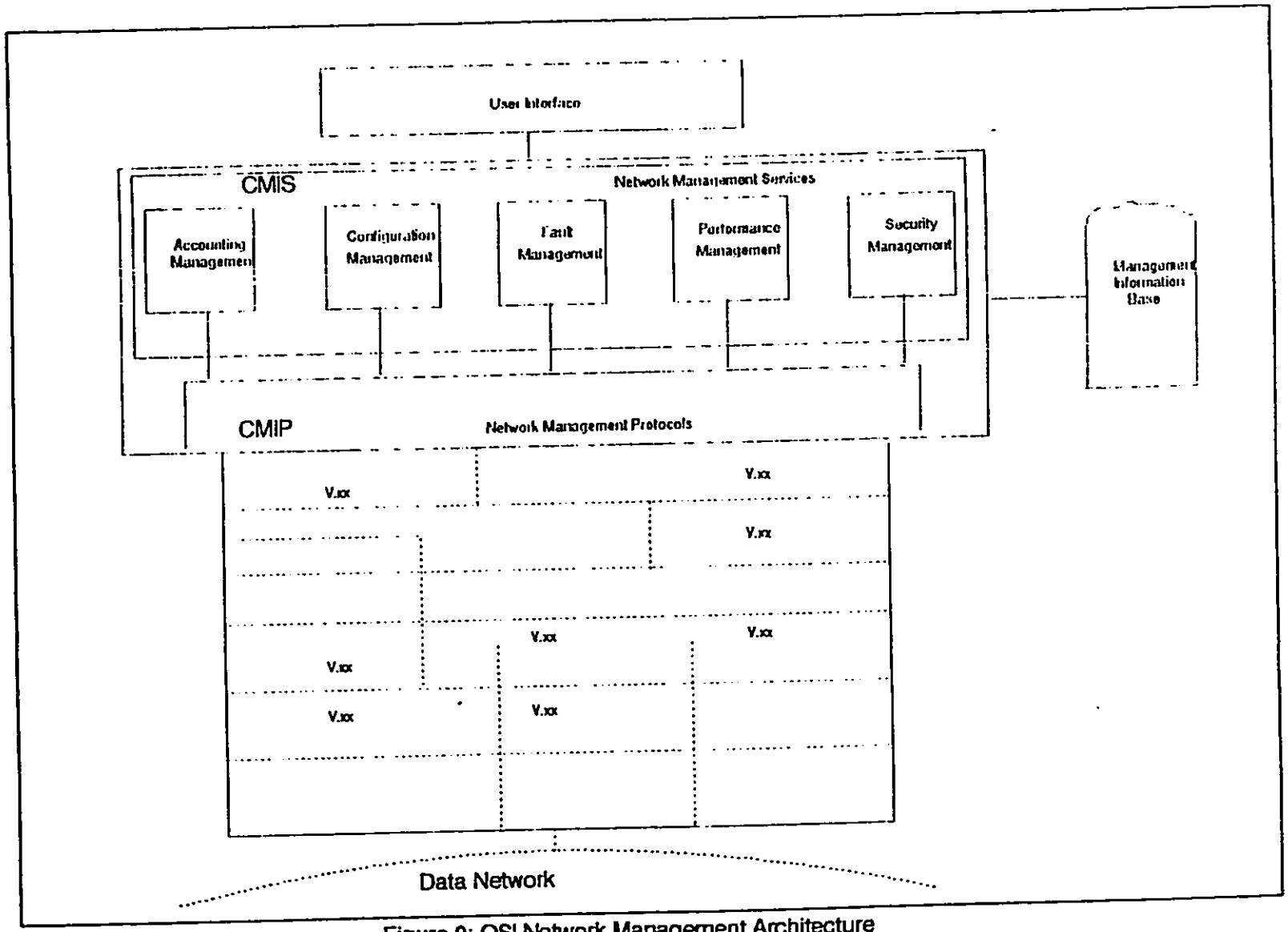


Figure 9: OSI Network Management Architecture

4.1.1 OSI/FORUM Architecture

The ISO Open Systems Interconnection (OSI) model is a logical grouping of the functions needed to communicate between computer systems [ISO1]. The functions are grouped into seven layers with the lower layers supporting the functions in the higher layers. The highest layer's functions are accessed by the user and lower layer's functions are used for stable communication.

OSI management standards draw heavily on the principles of object modeling based on ISO-OSI Reference Model. As illustrated in **Figure 9**, a simple model of OSI-oriented management can be described as a set of interactions between one or more managing processes and one or more managed processes. A managed process is responsible for one or more managed objects.

A managed object is an abstract representation of any resource from a management perspective. A managed object may represent a modem or it may represent a customer account[CARU90], a window size or a logical channel. The shared conceptual repository of all managed object information is known as the **Managed Information Base (MIB)** as is explained in related work[CARU90]. The information available in the information base is then manipulated by the services provided by the **Common Management Information Services (CMIS)** and communicated to a remote network control center according to the **Common Management Information Protocol (CMIP)**.

Within the MIB, information consists of attributes associated with objects. Each attribute of an object has a value. The CMIS provides services to retrieve(Get) and modify(Set) attribute values, and create(Create) and delete(Delete) objects. Such services provide the capability for manipulating and reporting managed object data.

ISO has tasked the ASC X3T5.4 to develop the OSI management standard [FERN89]. In addition to the network management services discussed below, this model attempts to represent three types of management functions [KELL87]:

- **Protocol management**,
- **Layer management** and
- **System management**.

Protocol management consists of the management of the internal mechanism by controlling parameters of each layer. Protocols can be separated into two categories.

1. Communication protocols(X.25, X.75, RS-232-C)
2. Management protocols(SNMP,CMIP/CMIS)

Management of two versions of protocols is a complex task. This task can be achieved by collecting detailed information about the exact differences between two versions of protocols.

Communication protocol management cause problems which, in most cases, are local in the nodes of data communication networks(LANs, WANs and MANs). Mismatches in communication protocols can result into problems such as loss of transmission link or loss of information.

Mismatches in management protocols can cause loss of exchange of management related information between nodes or networks. Such information is crucial to manage network resources

Layer management deals with the management of internal parameters associated with each layer. This includes state variable, timer values, thresholds, state changes, etc.

Versions of layers can cause problems at many levels. Since the functionality of each layer is restricted to certain tasks, the mismatch between them would only impact tasks associated wiith that layer. Mismatches between lower three layers are, usually, reflected as networking problems. Mismatches between upper layers can cause damage to the services provided by applications.

System management includes overall management of the resources of the system including protocols, gateways, multiplexers, terminals, users and computers.

OSI management architecture provides five categories of management services [KELL87, FORU88]:

- Accounting Management
- Configuration Management and Name Management
- Fault Management
- Performance Management
- Security Management

Accounting Management

Accounting Management is defined as the determination of cost of services and proper billing procedures.

Configuration Management

Configuration Management consists of facilities to initialize and shutdown managed objects, to change configuration parameters, to collect statistics, etc.

Fault Management

Fault management includes the detection, isolation and correction, of faulty operation. It also includes procedures such as fault notification, diagnostic testing and corrective action. A fault management module may mistakenly diagnose a fault because of a functional differences between two versions.

Performance Management

Performance management is defined as the evaluation of the performance of the system and also as corrective action to tune the system for improved performance.

Security Management

As the name implies, security management involves the security of the systems as well as of the network.

This concludes a brief description of OSI/FORUM network management architecture and its recommended services. In the following sections we will describe OSI-Based network management architectures in detail covering its objectives, and limitations.

4.1.1.1 *Forum Architecture Overview*

The Forum architecture has been developed to meet four objectives.

1. To provide a framework to support interoperability of systems that manage communication and computer networks.
2. To allow communication between multi-vendor systems.
3. To be flexible and extensible to manage a full spectrum of networks, regardless of size and kind.
4. To align with ISO and CCITT standards.

The version mismatch problem can arise due to following circumstances:

- when stringent rules to manage interoperability between different implementation versions are missing and

- when availability of interfaces to support transfer of such management related information is missing.

Forum has made an attempt to define management architecture and its associated terminolgy to manage interoperability between networks. Conformant Management Entity (CME) is the term used by the **Forum** to refer to a network management system that supports the Forum's standards - not including versions of standards- for exchanging management data[EmMa90]. The Forum is not only defining the ways to build management systems but also defining their inter-operation rules. The focus is on the interoperable interface that point where two different management systems meet to exchange management data. Before the successful inter-operation between two CME, they must share management knowledge.CME's, if defined as mentioned in Forum, are capable of interoperating but unable to handle data related to implementation versions.

