OCCASION

This publication has been made available to the public on the occasion of the 50th anniversary of the United Nations Industrial Development Organisation.

DISCLAIMER

FAIR USE POLICY

CONTACT

Please contact publications@unido.org for further information concerning UNIDO publications.

For more information about UNIDO, please visit us at www.unido.org

UNITED NATIONS INDUSTRIAL DEVELOPMENT ORGANIZATION
Vienna International Centre, P.O. Box 300, 1400 Vienna, Austria
Tel: (+43-1) 26026-0 · www.unido.org · unido@unido.org

20505

Distr.
LIMITED

IPCT.196(SPEC.)
4 January 1994

UNITED NATIONS
INDUSTRIAL DEVELOPMENT ORGANIZATION

ORIGINAL: ENGLISH

Handbook of Telecommunication Software*

Prepared by

Jan Vytopil**
Mathai Joseph***

---

* The views expressed in this document are those of the authors and do not necessarily reflect the views of the Secretariat of UNIDO. This document has not been edited.

** Professor of Computer Science, University of Nijmegen, the Netherlands.

*** Professor of Computer Science, University of Warwick, U.K.

V.94 20003

CONTENTS

Management Summary i

Preface v

1. Importance of software in telecommunication 1

2. What is telecommunication? 5

2.1. The primary telecommunication functions 5

2.2. Management functions 7

2.3. Implementation of the telecommunication functions 8

2.4. Overview 19

3. Main types of telecommunications software 20

3.1. Often used software-related terms 20

3.2. Operation software 23

3.2.1. Operating system 24

3.2.1.1. Operating system kernel 26

3.2.1.2. Additional services 27

3.2.2. Application software 31

3.2.2.1. Telephonic support 36

3.2.2.2. Telephony application 39

3.3. Network planning systems 47

3.4. Maintenance systems 50

3.5. Billing and accounting packages 53

4. What is so special about TCS? 56

4.1. Introduction 56

4.2. Size and complexity of the software 58

4.2.1. The consistency of system and software architecture. 59

4.2.2. Organisation of the development activities 60

4.2.3. The software techniques and tools employed. 62

4.4. Long working life required 65

4.5. Real-time operation 66

4.6. Stringent reliability and availability requirements 67

5. Prerequisites for entering the market 69
6. Annex 78
CCITT Communications Standards 78
Glossary 80
Index 84

List of figures

Figure 1: The primary telecommunication functions 6
Figure 2: Network management functions 8
Figure 3: Telephone exchange functions 14
Figure 4: Communication functions 17
Figure 5: Network and element management 18
Figure 6: Operation software 24
Figure 7: Operating system 2[illegible]
Figure 8: Call handling structure 34
Figure 9: Call handling functional units 34
Figure 10: Subscriber line interface unit 36
Figure 11: Telephonic support layer 37
Figure 12: Phamos architecture 52
Figure 13: Structure of development organisation 60

MANAGEMENT SUMMARY

Telecommunication software, and in particular the software to control a telephone exchange, has a number of characteristics that makes its development complex and costly

Such software is notable for

- **its size and complexity --**
  The complexity of the software is a reflection of the complexity of the functions of a telephone exchange, and the development of such software is a challenging task which needs particular attention. To keep the development manageable, the choices relating to (1) the system and software architecture, (2) the organisation of the development activities, and (3) the software techniques and tools employed must be exercised taking into account their effect on the overall quality and the outcome of the development effort .

- **the long working life required --**
  A typical switching system will be in use for at least 20 years and the basic components of the telecommunication software must be designed so that their use can be adapted and extended over this long period. One of the main design objectives for modern switching systems is to make the software 'future proof'. For this, the software must be as independent as possible of both the architecture and the hardware of the switching system, so that it can accommodate additions and changes to the hardware with little or no modification.

- **the need for real-time operation --**
  The software has to deal with a large number of simultaneous activities, most of which require guaranteed response times. It has to interact intensively with dedicated hardware, and a high degree of parallelism through multiprogramming and multiprocessing must be employed. The software must be predictable in its temporal behaviour, and this requires a certain simplicity and perspicuity of structure. It must also be performance sensitive, so the executable code must be efficient .

- **the stringent reliability and availability required --**
  Most conventional systems may be stopped at certain times without serious consequences. For a telephone exchange the requirements are totally different: failure of a telephone exchange for a period of even 15 minutes is so rare that it would be reported in the newspapers. To satisfy these rigorous requirements, a whole series of design features are necessary: (1) hardware and software units must be replicated and guarded so that faults can be detected and a faulty unit repaired without interruption of the normal service, (2) the software must be capable of detecting and overcoming any adverse condition, and (3) the maintenance procedures must be defined so that they can be carried out during normal service.

Based on this analysis the handbook identifies the following **prerequisites for entering the telecommunication market:**

- **A creative environment** for skilled and motivated university-educated personnel, providing contact between industry and university, and able to draw on a strong technological base.

- The presence of **research and development centres** where feasibility studies can be carried out, basic designs proposed and evaluated, and new technologies developed.

- **Standardisation of development methods**, concentrating on CCITT standards, using SDL, CHILL (or C) and MML as the basis for the development of tools for specification, with implementation on industry standards development platforms.

- **Conformance to international switching standards** to be able to obtain components and know-how at competitive rates and to take advantage of world-wide design experience.

- **conformance testing facilities**, for hardware and software, to enable shared and local development of software components.

- An **integrated framework for systematic development**, starting with subsystems and smaller exchanges and including design, manufacture and system support. There must be long-term political support and the activity must be the responsibility of a high-level government policy-making body.

- **Long term finances** to help a new supplier to provide a wide range of support over the long product lifecycle . A realistic financial plan would be to expect a return on the investment within 15-20 years, while building up sufficient reserves to support new product development. Financial and organisational support is required for participation in international standardisation activities, and for the later dissemination of the knowledge.

- **Market regulation** is almost essential unless a relatively high level of expenditure can be sustained for a long period.

- A wide range of educational material must be introduced into conventional tertiary education courses in telecommunication, computer science and engineering. This must be complemented by more specialised short and long term courses targeted at new entrants, and for retraining those already in the industry.

PREFACE

Over the years, software costs for telecommunication systems have steadily increased in contrast to the sharp downward curve of hardware costs: in this, telecommunication software is following the trend of every information technology-intensive application. Telecommunication software is complex and highly intricate and the effects of software defects can be as damaging as hardware breakdowns and yet are harder to detect. Program design errors are often manifested only by remote secondary effects that are almost unforeseeable at the design stage. So the production of reliable telecommunication software is particularly costly and time consuming.