FORUM has selected a protocol stack based on seven layer model that will support CME's configured as application layers. Unfortunately, issues and problems discussed in Chapter 2 and 3 related to multiple versions of stacks will not be dealt with Forums architecture. Imposing a restriction on selection of protocol stacks will also limit the application of FORUM architecture on the many existing architectures, plat forms, systems and version of implementations.

The **FORUM** has adopted an object-oriented approach to representing management data following the approach of ISO and CCITT[FORU90].

In the following section, we will analyze Forum architecture in detail. A **managed object** is an abstraction or representation of resources managed

across interoperable interface. A **resource** can be anything that is useful for management decisions such as a X.25 software version of a packet switching node, a modem or a personal computer. A single management object may represent a single network resource or many resources. The primary function of CME is to make visible 'managed objects' through a data reservoir known as **Management Information Base (MIB)**.

Several similar managed objects can be grouped together under managed object classes. The four properties that make up the definition of a managed object class are:

ATTRIBUTES
MANAGEMENT OPERATIONS
BEHAVIOR
NOTIFICATIONS

- ATTRIBUTE: can be thought of as data elements and values. For example, date rate of a circuit or main memory of a computer.
- MANAGEMENT OPERATIONS: are the operations that can be applied to a managed object. For example, data rate can be set to different values including disabling a value completely.
- BEHAVIOR: is the responses made by managed objects to the management operations. For example, if a specific services provided by X.400 goes down, then the state attribute of that managed object instance would have the value "disabled".
- NOTIFICATIONS: are the messages generated by managed objects. For example, disabling of a X.25 card may generate a system alarm such as "X.25 DTE #12345236 disabled".

To depict relationship between managed object classes, Forum Architecture uses an **inheritance tree**[EmMa90]. It also introduced a notion of **superclasses** and

subclasses. Each class has one superclass except the root of the tree. A class inherits all of the properties (attributes, behavior, operations, and notifications) defined for its superclass. For each managed object class, one or more rules must be written to identify a superior class and the distinguishing attribute.

Forum network management architecture has also many other known drawbacks such as its inability to answer many crucial question asked during management of heterogeneous networks[CeKu90].

1. If the network uses multiple protocol profiles, the administrators have additional problems, since the managed object definitions may be different for each protocol suite or profile.

2. Problems may arise if even a single management center does not comply to the network management requirements.

The existence of such discrepancies in the **Forum** architecture support the argument that the proposed architecture dealing with network management as well as the interoperability of the networks do not solve version mismatch issues.

4.1.1.2 Possible approaches to managing VM problems

The definition of "managed object" should include attributes dealing with specific version mismatch related resources. Two different X.25 versions can provide services different from each other and they both should be dealt with independently while defining 'managed objects' for them.

Different versions will map into different numbers of "managed objects". ATTRIBUTES definitions can include

specific details about version mismatch related information. Similarly, the other three properties that make up the definition object class -- management operations, behavior, notification -- may be defined to include version mismatch parameters. A management operation may involve disabling of a DTE from active CUG group. That function will activate the BEHAVIOR property to "disabled" and NOTIFICATION will be sent to the management center as "CUG #4 - disabled".

4.1.1.3 Examples of Version Mismatch problems which canot be handled in Forum

The management of two X.25 software versions will require a large collection of 'managed objects' to cover management aspects of all three layers as well as distinct manageable functions of each respective layer. The definitions of some of the 'managed objects' will be common for both versions while other will differ substantially. Such differences will exist because of the difference between two implementations. Certain 'managed objects' will be missing completely from one implementation as compared to an other.

Forum architecture is unable to deal with version mismatch problem in particular. Point 2 mentioned above should include a mechanism to provide network managers guidelines that will assist them to manage the interoperability between different implementations. It is in these areas that systems are expected to differ widely in support of different kinds of networks, requirements, technologies and markets and to facilitate product differentiation. This that can easily lead to creation of different versions of software and hardware.

Moreover, the following are some of the aspects purposely omitted from the Forum Architecture:

1. Technology on which system is based: computer hardware, operating system

2. The level of distribution or centralization

3. Specification of the human interfaces

The management of a network involves not only the management of applications and their associated services, but also the management and control of functions of individual layers[FORU88]. Rarely the network management deals with software stacks populated with multiple version of software[**Figure 9**].

4.1.2 *Non-OSI-Based:* SNA's View of Network Management Services

Each vendor has his or her own conceptual view of managing a network. Vendors tend to commit their resources to developing network management systems to satisfy demands which may be associated with factors such as the most commonly used network or the most commonly expanding and acceptable network.

SNA topologies have rapidly evolved from centralized, single-host, single-vendor entities into distributed control, multiprocessor, multivendor, multidata-centre processing environment [ROUT88] and to manage such networks is much more difficult compared to single SNA architecture. Each SNA host can be interconnected through packet-switching data network or by Token-ring local area networks and can also communicate with other vendor networks, such as DEC, Hewlett-Packard, Tandem and Wang.

Because of a large number of networks available which function under SNA environment, IBM has invested a large amount of effort to develop network management tools to perform management

functions. This effort has produced a widely acceptable package called NetView. NetView plays an important role to manage not only SNA based networks but also other networks communicating with this network by using proper (Software and Hardware) interfaces.

Under the architecture of network management called Open Network Management, the four areas are defined to cover all possible network management requirements.

- Problem Management
- Performance and Accounting Management
- Configuration Management
- Change Management

Problem Management Problem management is the handling of an error condition in the network. It includes such functions as:

- Detection and Resolution of problem
- Problem Determination
- Problem Diagnosis
- Problem Bypass and Recovery
- Problem Tracking and Control
- Problem Resolution

VM may be covered partially and unintentionally by these functions. Problem bypass and recovery modules may end up solving VM related issues.

Performance and Accounting Management:

It include functions such as:

- response-time monitoring,
- availability monitoring,
- utilization monitoring
- component delay monitoring,
- performance tuning,
- performance tracking
- control and accounting.

Configuration Management:

Functions like physical/logical resource identification and resource relationship identification are performed by this category of network management.

Change Management:

Change management is the process by which software change control, microcode change control and hardware change control is taken care of.

4.1.3 Elements of network management

First consider a network using a layered communication network architecture as recommended by the International Organization for Standardization's (ISO) Open System Interconnection (OSI) Reference Model. The basic concepts developed for OSI network management will include the following elements[EmMa90].

- Network administrator
- Network management - application
- Network management - protocol

Increasingly, communication managers are asked to plan and manage complex networks. In response to growing information need, service subscribers

tend to connect variety of equipment. With the evolution of technology, users continually replace their equipment with new improved versions of software and hardware without the prior knowledge of network managers. And then, try to access old applications with newer software and hardware versions. The services which were used by them successfully before they changed their environment may not accessible because of the new and supposedly better equipment. As a result, the phones starts ringing at the network management control center. Operation managers try to locate the problem and occasionally find that the existing application is not responding to a user because of the incompatibility between old and new communicating entities.

Let us discuss these elements in relationships to version mismatch problem.

Network Administrator

The role of a network administrator is usually complicated during the first few weeks and months of the arrival of a new release. The nature of complicity depends upon effort involved during testing different nodes. Most problems are usually detected during controlled testing. Version mismatch problems can only be detected when attempts are made to establish connection across networks installed with new releases. Especially, when the interoperability between new releases has not been tested. The amount of time required to detect version mismatch problems also depends upon the size of data traffic passing through the network.

Application Management

An important task to manage applications is to monitor, detect, isolate and correct the errors in the applications.

Nodes provide mechanisms to transmit data across networks from users to the applications and vice versa. Applications are resident on hosts which are connected to DCE's. Application software also pass through many releases due to sequential improvements and modifications. That particular aspect of evolution of application software into a multi-version environment is also a prime cause of operational breakdown in a network.

Layered Management Entitles

A network management system should manage services provided by different layers of an layered architecture. A network may have a large number of network management sub-systems, responsible for managing equipment provided by a vendor and responsible for managing a protocol stack capable of providing a type of communication. Also, a sub-component of network management can be responsible for managing a set of different versions of application software or communication software. It may be configured to provide a switching mechanism from one version to another at the request of a user or a network manager

4.1.3.1 Physical layer management

The main function of the physical layer is moving bits of information from the source to the recipient. The medium of information can be either analog circuit, radio channel or any other type of circuit. The management function at this level include control and management of the modem. This function may involve functions such as switching a modem off or on, and switching a remote modem into a specific loop-back test.

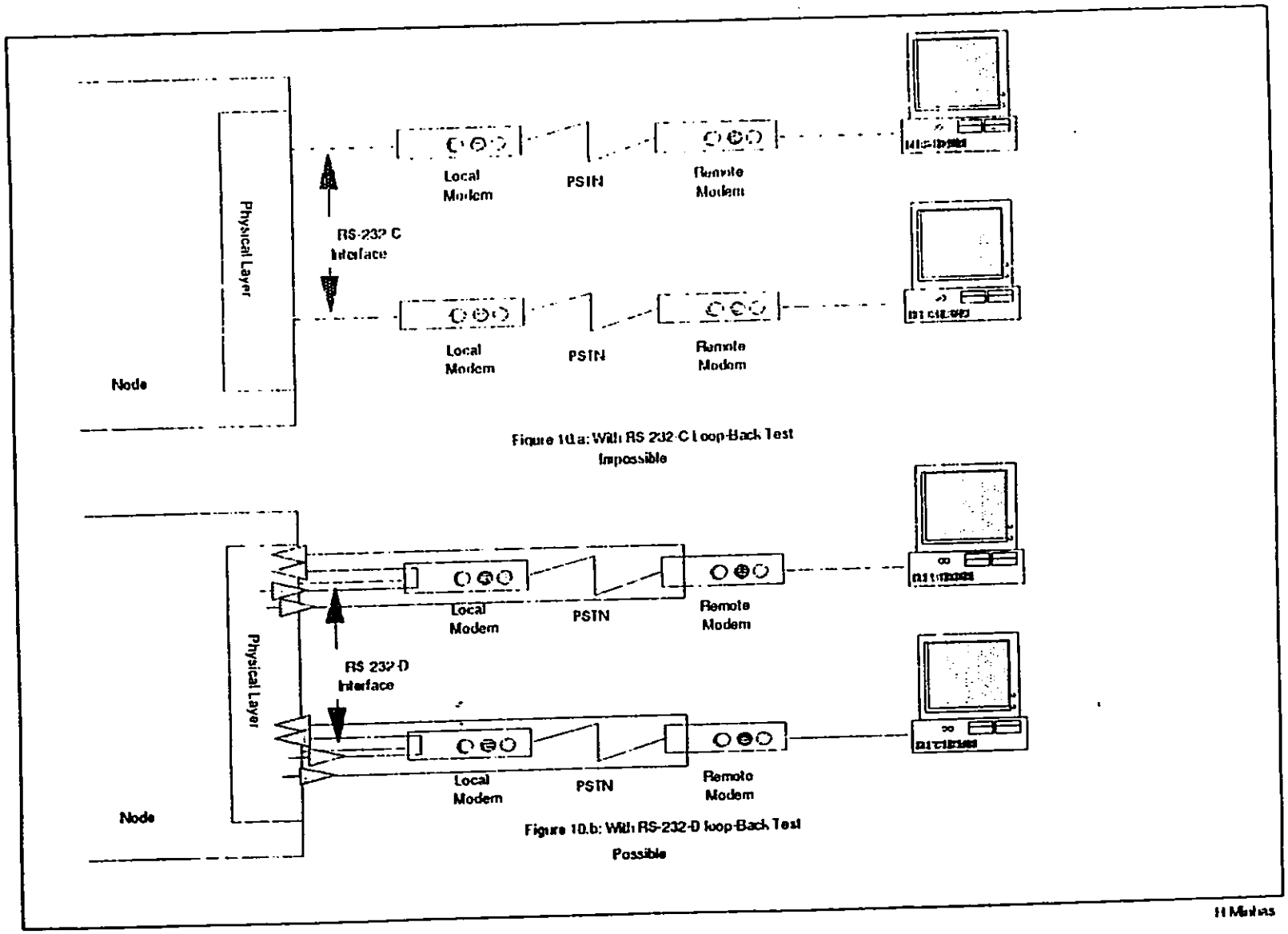


Figure 10.a: With RS 232-C Loop-Back Test Impossible

Figure 10.b: With RS-232-D loop-Back Test Possible

4.1.3.2 Issues related to the physical layer management

The commonly used physical layer standard is EIA's RS-232. This standard is widely acceptable for its usefulness and wide range of variations demanded by a serial transmission medium. This standard can be configured for a wide variety of applications while transmitting data serially over short and medium distances without modems, and long distances with modems.

RS-232-C standard require 25-pin type of interface. In this specification, pin 18 and pin 21 were left unspecified for future use. When this standard was upgraded to RS-232-D, those two pins were specified to perform physical layer management functions.

Pin 18 is used for local modem loop-back testing and pin 21 is used for remote modem loop-back testing. The signal on pin 18 remains high and transmitted data is looped back through the local modem.

Sender can verify the received data for accuracy and confirm the status of the circuit and the the modem.

1. Loop-back testing is performed to trouble-shoot problems with the physical lines. If proper software and hardware to support RS-232-D is not available, the management function cannot be performed **[Figure 10.a]**.

2. The mixing of different versions of software and hardware is extremely unrealistic for the management to achieve useful results. For example, loop-back tests can not be performed on RS-232-C hardware and software **[Figure 10.b]**.

4.1.4 Network management tools and version mismatch

There are a large variety of NM products, supplied by many vendors, to support various NM components. Most of these NM products support various specific components. For example, modem network management systems(NMSs) provide management for modems; varios SNA NM packages and products provide NM for SNA networks only. Furthermore these various products in general do not interoperate, often even for different products provided by the same vendor. This means that NM manager would require a large number of NM products from many vendors. This becomes an expensive proposition in terms of equipment and personel. Essentally, network management is donein a nonintegrated manner[CARU90].

The problem of managing networks depends upon the availability of suitable tools specifically designed for tasks associated with network management. Numerous data line monitors are available to trouble-shoot and test lower three layers of OSI-stack such as X.25 protocol or other related protocols. Very few test tools will provide built-in functions capable to detect functional differences between two specific implementation such as X.25(84) and X.25(88). It becomes functional testers task to overcome this problem. Testers would have to write test suites specifically for detecting version difference. The accuracy of test suites will depends upon the testers experience with different versions.

Inconsistency and incompatibility between tools are one of the causes for inspiring:

- organizations responsible for developing networking software and

- organizations responsible for manufacturing networking hardware

toward embracing Open System Interconnection (OSI) architecture.

This may provide promising platform to achieve global network management, but till then, we have to establish infrastructure to detect or alleviate version mismatch problem in existing networks. It will be quite a while before the existing networks provide OSI compatible services over OSI-compatible protocols.

In the meantime, there is a need to upgrade and modify existing NM tools so that they can be used to detect the VM problems. Universal OSI compatibility is a long way from being available and, until then, managers has to manage different versions of software.

4.1.4.1 Selection of network management tools

Specific tools are always required to manage and support specific network architectures and standards. The task of selecting proper tools for detecting version mismatches is as complicated as the issue of network management itself.

Example One

The complete management of modems connected with a DCE will require DCE's software and hardware to perform battery of tests such as:

- local modem loop back test,
- remote modem loop back test,
- local modem analog loop back test,
- local modem digital loop back test.

Example Two: DECnet's ALL-IN-ONE software will only be used to perform network management to DEC's own architecture.

At this point, we can summarize common problems related to NM tools.

- Various vendor-supplied products do not interoperate because of vendor's strategic business decisions.
- Improved versions of the network-management tools are also coming at much faster rate.

Then, the network manager's task include the management of an environment full of versions of applications by employing different and unrelated versions of NM tools.

4.2 Inter-framework problems related to network management systems and architecture

In this chapter, it is important to distinguish between two interrelated topics known as:

- interoperability associated with networks (LANs, WANs) and,
- interoperability associated with network management architectures (OSI, SNA, FORUM).

Lack of interoperability between networks containing multiple versions of software, even when based on the same standard, is an extremely common and well recognized undesirable phenomenon[CRAI88].