Maintenance and updating of telecommunication software through the life-time of the system is also a hard and exacting task and therefore no less costly. The extent of the problem becomes compounded when a telecommunication system is designed for use in many countries, each with its own network characteristics and constraints.

To the developer of telecommunication equipment, the specification, implementation and testing of software now account for by far the largest slice of the development budget. The development of a family of systems, intended to cover an entire range of exchanges from low capacity local exchanges to large trunk transit exchanges, would require the production of telecommunication software in terms of thousands of software programs, with continuing expenditure of effort for at least two further system generations.

Consequently, telecommunication software development is a major source of concern to telecommunication industry management. But why is this so? What is so special about telecommunication software? And, what implications do these

factors have for countries intending to enter the telecommunication market?

In this handbook , we will examine these issues.

The handbook is structured as follows: Chapter 1 provides an introduction to the importance of software in telecommunication. In Chapter 2, the basic functionality of a telecommunication system is defined and the implementation of these functions is discussed. The place and role of software in the implementation of the basic functions is briefly identified. In Chapter 3, the specific properties of telecommunication software are discussed in detail. In Chapter 4, conclusions are formulated about the factors that make telecommunication software so difficult to develop and maintain. Chapter 5 discusses the prerequisites for entering the telecommunication market. The Annex contains a brief overview of CCITT telecommunication standards.

This report is intended as a handbook for decision makers familiar with general telecommunication and informatics issues but with no specific expertise in telecommunication software. Our intention is to provide a concise document that enables one to understand the basics of telecommunication software development. While it contains a certain amount of necessary technical information, the report is also intended to be comprehensible to non-technical decision makers.

1. Importance of software in telecommunication

On February 14, 1876, Alexander Graham Bell applied for a patent for a telephone. It was granted on March 7, 1876, and three days later the first telephone conversation took place. New Haven, Connecticut, was in 1878 the first city to have a commercial telephone exchange, and this served 21 subscribers. The monthly rental of a telephone at that time was approximately $38 and the average income about $20 per month.

Telecommunication thus has a tradition of more than a hundred years and starting from the early days when it was limited in use to a few fortunate places it has now spread to every country in the world. Some of this spread is relatively recent: from the 1960's telecommunication has changed more than in preceding 70 years.

For example, telephone networks have expanded considerably

- in the number of subscribers, and
- in range (almost every country has an automated long distance service)

and making international and intercontinental calls is common. The structure of telecommunication services has also changed considerably and the design of contemporary telecommunication equipment bears little resemblance to that of its predecessors. Distance and time are no longer constraints and the provision of efficient telecommunication services has made it possible to use global time differences to advantage, rather than be constrained by them. If these changes are to be attributed to one factor, then it must be to the introduction of automated services made possible using *stored program control* of exchanges.

The first exchanges were manually operated: the caller informed the exchange operator of the desired connection and the operator then manually inserted the calling plug to the jack of the far-end subscriber and rang the bell of the telephone being called. Once the call has been completed, the operator disconnected the plugs and noted the call and its duration. The operator managed the use of the exchange and on request could provide many additional services. But, clearly, human operation of the exchange was slow and error-prone and the number of calls that could be handled was limited. However, the human intervention of the operator made it possible to provide very sophisticated services that the present day exchange cannot provide automatically.

It did not take long for attempts to begin to automate the operation and maintenance of the exchange. In 1889, Almond B. Strowger invented the automatic, electro-mechanical line selector which led to the automatic telephone system. Even with this level of automation, in the early systems dialling was not as simple as it is today: for example, the subscriber had to press a button on the telephone nine times to dial the digit '9'. Each time the button was pressed, a pulse of electric current was sent to the central station, where relays operated the electromagnets that selected the desired line. But since that time, more and more of the facilities of the exchange have been automated and today manual assistance is needed only for exceptional services.

There have now been three generations of exchanges, with differences in both connection and accounting services. The first exchanges required manual connection and manual call accounting and the 'exchange' was really just a switchboard: long distance connections required communication between operators to establish the call. This was followed by the electromechanical exchanges using delay or cross-bar switching techniques for automated connection between caller and receiver. Call accounting was still simple and call duration was measured in terms of meter 'ticks' but long-distance dialling was possible using special code prefixes. In he 1960's, the first computer controlled exchanges (the so-called 'stored program control' (SPC) exchanges) went into wide-scale use and this transformed the quality and extent of telecommunication services. In addition to the services previously provided, calls could be routed to balance traffic on lines and to improve the availability/utilisation of the network, automated support for fault diagnosis and maintenance was provided and itemised call accounting was possible. SPC exchanges could routinely handle large networks of subscribers (5000-50,000 lines), thus providing many new facilities but with greatly enhanced economies of scale.

The enormous possibilities opened up by the use of SPC techniques made it relatively easy to underestimate the difficulties that could be encountered. In his history of the telephone, Brooks[1] notes "Of all the new wonders of telephone technology - the one that gave Bell Labs the most trouble, and unexpectedly became the greatest development effort in Bell System's history, was the perfection of an electronic switching system, or ESS. In the early 1950s, a Labs team began serious work on electronic switching. Kappel, in his first Annual Report as AT&T president, in the year 1956, wrote confidently, 'At Bell Labs, development of the new electronic switching system is going full speed ahead. ... The first service trial will start in Morris, Il, in 1959'; ... Shortly thereafter, Kappel said that the cost of the whole project would probably be $45 million

---

[1] J. Brooks: 'Telephone. The First Hundred Years', Harper, and Row, New York, London, 1975, pp. 278-279.

... Kappel's tone on the subject in the 1964 annual report was, for him, almost apologetic ... Another year and millions of dollars later, on May, 30, 1965, the first commercial central office was put into service... After 1965, ESS was on its way ... But the development program, when the final figures were added up, was found to have required a staggering four thousand man-years of work at Bell Labs and to have cost not $45 million but $500 million." In retrospect, this was perhaps not surprising given that this was the first such system and that it had 1.1M lines of code.

But ten years later, the complexities of software development for SPC exchanges were still being underestimated. In 1978, a press report[2] announced: 'Software is at the heart of the System X development. It will account for between 30 and 40% of the entire cost of an (System X) exchange'. But by 1985, when System X exchanges were entering the UK national switching network, many manufacturers estimated that software accounted for 75% of the cost of developing a system

The adaptation of a switching system to enable it to provide all the facilities and features required in the telephone service of a country is also a costly process. In the case of ITT's System 12 adaptation to the North-American market made it necessary to incorporate more than one thousand specific telephone features and service requirements. In early 1986, after spending about $200 million, a decision was made to cease further development of System 12 for the US market. System 12 is considered to have been one of the most expensive switching system developments ever undertaken. According to press reports[3] , by March 1985 over $1.1. billion had been spent on the development of System 12.

Since that time, new SPC systems have been designed and put in service in several countries and in the US AT&T has advanced from ESS1 to ESS5. In the UK, GEC Plessey Telecom's System X exchanges have replaced most of the earlier generation exchanges in the country and a similar situation holds in Sweden with Ericsson systems. The French telecommunications systems have a slightly different orientation and are still dominated by systems supplied by CIT Alcatel. Each development has required massive investment, often with direct or indirect government support, and telecommunications system manufacturers have been actively seeking overseas markets in order to recoup their investments.

---

[2] Financial Times [J. Poole], 7 May 1978

[3] Wall Street Journal, December 19, 1985, p.1

A number of developing countries have embarked on SPC system development either on their own or in collaboration with foreign partners: examples are Brazil and India. The Indian case is particularly interesting as the country took a three-pronged approach to introducing digital switching technology. First, a number of systems were imported and installed in major cities as local and trunk exchanges for telephone and telex switching. In parallel with this, a large fabrication facility was set up to manufacture exchanges based on foreign design. And finally, a well-endowed development programme was established to design large exchanges for future use. All three parts of this strategy have contributed to the vast improvement of the telecommunications infrastructure of the country and are now competing for future markets in India.

It is difficult to overestimate the importance of software in telecommunication systems. In some ways, a modern telephone exchange consists mainly a computer of modest size and a considerable amount of software and it directly alters very little of the rest of the telecommunication hardware (e.g. subscriber lines and line termination units, trunk lines etc.) to which it is connected. However, this is a very limited view as it is the software that provides the functionality to transform the whole network into a modern telecommunication system.

A rough measure of the importance of the software can be had from the following figures: the cost of developing 1 M lines of code, with a productivity of 3 lines/day/person is 300 000 man-days, or 1500 man-years. But many large modern exchanges have upto 5 M lines of code, representing on these terms around 7500 man-years of effort. Of course, not all of that code is new for each system and with the availability of development know-how it is possible for the code for a new system to make use of code developed for earlier systems; but the initial investment remains high and a relatively large on-going effort is required to keep together software technologists and design teams.

2. What is telecommunication?

As a concept, telecommunication is deceptively simple: it is primarily a process that permits the transmission of information from a sender to one or more receivers in any usable form, such as visible or audible signals, by means of any electromagnetic system.

To provide a reliable, economical and practical telecommunication service the telecommunication system must take account of the special requirements of different modes of communication (such as speech, video or data communication), the typical distance between the sender and receiver and specific properties of the part of electromagnetic spectrum being employed. As a result, different telecommunication systems exist, each optimised to a particular kind of communication mode, distance and signal carrier.

In this report we concentrate on conventional public telephony systems, i.e. communication systems, that carry speech signals via (mainly) cables over wide area distances. At the level of abstraction at which the communication software will be discussed, the differences between private and public telephone exchanges are not so significant but, for example, the specific features of ISDN-exchanges (such as packet switched data communication) and mobile communication systems will not be discussed.

This definition of telecommunication covers only the primary activity, the actual process of communicating. However, to provide telecommunication services to a large number of users in an economical way, the related and supporting activities, such as maintenance, administration of the use of the system (billing and accounting) and planning of the facilities are usually also supported by computer-based systems. Software for these systems - network management systems - is becoming increasingly important and will be discussed in this report.

2.1. The primary telecommunication functions

Information for telecommunication must be (1) transformed into an electrical signal (conversion), (2) accompanied by an identification of the destination (signalling), (3) provided with a communication path from sender to receiver (switching), (4) delivered to the receiver (transmission) and, finally, (5) transformed by the receiver into audible or visible information (conversion). Specifically,

- *signal conversion:* includes all the functions necessary to convert audible speech (telephony) or images (facsimile) into an analogue or digital (electrical) signal, and the receiving signal back into audible speech or an image. This conversion must be performed in such a way that (1) as little as possible of the essential information is lost, and (2) the cost is affordable to a very large number of users. These two opposing constraints are resolved by making a compromise on the portion of the bandwidth of the audible signal, or spatial resolution (in the case of facsimile), that is actually subject to conversion.
- *user interface:* enables the user to interact with the communication system to request services and to get information about the outcome of a request.
- *signalling:* translates user service requests into electronic messages that are forwarded to the rest of the system for processing. The complexity of the signalling function is related to the number of services available to the user. In conventional telephony, signalling between a user and the exchange is quite simple, but in digital telephony, which offers more services, the signalling function can be quite complex.
- *switching:* the function that allows a temporary connection to be made between two (or more) telephone sets. This function can be considered to be the telephony function. It provides an economical way to deliver telephony services by sharing conversion and transmission resources among users.
- *transmission:* provides the transport to the end-user and signalling information from one location to another. During the transport, adverse conditions in the environment can cause damage or loss of information; these effects can be reduced by employing redundancy in information transmission.

These functional areas are related and influence each other:

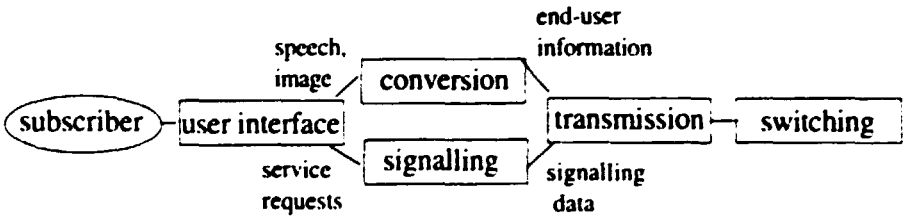

Figure 1: The primary telecommunication functions

2.2. Management functions

Management of communication networks involves deploying resources and co-ordinating their use in order to plan, design, install, operate, expand, analyse and administer communication networks, making it possible to provide the primary telecommunication services to the users. The following management functional areas can be defined:

- *Accounting Management* enables the system administrator to determine the usage of the system and record the associated costs. Accounting data is collected in the system in several layers and is processed so that, in combination with charging data (time zones, units of charge, etc.), it will result in a cost report.
- *Fault Management* is the set of facilities needed to detect and identify faults, to exercise diagnostic tests in order to obtain a better understanding of the nature of the error and to take corrective action where possible. These activities ensure high availability of the communication system by quickly recognising problems and performance degradation, and by initiating controlling functions when necessary, including diagnosis, repair, test, recovery, work around and backup.
- *Performance Management* encompasses all the functions required to evaluate the behaviour of the system and its effectiveness. Statistical data are gathered in order to present the system manager with the relevant information base. These data can be accumulated to give reports on a daily, weekly or yearly basis.
- *Configuration Management* facilities enable the network operating staff to read and modify the configuration data of a system such as addresses, routing information, the relation between hardware components, etc. Configuration management is considered the central process of network management. All the other activities, such as fault, performance, security and accounting management are supported by configuration details from configuration management. For each of the objects managed, the attributes, existence, state and relationship data are administered.
- *Security Management* provides an organisation's security administrator with the facilities to ensure that communication in the network is in conformance with the security policy. It is a set of functions to analyse and minimise risks, implement a security plan and monitor the success of the strategy. Special functions include surveillance of security indicators, partitioning, password administration and warning or alarm messages on violations.
- *Network design and capacity planning* is the process of optimizing the use of the network, based on data for network performance, traffic flow, resource utilisation, networking requirements, technological trade-offs, and estimated growth of present and future usage.

The functional areas are related and exchange information. They also exchange information with the network elements that are being managed (such as private telephone exchanges, telephone exchanges, etc.) and the network managers (network controllers, administrators and planners).

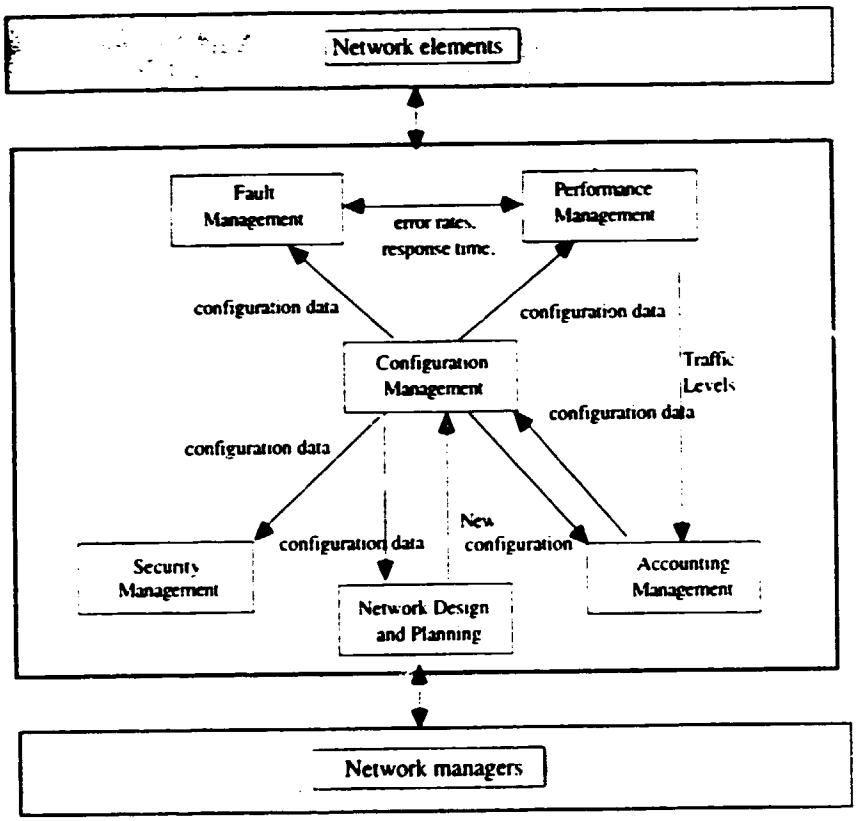

Figure 2: Network management functions

2.3. Implementation of the telecommunication functions

The primary telecommunication functions are performed by a network of telephone exchanges interconnected by trunks, and the user equipment (e.g. telephones) connected to an exchange by a so-called local loop.

Conversion

In a conventional analogue telephony, speech is converted (conversion function) in the telephone into a continuously varying signal that is maintained in that form until it reaches the telephone at

the destination. In modern digital telephony, the analogue signal is translated into a digital signal. The analogue to digital conversion process usually takes place in the telephone set or at the periphery of the exchange, in the subscriber interface.

The two common methods of analogue to digital conversion are known as *pulse code modulation* (PCM) and *delta modulation*. In PCM , the electrical speech signal is sampled 8000 times a second and the magnitude of the sample is assessed on a scale of -127 to +127. This magnitude is then coded as a series of 8 bits, called octets. Thus each analogue speech signal gives rise to a digital signal of 8*8000 = 64,000 bits per second. Delta modulation also involves sampling the analogue speech signal, but at a very fast rate of 50,000 samples per second or more. In this case, the difference between one sample and the previous one is coded as either 0 or 1 depending on whether the sampled signal has decreased or increased in magnitude.

The conversion of speech to an electrical signal in the microphone is by its nature implemented completely by hardware. The analogue to digital conversion, and vice versa, is a highly repetitive and time-response-sensitive action. It is thus implemented in hardware by dedicated VLSI circuits.

User Interface

In the conventional telephony, the user interface is very simple (rotary dial or push buttons for input and dial-tones or bell for output). Rotary dials were introduced in 1886 at one of the first Strowger exchanges. They are still used on a majority of the installed telephone sets. However the push button key set is progressively replacing the rotary dial and is becoming the de-facto standard for user interfaces. The configuration of this user interface (arrangements of buttons, their names, order of digits - 0 before 1 or after 9, etc.) are standardise in CCITT Rec. Q.23/E.161.

In modern (digital) telephony, the user interface may additionally include (a small) LCD display, an extended keyboard and a number of LEDs. The digital telephone interface can have many variations and is user dependent and programmable. It is therefore implemented by software (with at most a few kilobytes of code) that executes on small microprocessors located in the telephone exchange. This software can be stored in (E)EPROMs or RAMs and downloaded from the exchange. A more recent trend is to use software, running on a personal computer (PC) with a user-friendly interface, to control a telephone set . Such PC software can be quite complex and large.