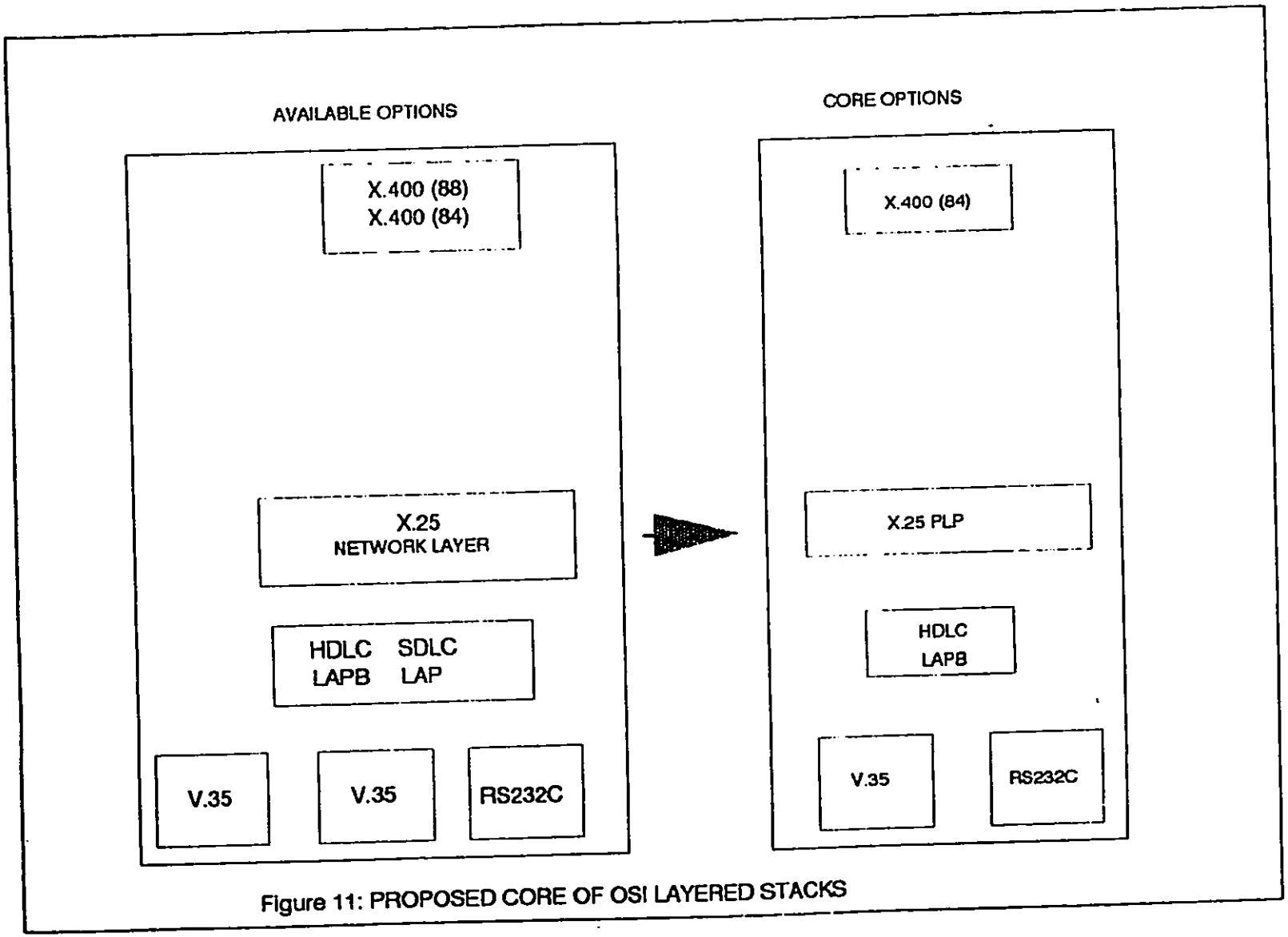


Figure 11: PROPOSED CORE OF OSI LAYERED STACKS

Lack of interoperability between network management systems can completely breakdown network management process. Such a breakdown will make it difficult for managers to detect, isolate or correct problems in the network. The involved networks may not be sharing crucial network management related information. The underlying problem may be due to the fact that two remote management systems are not intercperating consistently. Under such circumstances, problems directly linked with two versions of software may pass undetected and unresolved.

management standards with the focus of managing particular kinds of networks. ISO is defining how to manage OSI networks, and CCITT is defining how to manage telecommunications networks. Forum is trying to apply those management standards to manage all classifications of network [EmMa90].

4.2.1 Network management architecture incompatibility

Both the governments and the private sector recognize the need to develop an architecture based on universally acceptable architecture such as ISO's seven layer model. This architecture is universally acceptable but far less to be considered as universally available. Vendor-specific implementations of data communication protocols led to isolated domains of information, very expensive and difficult to bridge. Recent advances in communication technology based on the OSI model offer alternatives to vendor-specific network solutions[GOSI88]. GOSIP[GOSI88] specifies a set of OSI protocols for the use of government agencies(**Figure 11**)(Proposed Core Architecture for Layered Stacks).

There is one more dimension to version mismatch problem. It is becoming apparent in currently available standards and non-standard network management architectures. Many vendor-specific Network Management Systems (NMSs) exist, but there is very little

correlation and integration between these [CARU90]. The development of NM related standards, architectures and protocols rarely include solutions to version mismatch problems. Many major organizations are suggesting network management architectures and associated services without considering solutions in their proposed network management architectures, protocols or services to provide operational management to multiple version environment.

For comparison, consider two network management architectures to outline the differences in their suggested NM services. IBM's Open Network Management (ONM) architecture divide network management services into four categories.

In the next section, we review the progress and problems encountered of three major thrusts to avoid interoperability problem between network management systems. There are many organizations dedicating their efforts and time to find solutions for both types of interoperability related problems. For example, **Forum** is an organization tasked to establish network management architecture capable managing a full range of networks including LANs, WANs and international public and private packet switching networks.

The OSI Network Management Architecture has divided services into five categories. **[FIGURE12]** (Management Services Provided by SNA and OSI). This kind of inconsistency and incompatibility in NM services cause mismatch problems.
IBM's and OSI's network management architectures [FERN89] are very similar in terms of layering but their function and service level design and description is quite different.

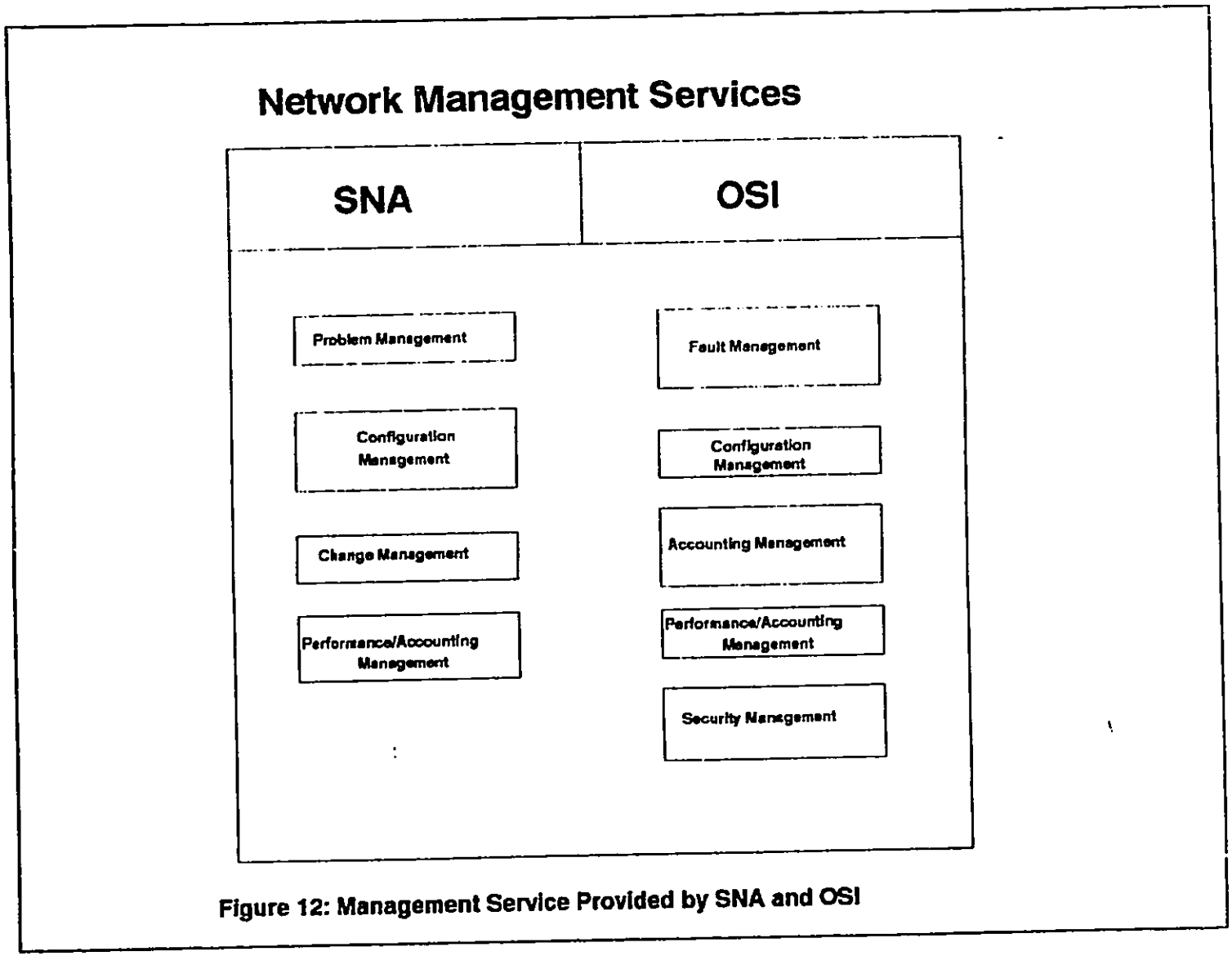


Figure 12: Management Service Provided by SNA and OSI

4.2.1.1 Application incompatibility management

CCITT's X.400 standard is extensively used as a electronic message handling system, as discussed in Chapter 2. It has been implemented by many organizations successfully [ENVOY100].

Application software versions may have many common characteristics and many operational differences:

1. Application system may be implemented on a medium-size machine or main frame.

2. The programming language can be one of many high level languages mixed with specific areas programmed in an Assembler Language.

3. Message Handling System may support a selective group of messaging formats such as: simple ASCII, telex, FAX and/or graphics.

4. It may offer all or a selective set of basic service elements as defined in the standard.

5. It may offer all or a selective set of optional service elements as defined in the standard.

6. It may support one or more communication protocol such as: X.25, TCP/IP.

Interoperability between such distinct, independently operated and physically separated applications require an arrangement between cooperating systems to achieve function by function controlled interoprability testing. Both parties must provide implementation specification documents, outlining the descriptive list of implemented functions and services and also list of function or services which are left unimplemented.

An other problem with test tools is that they are rarely compatible with each other.

4.3 An extension to current OSI integrated network management approaches to cope with version mismatch

The development of network management standards constitutes the standardization of network management topologies, protocols, management information databases, management-related report formats and management-related information interpretation logic. Number of groups are working on these areas. But what is missing in all these oSI efforts is any considerable discussions about versions or releases of such entities. In [STALL93], stalling points out that part of SNMPV2, perticularly capability definitions and conformance statement do treat version mismatch at software release level.

The idea of OSI management standards is to maintain and control interoperability between nodes designed based on OSI standards but not versions of standards. An extensive amount of work had been published and is still in progress inside various groups involved in the International Organization for Standardization (ISO) and also in International Consultative Committee for Telephone and Telegraph (CCITT) [SHAW89, FORU88, CARU90, JOMU90].

Network management standards usually attempt to satisfy requirements or demands put forward by a specific vendor, a vendor's product, a vendor's network, a vendor's topology, or a vendor's communication protocol. Standards should not only cover the requirements of vendors but also end-users as well as network managers. Standards should be designed to be flexible enough to withstand their own regular evolution.

NM services and architecture assume a specific architecture which makes specific network management useful. OSI defines network management guidelines, management protocols, and management tools for networks compatible to the OSI layered architecture.

Information networks are extremely complex, involving different software modules and different vendor supplied hardware components. Hardware components may include modems, multiplexers, front-end processors, PCs, workstations, and mainframes and network topologies such as Local Area Networks, Wide Area networks, Digital Network Architecture, Systems Network Architecture, Novell, and, sometimes, microwave networks, satellite networks, mobile data communication networks, packet switching networks, and many customized topologies. This complex environment uses many different type of operating systems, data communication protocols, database management systems and innumerable application programs. Also, all major network architectures uses layered structures like OSI's seven-layer model. There are large numbers of software and hardware versions which became crucial parts of networks during their continuous growth over a number of years, if not decades.

Other factors partially considered in NM models are:

- Management of network growth.
- Network adjustment due to new versions of standards.
- Administration of the multi-version networks.

4.3.1 Requirement For additional version mismatch information

The first objective of a management architecture is to provide a framework to support the interoperability of multiple version software systems. In such management systems, it is necessary to identify what data the management systems need to exchange and what data will not be available for exchange because of multiple versions.

A successful exchange of management related information requires that all communicating parties agree to follow strict rules based on standard protocols.

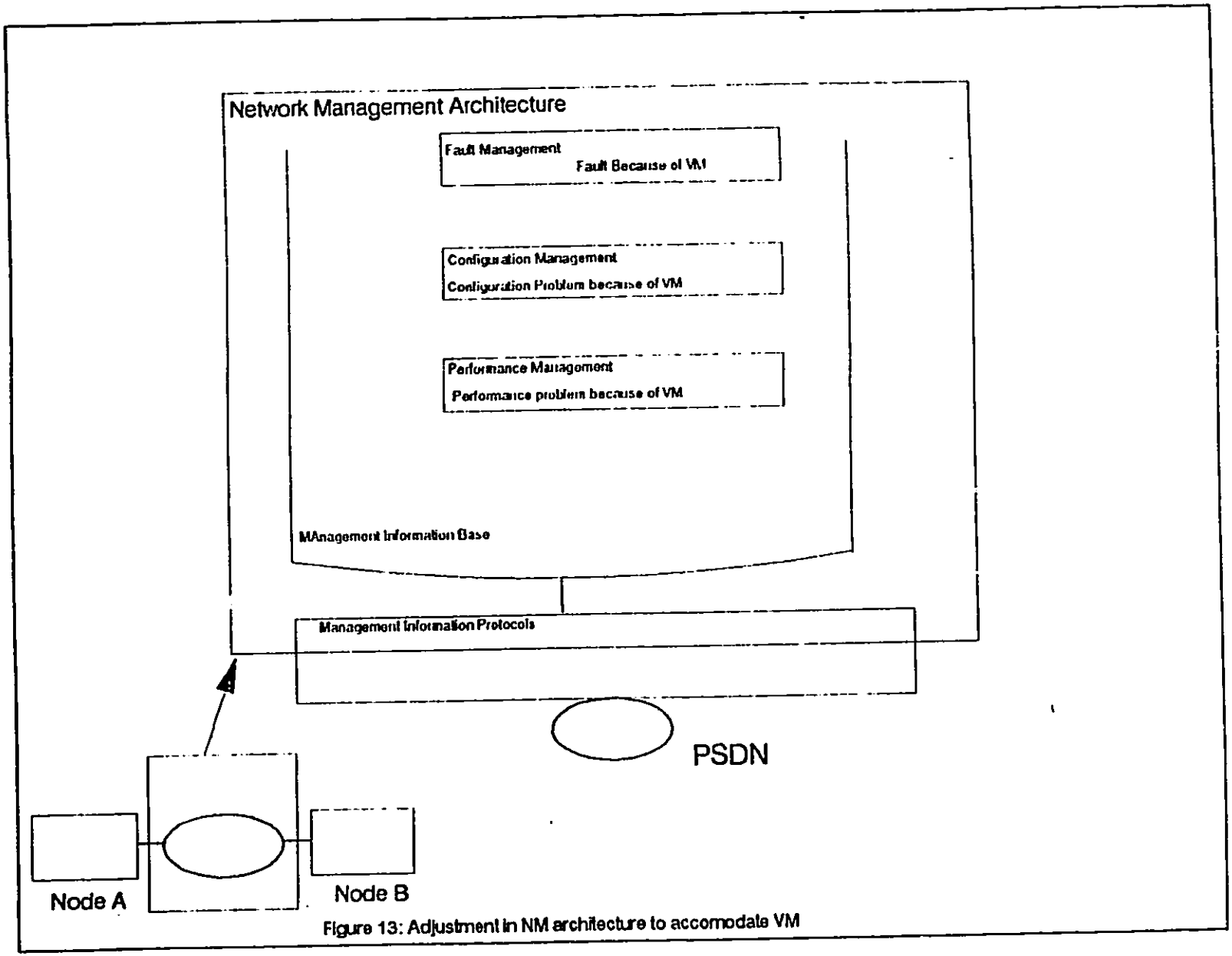


Figure 13: Adjustment in NM architecture to accomodate VM

H. Minhas

The second objective is to be sure that the architecture is flexible and broad enough to perform the management of a full spectrum of networks, regardless of the installed software versions.

The main modules in an architecture are the interface to the users, network management applications, information management database, network management protocols, and underlying protocol stacks.

A management information database (MIB) as defined in OSI network management architecture should be required to handle information related to not only the network management processes but also version related information **[Figure 13]**.