Signalling

Signalling involves passing control information between different parts of the network, such as the exchange and the telephone set, or between exchanges. The passage of control information is governed by communication protocols which are executed by both communicating partners.

Signalling is the basic function upon which all telephone operations are dependent. There is a large number of signalling protocols. There are national and international versions and flavours of signalling protocols, and there are signalling protocols for the exchange of information between he subscriber and an exchange, and among exchanges. Most signalling protocols are quite esoteric and there are very few experts that know and understand more than a few of these protocols. Unfortunately, an exchange that has to function in different countries must be equipped with a large number of signalling protocols even if in any actual system only a few of these will be used. The differences in signalling protocols and the need to implement a large number of protocols contribute significantly to the large cost of developing telephone exchanges.

In conventional telephony, the signalling between subscriber equipment and the exchange is basically quite simple but describing it briefly is no simple exercise. It involves the exchange of electrical signals between a calling station (subscriber handset) and the exchange to report 'Seizure' (handset "off-hook"), 'Clear Forward' and 'Address (or numerical) information', electrical signals from the exchange to the called station for 'Ringing current', electrical signals from the called station to the exchange for 'Answer' and 'Clear back', and audible tones from the exchange to a station for 'Dial tone', 'Ringing tone' 'Busy tone' and ' Special information'. The proper duration of the electrical signals, their electrical value and proper sequence, in other words *the communication protocol*, are described in CCITT Rec. Q.23 This protocol is quite simple and in traditional telephony is usually implemented by electromechanical devices. In modern handsets it is sometimes implemented by software running on small (4 bit) microprocessors.

Push-button telephone sets (usually) use a different signalling that employs *Dual Tone Multi-Frequency (DTMF)* coding. In a DTMF code, a digital signal is composed of two frequencies emitted simultaneously when a button is pressed. This coding is the basis of DTMF signalling defined in [CCITT Rec. Q.23, Volume VI-1).

In digital telephony, in particular for ISDN, the signalling (*Digital Subscriber Signalling* - DSS) is more complex and its implementation involves a number of layered protocols. DSS is

a sophisticated system intended for application to every telecommunication service, whether for transmitting voice, data or images. In addition to controlling communication of different types of services, DSS has to control *two* concurrently operating communication channels (B-channels) over *one* pair of wires. DSS controls access to an exchange by passing messages to it over a so-called D-channel (a channel is a 'portion of the information carrying capacity of an ISDN-interface'). Subscriber signalling is thus also called D-channel signalling.

The D-channel protocols are partitioned according to the Open Systems Interconnection (OSI) model. Only the three lower layers in the OSI model are considered for the basic definition of the digital subscriber signalling:

- layer 1 (the physical layer, CCITT Rec. I.430, Volume III-8)
- layer 2 (the data link layer, CCITT Rec. Q.920-921, Volume VI-10)
- layer 3 (the network layer, CCITT Rec. Q.930-940, Volume VI-11)

The dialogue between user equipment and the network is carried by messages of variable length. For circuit-switched calls, 23 message types have been defined and the description of the protocols is very extensive (435 pages of Volume VI-8 of the CCITT Blue Book).

The functionality of layers 1 and 2 is oriented towards controlling physical circuits, error detection and correction etc. It is thus quite repetitive and time constrained and is mostly implemented in dedicated VLSI circuits. The network layer is generally implemented as software executed on a processor in the telephone set or the subscriber equipment. As for any communication protocol implementation, this software must execute under real-time constraints.

The signal from the subscriber to the exchange provides information about the desired telecommunication service, such as the address of the called station. The exchange has to process this signal and this means that it has to switch the call to an outgoing line of the exchange. In case the called party is connected to a different exchange from the caller, a path through the exchanges must be established; signalling between exchanges is thus necessary.

As in the case of subscriber signalling, there is a large number of inter-exchange signalling protocols (DDI, E&M, CEPT-L1, Cailho, C11, C2 etc. CCITT R5 and national variations, CCITT 5, 6, 7 etc.) and they differ greatly in complexity. The simplest protocols date from the time when the telephone networks were designed only for the transmission and switching of analogue data. The most extensive and complex signalling systems are so-called common channel signalling systems. These are protocols employed over trunks dedicated to signalling, (common channel signalling trunks).

The CCITT Signalling System Number 7system can be seen as the culmination of various inter-exchange signalling developments. Signalling System No. 7 is a layered protocol that permits the application processes in a telephone exchange to communicate directly with their counterparts (e.g. applications in another exchange, or a database). The protocol is intended to support a wide range of digitally based services, and is optimised for data links operating over 64Kbps digital channels. It is also suitable for both national and international operation.

Signalling System No. 7 is described by a number of functional parts. By putting these together, it is possible to create a specific system according to particular needs ([CCITT Rec. Blue Book, Vol. VI, Fascicle VI.7/8/9]).

It has two functional parts: a common *message transfer part*MTP, and separate *user parts* (UP). The MTP serves as a transport system providing for reliable transfer of signalling messages between the locations of communicating entities. The UPs (such as telephone, data and ISDN, etc. parts) can be defined for specific private networks or for the transfer of information for management or maintenance purposes.

The MTP is defined in three layers:

- level 1: the *signalling data link*. (Rec. Q.702) defines a 'bi-directional transmission path, comprising two data channels operating together in opposite direction at the same data rate'. It relates to the physical medium (digital or analogue) which permits the full duplex transmission of information.
- Level 2 corresponds to the *signalling link function* and determines procedures for ensuring their reliability (REC. Q.703)
- Level 3 corresponds to *signalling network management functions* such as determining whether the exchange is the end-point for a message, routing of messages, and traffic management.

UP's are defined as level 4 activities of the System No.7 protocol stack. Every UP defines its level 4 functions, e.g. the messages (or signal units - SUs) and protocol for exchanging messages are defined for specific private networks or the transfer of information for management or maintenance tasks. The .telephone UP (TUP) is intended to serve the most important users of System No.7. It enables all types of telephone circuits to be served, whether they be analogue or digital, terrestrial or satellite etc. The coding of SUs is defined in Rec. Q.723.

System No. 7 is implemented in a large numbe- of dedicated IC (physical and link layers of MTP) but most of the functionality, in particular the UP's, are implemented in software that is executed on a number of processors.

Transmission

Transmission of the signal between subscriber equipment and the exchange takes place in a so called *local loop* , and in trunks between exchanges.