Queries may be performed on the data base to provide answers about the status of the network in real-time. Information can be processed off-line and reports can be generated. Another feature of this information base should be to store information based on the application and protocol versions under operation.

Another important module is the management protocol which will be used explicitly to move network management oriented information to and from nodes to the management node and then management oriented information will be extracted out of the packets and transferred to the information base for on-line access or for further analysis.

4.3.2 Extensions to integrated network management architectures to cope with VM

The task of managing networks becomes extremely difficult when it includes the management functions involving multi-version software. Most of the network management related specifications provide functions to analyze the management related information (MIB's)to manage cooperating network resources but falls short in providing services for the management networks (which have many versions of applications, databases, operating systems installed on many computers).

Many networks have developed their own guidelines to provide short-term solutions to the problem. In case of trouble, actions may include disconnecting a particular physical system from the network or a drastic measure such as re-installing a previous release across the network.

What is missing is a common strategy to improve understanding as well as to provide universally acceptable guidelines useful for network managers. The latest version of SNMP addresses several of these concerns by including release information and other key attributes of the software version in the management information base. This is not yet done in the OSI world. However, the new release of SNMPV2 has not been universally adopted in spite of these enhancements.

The following guidelines are written for OSI network managers, and span the activities that should be undertaken by respective groups in each of the three phases of the life cycles, starting from software standards development to the successful installation and use of the final implementation in the network. See also [STAL93] for SNMP/TCP/IP related work.

4.3.3 Guidelines for managing(detecting or resolving) VM in these new VM-sensitive network management frameworks

Standard development process:

1. The categorization of functions, services and options should be undertaken by using a newly modified approach as explained in Chapter 3 in this proposed standard development process.

2. The core of the Protocol, when the standard is related to communication protocols, should be defined as clearly, concisely and accurately.

Implementation process

1. An implementor must provide a list of functions implemented in the product.

2. An implementor must provide a list of functions not implemented in the product.

3. An implementor must provide a list of functions partially implemented in the product.

4. He should further provide a list of functional differences which are not limited to availability/non-availability of complete functions but also the availability/non-availability of sections or sub-sections of functions.

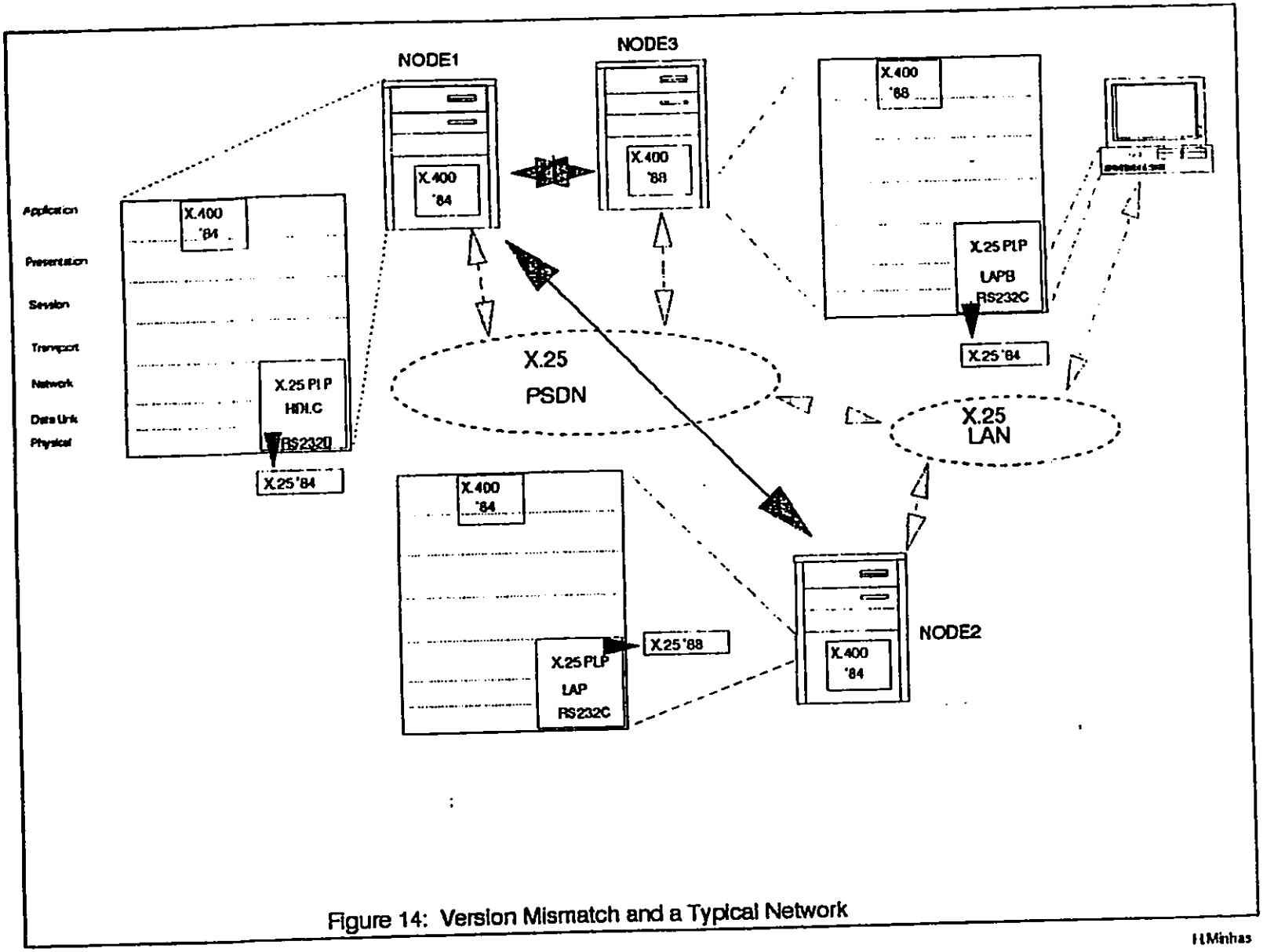


Figure 14: Version Mismatch and a Typical Network

HMinhas

Network management process:

1. Prior to the installation of any new version of a protocol, an attempt should be made to understand the functions which will limit communication. This information should be passed down to the service subscribers to avoid a situation where subscriber may want to access a service which may be available but cannot be accessed through the protocol stack used by the subscriber.

This situation is depicted in **Figure 14.** The broken line shows that communication across two nodes for some services is impossible while the unbroken line shows that the communication for all the services is possible.

2. The availability of the complete description of network architecture is required by the network manager and software developers specifically related to network communication.

Consider a VM network as shown in **Figure 2.** The network with three nodes. These nodes are installed with different versions of software and connected to a same backbone network. The possibility of the existence of such a network is real and communication between them is common practice.

3. Management architectures, management information communication protocols, management information database and objects to be managed are being individually defined and developed at this moment by many large vendors for their next generation of network management products. Similar types of work are undertaken by many standard development committees and organizations such as CCITT and

ISO. Unfortunately, the existing standard development process does not consider the software version mismatch issue.

4. OSI's network management architecture has proposed the rules to implement Management Information Base, the database which will receive, analyze, and store network management oriented information which can be accessed remotely, given that the OSI network management protocols are used. During the management of a version mismatch network, information stored in the MIB NM modules may be wrong because a trouble report had been generated. The real reason for flagged problems was version mismatch. With this in mind, it is suggested that MIB should be designed to consider such cases.

5. Controlled testing should be performed in collaboration with implementation groups between as many different versions as possible. This testing should include functions which have been flagged by software designers as expected.

6. Network managers should produce a cross-connectivity matrix by which they provide a complete list of implementation expected to connect with it.

The version mismatch problem is still considered secondary compared to other high-priority issues. Universally acceptable standards are not available at present in the area of network management. Almost all efforts are currently devoted to the development of NM standards.

However, if the standards developed by international bodies include the VM problem in their initial draft

standards, there will be a significant savings in terms of time and money usually spent to perform NM functions. This saving will more than compensate for the corresponding slowdown in OSI standards development.

4.3.3.1 Mapping between stacks

When a NM architecture proposes that management software should be distributed across nodes, then the logical mapping between one protocol stack to another stack is the only way to achieve communication.

Any vendor who has a wide range of networks and applications installed and also supplies NM oriented products must find ways to communicate international NM standards. This communication is not limited to the exchange of reliable and accurate information but must include procedures to decode and interpret management information for local use as well as to provide mechanisms to transfer NM-oriented information to remote management oriented nodes. This can only be achieved by providing function-by-function and services-by-service mapping between protocol stacks.