The subscriber loop carries both the end-user signal (such as digitised speech) to be switched and the necessary signalling information between the exchange and subscriber set. There is a variety of subscriber lines (loops). The most important are analogue subscriber loops used to connect to exchange telephone sets, facsimile machines etc. The digital subscriber lines, in particular ISDN subscriber linescarry information about the end-user and signalling information in digital (PCM) form.

The trunk line carries the end-user signals, and in the case of *associated signalling* trunks, also the signalling data. However, *common channel signalling* trunks do not carry any signalling because this is sent by means of data messages over a *common channel signalling channel*, which is another kind of line in an exchange. In order to maintain and operate an exchange, it is connected by lines different from those for common channel signalling to either data terminals or remote computers.

Switching

Switching is implemented in the telephone exchange, in particular in a part (or parts) of exchange called the switchingmatrix. However to perform the switching, additional functions need to be performed, e.g. the signalling information from the subscriber must be obtained and translated into commands to the switch, and if necessary passed on to a connected exchange. An exchange is a concentration point of telecommunication activities and thus the natural point for gathering information about the use of tele:ommunication services. As we will see, a large number of maintenance functions are implemented in the telephone exchange.

>From the viewpoint of the functions that an telephone exchange performs, we can conveniently think about an exchange as consisting of a *switching matrix* and *common control* surrounded with a multiplicity of (subscriber) *interfaces*:

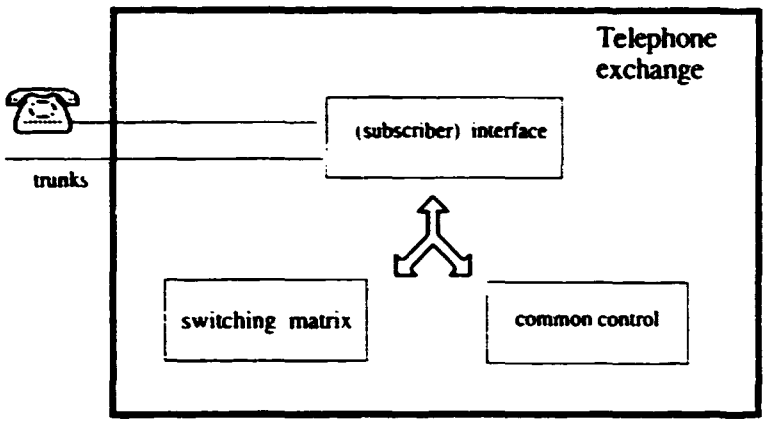

Figure 3: Telephone exchange functions

The *subscriber interface* is an access point for a line (whether analogue or digital) to the exchange. It interfaces the common modules of an exchange - the switching matrix and common control - with various types of analogue and digital user equipment. It translates a wide variety of user signalling systems into the functional signalling required for the call control function (and this is performed by common control) and handles time control tasks associated with line signalling, thus allowing the common control to concentrate on functional call control tasks. In structure, it consists of two parts: *peripheral circuits* (line circuits, trunk circuits, tone receivers, tone generators etc.) and *peripheral control units* for carrying out the lower-level control functions of the system, i.e. those related to the peripheral circuits.

At an abstract level, the first function of the subscriber interface is to separate the two components of the information stream flowing into the exchange over the line from the subscriber, namely the subscriber (information) signal - the end-user signal- and the signalling (and packet-switched data in ISDN exchanges). The end-user signal is transmitted through the switching matrix. The signalling and packet-switched data are processed by the common control. Signalling messages from the subscriber are translated into messages to the common control that executes the call processing functions. Packet-switched data is passed on to a packet-switch that takes care of further processing.

The subscriber interface functions may include different levels of processing capabilities, which make them appear as more or less intelligent peripherals of the common control. The peripheral control units can be seen as (peripheral) parts of the common control. Levels of processing, most of it software implemented, vary for each type of interface unit according to the functional architecture, and differ from exchange to exchange. The functional units that perform the

subscriber interface functions usually contain a real-time operating system, software to handle a specific type of signalling and to deal with the interface to common control.

The *switching matrix* performs the circuit-switching activities to carry out the physical, bi-directional and uninterrupted transport of the signal between the lines involved in a call. The switching matrix is usually a subsystem specifically designed and optimised to perform the circuit-switching function. In modern exchanges, the switching matrix is digital. It can be considered a black box that interfaces to the external environment by (internal) PCM lines. Analogue lines, for both subscribers and trunks, are first digitised and when necessary concentrated or multiplexed on PCM lines. The switching itself is usually carried out by specific hardware architecture. The switching matrix includes substantial internal processing capabilities which are often implemented as firmware functions. Typically, the witching matrix interfaces to common control through intelligent interface units implemented by microprocessors.

The *common control* receives, processes and forwards the signalling. It gives commands to the switching matrix about which lines should be interconnected (switched), when to start and when to terminate a connection. In some exchanges that handle ISDN traffic, the common control carries out also the packet-switching function.

The set of the activities carried out by the common control may be divided into two major groups: *switching activities* and *operation and maintenance activities*

The switching activities include everything related to the implementation of call processing; the operation and maintenance refer to the management of the exchange. In a comprehensive switching system, the activities are implemented as a library of application programs provided with relevant data structures and supported by an operating system. For each type of exchange, a suitable generic packageis produced from the library by integrating the desired functions (library packages).

Each generic has a typical size ranging from several hundred thousand to over 1-2 million lines of code. In terms of code size, the switching activities cover less than 25% of the size of a generic, the operation and maintenance account for over 55% and the remaining 20% is taken up by the operating system. However in terms of CPU time, the switching activities consume the larger share, especially during peak traffic conditions. The operation and maintenance activities usually place at off-hours. They are operated at lower priorities than switching functions and

may be delayed if necessary. For very large exchanges, the processing load can approach peaks of 1, 000. 000 call attempts per hour, with peak values of about 30. 000 calls in progress at the same time. Such processing capabilities are delivered by highly distributed and parallel systems. Thus we may think of the common control as a network of processors that share the functions to be executed or the devices to be controlled (in particular when peripheral control units are considered a part of common control),

The operation functions cover a broad class of features, such as *configuration* and *reconfiguration* of the parameters used in the exchange, as well as the organisation and characteristics of every logical and physical resource in the exchange. Also, the gathering of the charging data, traffic data and statistics related to the traffic flowing through the exchange comes under the operational activities.