4.3.3.2 Mapping between layers

To manage networks successfully, some type of mapping between layers should be developed at the time the layer is being implemented by acquiring information or all the layered products which may communicate with the layer under design. The expected interoperability issues should be addressed to limit the amount of surprise the network managers undergo.

4.4 Backward Compatibility/ Forward Compatibility

Numerous implementations of protocols standards and applications software are available. Networks install versions without providing in-time notification to other networks. Notification to all networks is extremely difficult because of their sheer number. The interoperability between different versions is always a real possibility. As a result, network managers must prepare themselves to encounter interoperability and incompatibility issues which may arise due to version mismatch after the installation of their own new release. The manageability of networks will depend on the compatibility between versions.

Backward compatibility

Backward Compatibility of software can be defined in terms of execution of the a new version or a release with the preceding version of the software. In other words, the new release can support all the functions and features previously supported by the old release and it can do more.

Forward Compatibility

Forward Compatibility can be defined in terms of execution of an old version or a release of software with the succeeding version of the software. In other words, the old release can support all the functions and features in the new release. This means that new release and old release can communicate together without any incompatibility and without any change in user environment

The software version or release in discussion may be an application software such as a database application or another utility, a PC based software package, or a communication protocol software.

A backward compatible software may not be absolutely provide forward compatibility. Upper version is almost always backward compatible to the lower version but lower version are rarely forward compatible to the upper version..

One should make an effort to predict VM associated problems before one install software based on its backward compatibility or forward compatibility claim. This can be achieved through rigorous function-by-function testing. If there is incompatibility between two functions, one should differentiate weather to ignore the incompatibility or correct it if considered as version mismatch problem before installation. The error-free testing between two versions does not insure that every function and service will be available by using any one versions.

4.5 Functional profiles

"Functional profile" is a relatively new and key concept in the modern communication networks in general and OSI network architecture in particular.

As the name implies, a functional profile is a route for deriving useful functional services by using available standard implementation/s from the stack of implementations installed on a hardware platform.

There may be many available paths to access an application placed on the top application layer through bottom supporting layers. "Functional profiles" has been defined to deal with the problem of providing a range of options to access an application from a large population of users using a controlled set of networking architectures in their local environment.

OSI architecture can be applied to any field of business. At each layer in the OSI stack, a number of independent softwares may be provided where each software provides a different way to access the same application. Standards are being developed by vendors, users, government IT groups and many others and it is the responsibility of central organizations to provide rules to support universal interconnectivity

On the other hand, functional profiles are being set into a number of industries, corporations and Government agencies. This process may produce many open system functional profiles. Functional profiles may include many versions of standard protocols. It has been noted that the main requirement of functional profile design is to get rid of unsuitable options. But the question remains, "what about all the existing applications, protocols and platforms?".

Development of standard profiles is another important area where the VM problem oriented options can be included. In other words, a profile can include X.25'84 as well as X.25'88 (In Appendix 2, historical evolution of X.25 specification standard has been outlined. It is clear that even the functional profile adjustment will make it difficult to deal with all possible versions related to the specification standard) which are two options for the user to access an application.

4.6 VM related topics:

ISO has recognized that if interoperability between computers and networks is considered, then there must be a universally acceptable standard which will be followed by an implementor. For the achievement of this aim, it is necessary to confirm whether the implementation is the exact derivative of a specification standard. The implementation needs to be tested function-by-function against the specification. This area of testing has been referred to as "conformance testing".

Conformance testing confirms or denies the implementation based on whether it meets the specification. Not a specification Sometimes, the implementations are tested for backward compatibility but are rarely tested against three or more previous version.

It is difficult to predict whether a software implementation version that has passed a standard conformance test suite conforms to versions derived from a specification standard where all the versions are sequentially derived. A release can conform to a version of specification standards but this does not guarantee that conforming implementation versions will also conform to each other.

Because of the limited scope of this thesis, our study will not deal with the question of how to modify conformance testing method to accommodate issues of VM.

Besides moving towards the goal of interoperability and integrated network management, the trend is towards automated network management. It would be worthwhile to incorporate version mismatch handling in these automated systems.

Chapter 5

Conclusions and Assessment of Proposed Process

5.1 Summary and Conclusions

The intention of this thesis was to investigate in detail the problem of software version mismatch in the OSI framework.

We have seen many types of version mismatch problem. Briefly, version mismatch between communication software can deteriorate node interoperability. Version mismatch between networking software can impair cross-network interoperability. Version mismatch between applications impact interoperability between applications as well as applications and its users.

Due to the complex nature of the VM problem, no short term OSI solution is foreseen. We suggest certain changes to the existing standard development process as well as to the network management process as a partial solution to this problem, but expect that these suggestions will not likely be implemented in the near future for reasons given in the following sections.

In addition, suggestions are made to standards development bodies to improve the existing process to deal with the version mismatch problem. We have pointed out in Chapter 4 new proposals by which network trouble-shooters can use information provided by software implementation teams and make recommendations to use that knowledge at a proper time to achieve successful network management.

Real-life examples were used (X.25 and X.400) to support the proposed new

approach used to categorize functions and parameters differently.

To understand the issue more clearly, we have seen many real-life network-related examples from CCITT Recommendations including X.3, X.28, X.29, X.25 and X.400. Examples have been selected from the area of communication protocols by using X.25 and area of application standards such as X.400 Message Handling System to understand the version mismatch problem more clearly

5.2 Advantages and disadvantages:

After detection, each network management related problem once detected needs fixing. This fixing process may involve rigorous testing to establish the exact nature of the problem. Early distinction whether the problem, in question, is generated due to software version mismatch can help network managers great deal. This piece of information will guide them to form proper problem resolution strategies. We have seen that version mismatch problem will require a completely different type of action.

Attacking VM issue within standard development process is a key to successful resolutions to the VM problem. This effort, in turn can save money, time, and energy spent by network managers.

The main disadvantage of considering VM within standards development is that it will slow down the standardization process which has already been condemned as a long and slow process. In many cases, it spread over many years. CCITT's X.25 protocol recommendation took twelve years and three main releases -X.25'76, 80 and 84- before it had stabilized but now ISSN's arrival will have a destabilizing influence. The proper response, of course, is to

recognize that the benefits of incorporating the suggestions into the standards development process will outweigh the cost of corresponding delays.

It would require highly skilled technical people to resolve VM oriented problems. They must understand both versions completely. This will enable them to pin-point exact location of software module in both the versions in which problem is occurring.

The VM issue had been rejected as not one of the high priority tasks for standard development groups.

For example, proposed OSI network management standards and architecture does not make an attempt to resolve faults during fault management procedures when considering two different protocol versions of communication protocols.

Another reason is that this problem seldom effects users because network managers tends to eliminate or restrict services which they perceive would become victims of software VM before making services commercially available to the public.

For example, X.28 contains many parameters which are useful only to configure user environment. On of such PAD parameter is Parameter 5 which deals with different options to receive CR and LF combinations from the PAD. Network managers will restrict options which they know will cause problems for the user environment. But these argument don't suggest that we should procrastinate the tackling of the VM problem.

5.3 Future Work:

In this thesis, we have not looked at other management aspects of VM networks like the management of the hardware versions associated with networks. Many other types of network such as mobile communication networks and satellite communication networks are not considered because the nature of version mismatch problems varies from architecture to architecture. The software standards used in such networks and the impact of version mismatch can also be studied.

Testing of protocols and applications also requires great deal of work and understanding to include version mismatch solutions. Test suites will have to include test cases based on interoperability between expected versions and expected architectures.

There are many groups involved (CCITT, ISO) in the process of defining network management standards. Instead of limiting this process to defining architectures and protocols for specific network architectures, it should attempt to manage networks which are filled with distinct versions of software and hardware. SNMP, version 2, does this to some degree - the best features of its VM-related information base are very similar to our proposals for OSI VM management; this provides additional support for the OSI-related proposals in this thesis.

Versions of operating systems and development languages also have considerable impact on the final released versions of communication software implementation which is also an area to be looked at.

REFERENCES

[BRAD88] Bradley B.," Digital Network Architecture and the OSI Model ", Data Communications, February 88.

[BELL87] Bellcore Communication Research," OSI Protocol Requirement and Objectives for Operations Systems and Network Element Interface ," Technical Advisory TA-TSY-000285, Issue 3, December 1987.

[BuPu89] "X.25: It's Came a Long Way", Computer Networks and ISDN Systems 16 (1988/89).

[CCIR86] " An Introduction to the Work of the CCIR and CCITT ", British Telecommunication Engineering, Vol. 5, Oct 1986.

[CCIT88] CCITT," Final Report to the Plenary Assembly - Part III.2: Draft Revised Recommendation X.25 ", Document AP IX-49-E, April 88.

[CRAI88] Craigie, J.," ISO 10021 - X.400(88): A Tutorial for Those Familiar with X.400(84)", Computer Networks and ISDN Systems 16 (1988/89) 153-160.

[CARU90] Caruso R.E.," Network Management: A tutorial Overview ", IEEE Communication Magazine, March 1990.

[DAVI86] Davidson P.J.," Review of the CCITT Recommendations for Integrated Services Digital Network ", British Telecommunications Engineering, Vol. 5, Oct. 1986.

[EmMa90] Embry J.,Manson P.,Milham D.," An Open Network Management Architecture: OSI/NM Forum Architecture and Concepts ", IEEE Network Magazine, July 1990.

[EWIC84] EWICS/TS, " Guideline for Verification and Verification of Safety Related Software - a report of a TC7 Systems reliability, safety and Security

Committee ", North-Holland, Computer Standards and Interfaces 3 (1984) 91-99

[FERN89] Fernandes J.," SNA and OSI: Which Manages Multivendor Network Best ", Data Communication/April 89.

[FORU90] "OSI/Network Management Forum-Architecture", Forum 004 issue 1, Jan 1990.

[FORU88] OSI Network Management Forum-Forum Interoperability Interface Protocols," Application Through Physical Layer Application ", Octt1988

[GOSI88] Government Open System Interconnection Profile," Category: Hardware and Software Standards; Subcategory: Computer Network Protocols ", FIPS PUB 146, 1988 August 24.

[HARR88] Harrop M., " User requirements for OSI ", North-Holland, Computer Standards and Interfaces, 5 (1986) 271-275

[ISO1] Information Processing Systems - Open Systems Interconnection - Basic Reference Model ", DP 7498 , (1983).

[ISO2] International Standards Organization, ISO 3309," Data communication. High-level Data Link Level Procedures Frame structure.

[ISO3] International Standards Organization, ISO 4335," Data communication. High-level Data Link Level Procedures - Elements of Procedures.

[ISO887] ISO 8878, Use of the X.25 Packet Level Protocol in Local Area Networks, Nov. 88

[ISO888] ISO 8881, Use of the X.25 to provide OSI Mode Connection-Mode network services, Sept 87.

[JOHN88] McConnel J.," Stepping off the OSI Standards bandwagon ", Data Communications, June 88.

[JoMu90] Joseph C.A., Muralidhar K.H.," Integrated Network Management in an Enterprise Environment ", IEEE Network Magazine, July 1990.

[KaBa88] Kairi K., Barnard D.T.," Design and Implementation of an X.400 Stand-Alone User Agent", North-Holland,Computer Standards and Interfaces 7 (1988) 219-232.

[KELL87] Kelly W.J.,"Network Management and the OSI Model," Data Communications/December 1987.

[KRAE89] Kraemer T.F.,"Product Development Using Object Oriented Software Technology", Hewlett-Packard Journal, August 1989.

[LLOY86] Lloyd R.V.S., "Putting Standards to Work", North-Holland,Computer Standards and Interfaces 5 (1986) 9-12.

[MEEK88] Meek B.L.,"Is Standardization just regularization?", North-Holland, Computer Standards and Interfaces 7 (19988) 257259.

[MELE86] Meledez W.A.,"The Upper Layers of the ISO/OSI Reference Model (Part I)," North-Holland, Computer Standards and Interfaces 5 (1986) 13-46.

[MOUD86] Moudiotis G.,"Development of CCITT Standards for Packet Switching Data Networks", British Telecommunication Engineering, Vol.5, Oct.1986.

[ROUT88] Routt, T.J.,"SNA network Management: What makes IBM's Netview tick", Data Communication / June 88.

[SHAH89] Shaw K.,"Managing Networks of the '90s",Data Communication 1989.

[SiDo88] Sinderen M.V., Dorregeest E.," A Critical Analysis of the X.400 Model of Message handling systems", North-Holland, Computer Standards and Interfaces 7 (1988) 363-375.

[SuZa85] Sullivan M.T., Zader R.T.,"The Role of Standards in Network Evolution", North-Holland, Computer Standards and Interfaces 4 (1985) 79-84.

[STAL93] Stallings,W.,"SNMP, SNMP v2,and CMIP, A PRACTICAL GUIDE TO NETWORK MANAGEMENT STANDARDS", 1993.

[TuRu88] Turman B., Rubin R.,"Bell Operating Company Packet Interfaces", North-Holland, Computer Networks and ISDN Systems, 16 (1988/89) 187-196.

[UNSO88] Unsoy M.,"How Packet-mode transmission services will evolve in ISDN", Data Commun./April 1988.

[WARD89] Ward D.S.," Using Open System Interconnection Standards as the Base for a Comprehensive Distributed Systems", North-Holland, Computer Standards & Interfaces 9 (1989) 105-112.

[WALL88] Walker R.,"Why Users Must Co-operate Internationally on Standardization", North-Holland, Computer Standards and Interfaces 7 (1988) 57-62.

APPENDIX 1

Pin	Circuit		Description
	EIA	CCITT	
1	AA	101	Protective Ground
7	AB	102	Signal Ground
2	BA	103	Transmit Data
3	BB	104	Receive Data
4	CA	105	Request to Send
5	CB	106	Clear to Send
6	CC	107	Data Set Ready
20	CD	108.2	Data Terminal Ready
22	CE	125	Ring Indicator
8	CF	105	Receive Line Signal Detector

25-pin connector to be used with EIA 232 C/D interface

Core bit rate: 2400, 4800, 9600 b/s

Core service option: synchronous, full-duplex, continous carrier

Core of Standard EIA 232 C/D (Physical Layer)

Figure 15

Appendix 2

First idea about X.25 specification was conceived about decade ago. Since many major and minor enhancements to this protocol specification have been made. X.25'88 has brought an important feature into the specification known as dial-in/dial-out facility.

Major changes made to X.25 over the Decade:

The most important additions were the following. In 1977, data link layer was expanded to include Link Access Procedures (LAPB). It corresponds to a subset of HDLC procedures that was approved by ISO and various technical advantages over LAP. LAP is no longer supported by many networks.

In 1980, major changes were made in the standard. It included standard packet size of 128 octets. In the specification, 24 optional user facilities were included of which six were tagged as essential.

As expected, in 1984, the technical accuracy and scope of the standard has been expanded to support OSI network services[ISO887]. In this specification, the number of optional user facilities reached 33 and out of which 9 were tagged as essential. In this release, Datagram service which was introduced in 1980 was dropped because major player had showed any interest to implement it.

X.25 and LAN:

The emergence of LAN technology had provided user an option of connecting data processing facilities under one roof to benefit from faster and cheaper interconnectivity. Then came a major desire to connect LAN's together. To meet that demand, ISO showed interest to support LAN inclusion in X.25 under the name of X.25 PLP [ISO888].

GLOSSARY

CCITT	The Consultative Committee on International Telephony and Telegraphy
Connection Oriented Service	A network service that establishes a complete path between end node before passing the first data packet.
Connectionless Service	A network service that routes data in individual packets over a path that is selected as the data arrives at each node.
Entity	Active element such as program segment and integrated circuit, that provides the services.
Function	Each process can be described as a function. Programming languages provide generalized tools that enable users to express the specific requirements of their applications.
Implementation Version	Any system which provides improvements and modifications over its predecessor system. The two systems products are different in terms of installation time and functional capabilities.
Layering	A widely accepted structure technique, and the one chosen by ISO is called layering. The communication functions are partitioned into vertical set of layers. each layer performs a related set of functions required to communicate with another layer.

Mandatory Functions / Services

Standards denote as mandatory capabilities which must be implemented before the software conforms to the specification.

Node

A computer that participates in passing data across a network.

Optional Functions/Services

Standards denote as optional capabilities which may or may not be implemented in a conforming system.

Open System

An open system is a system capable of communicating with other open systems by virtue of implementing common international standard protocols. However, an open system may not be accessible by all other open systems.

Subscriber Parameters

A subscriber may select his/her particular choice of setting for certain key system varible such as throughput class, baud rate (EXAMPLE 2 and EXAMPLE 3).

Specification Standard

A document produced by international organizations to provide guidelines to built software products.

Services

Services are the benefits provided by the software implementations.

Protocol

Set of rules governing the exchange of data between two entities.

Standard protocol

A protocol produced by a group of experts with the intention of promoting common rules of behaviour between communicating entities.