The maintenance activities are concerned with *detecting faulty situations* (such as line failure), *locating and isolating* the consequences of a failure and providing tools to facilitate the *removal* and replacement of faulty elements. All this must be performed without interruption or substantial degradation of the functionality of the exchange. In order to permit high global availability of the exchange functions, the functionally important parts of the exchange are replicated so that when a fault occurs the faulty unit is disconnected and a replicated, redundant unit is put into service.

In contrast to the switching activities, the operation and maintenance activities are very similar to conventional real-timemultiprogramming applications in business and administrative informatics. In this segment of the market, the high competitive pressures result in the availability of systems with very attractive price/performance ratios. For this reason, the trend is to move the operation and maintenance activities away from the central control of the exchange into service computers that are based on widely available, standard hardware and software. The service computer may be located at the same place as the exchange or at a remote site with a connection to the exchange via a conventional data communication link. The service computer may be also connected to more than one exchange, thus enabling cost sharing and integration of the operation and maintenance procedures over a number of exchanges.

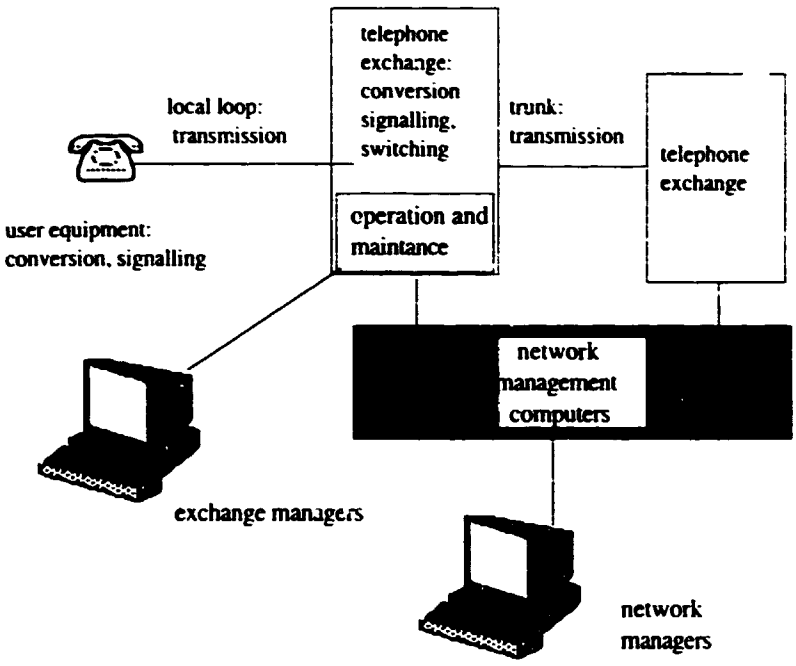

Figure 4: Communication functions

Management functions

A modern telecommunication network consists of a large number of heterogeneous *network elements*such as telephone exchanges, transmission systems, radio-based systems, computers etc. Network elements are usually manufactured by different companies and can differ in their functionality, year of deployment and sophistication. However, all these elements must be managed in a consistent way so that no resources are wasted and the services are provided to the users at the agreed level.

The central core of a management systemis a database which contains information on all the managed network elements. It provides a model of what is happening in the network. The model is continuously updated, gathering information from the network elements themselves, and implementing commands from the operators. The management system acts as an 'enabling layer' sitting between network elements, and network operation and maintenance personnel.

The network management functions are implemented in three activities that are continuously and concurrently executed:

- gathering information concerning the use of telecommunication services and the performance of the network elements,

- storage, processing and display of information to network managers, and
- executing the control commands of network managers by manipulating the elements of the network .

>From the point of functionality, the network management system can be structured in 3 functional units: *network elementelement managemen: and network management* From the organisational point of view, the management activities are divided over a number of Operation and Maintenance Centres (OMC) and a Network Management Centre (NMC). An OMC handles management of one or more network elements. It typically comprises all or some of the following applications:

- fault management, including alarm handling and graphical presentation of component status, routing of alarm messages, command log etc.,
- command and file handling, including time handling and command logging,
- collection of charging and/or performance data,
- functions to facilitate the handling and administration of network elements, and
- supporting applications such as authority management and network model handling.

The Network Management Centre handles the co-ordination of a number of OMCs. The following applications are some examples of NMC functions:

- network surveillance,
- management of System No. 7 signalling network, and
- network traffic management.

The OMC and NMC applications may be located in one integrated system covering the whole network or a specific part of the network.

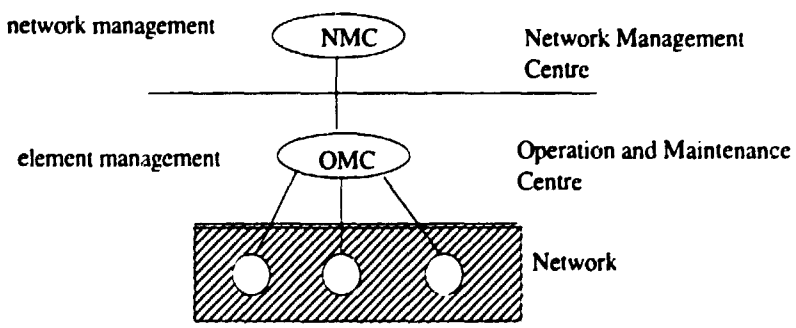

Figure 5: Network and element management

The systems to support network management are often developed by manufacturers of telephone exchanges. In the past, proprietary hardware and software were used but nowadays standards are the keyword. The operating system is often UNIX, the programming language C, communication between elements and centres is based on industry standards such as X.25 and SQL-compatible relational database management systems are mostly used. The hardware platform is often an industry-standard workstation or minicomputer.

2.4. Overview

function	software size	complexity	type
conversion	n.a.	n.a.	
user interface	very small	low	
user signalling	small	low...medium	srt
inter-exch. sign.	medium	low ...high	hrt
transmission	small	low	hrt
subscriber interface	small	low	hrt
switching matrix	n.a.	n.a.	n.a.
common control	very large	very high	hrt, srt, batch ...
element manag.	large	medium	srt, batch
network manag.	large	medium	srt, batch

In this table, the following abbreviations are used:

hrt ... hard real-time system, i.e. a system in which some activities must be performed within a defined window of time without fail (the so-called hard real-time requirement)

srt ... soft real-time system, i.e. a system with no hard real-time requirements, and with some activities that must usually be performed within a defined window of time

batch... a batch system, i.e. a system with no a soft- or hard real-time requirements

n.a.... not applicable

3. Main types of telecommunications software

In the previous chapter we have discussed the functionalities necessary to provide telecommunication services and the role of software in providing these functionalities. The main conclusion was that software is being used in most of the telecommunication subsystems: however the most important of these is the software in telephone exchanges, and the maintenance software.

In this chapter we discuss in more detail the properties of telecommunication software. Unfortunately, in the field of telecommunication there is no standard and universally agreed-upon terminology for software concepts and terms. Therefore we shall first define the main software terms used in this report and relate them briefly to equivalent terms also used in the telecommunication industry.

In the remainder of this chapter, we describe the telecommunication software of a *generic* exchange, without referring to any specific system. However to make the description more realistic, we will sometimes relate our description to software of specific exchanges.

3.1. Often used software-related terms

Layers, layered structure

In order to master the complexity of any large scale software package, the software is usually structured into (horizontal) This means that the total software is divided into hierarchically related sets of units of execution. These units of executions are sometimes called asks, processes, finite state machines etc. The terminology used by different manufacturers differs and we will use the term 'process. Processes that are higher in the hierarchy may use the services provided by the processes lower in the hierarchy.

Interface of a layer

The services of a layer can be obtained by sending a message requesting execution of a particular service, by calling a procedure implementing a service, or by manipulating a shared variable common to both layers. The interaction mechanism (message passing, procedure call, etc.) taken together with the interaction means (a particular message, a particular procedure call, etc.) is called the layer interface.

Process

A *process* is a unit of computation executing on a processing unit. During its execution, the process may be in one of a (possibly infinite) number of states. Processes communicate with each other by sending and receiving *messages*. A messageis dynamic entity: it can be created, destroyed, moved from one processor to another, etc.

Program

A process executes a *program*.. A program is a unit of compilation, i.e. it is a static entity. A program is written by a programmer and is compiled into executable code. A pool of worker processes may be created from a program allowing a particular application to allocate its worker processes to specific tasks.

Programs can also be grouped into *packages*, *i.e.* sets of functionally coherent programs that perform a function that is well-recognisable (and meaningful) with respect to a particular application. For example, a 'Signalling System No.7 package' can implement the functionality of the Signalling System No. 7. A package can be described in general terms without dealing with complex and tedious details.

The terms 'process', 'program' and 'message' are used here as generic terms. In different exchanges, depending e.g. on the programming language used or other considerations, different terms can be used. For example, when the programming language CHILL is used, the unit of execution is a CHILL-process. Other terms used are 'task', 'application activity' etc.

In telecommunication, the units of execution are often *finite state machines* (FSMs) which are processes that may be in only a *finite* number of states. Telephony tasks are carried out by a host of FSMs, each of which is in one of a finite set of states at every instant of time. The FSMs communicate by exchanging messages. Each FSM is allowed to receive a certain (finite) set of input messages and generate a certain set of output messages. That is, upon receipt of a message, an FSM investigates the states it is supposed to act on. Messages not allowed in any state are discarded. If a valid message has been received then the FSM changes to a (new) prescribed state, and if so defined, outputs a message.

The messages are sometimes called 'signals' or 'solicitations'. Communication by means of 'shared variables' or 'procedure calls' is also frequently used.

Processing unit

A processing unit consists of a processor (sometimes called a CPU), memory, and optionally I/O or communication devices. A processing unit is a mechanism to support the execution of processes.

Module

A module is a hardware or software component that is discrete and identifiable. A module can be considered as an abstract entity, a 'black box'. It is designed to perform a function or a group of functions, independently of its actual construction. A module is essentially defined by its interfaces - hardware and software - with the other modules, and not by its internal organisation.

3.2. Operation software

The operation software, i.e. the programs located in the exchange, comprise of two layers: the application layer and the operating system layer. *Application programs* control the operation of the exchange. The *operating system* provides administrative support for the application processes.

The application layer and the operating system layer can be further divided into more layers. In most telephone exchanges, the operating system consists of at least two layers: a *'kernel'* layer and an *'additional services* layer. The application layer is often structured in more than two layers. Very often a layer on top of the operating system abstracts the specific (and often widely different) characteristics of telephony hardware into a less divergent functionality. We call this the telephonic support layer. On top of this layer, the typical telephony functions such as call processing are added, to form a telephony application service.

For example in ITT 1240, the application layer is divided into a 'telephonic support 'layer, a 'call handling and maintenance' layer and an 'administration' layer . The operating system comprises of an 'operating system nucleus' layer and an additional 'services' layer (such as time , input/output, recovery, and network services). The operating system comprises of a nucleus and a number of services.

The application layer can also be described as a set of packages. The size of application packages used in telecommunication lie in the region of several tens of thousands of lines. We will divide the telephony application processes into call processing, and operation and maintenance applications.

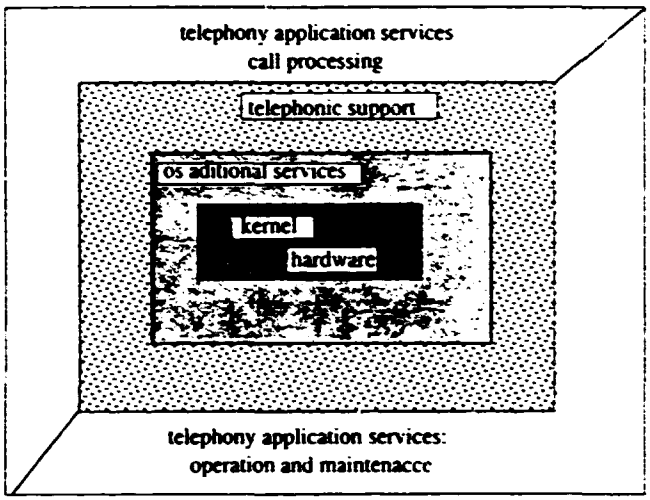

Figure 6: Operation software

3.2.1. Operating system

The fundamental task of the operating system is to allocate the system resources, such as processors, memory, I/O facilities etc. to the various activities that have to be performed. In telephony, this allocation is complicated by the fact that some of the activities (1) have to run *concurrently*, (2) are *time-critical* and (3) have to be performed in spite of possible hardware of software failures (i.e. be *fault-tolerant*).

There may be hundreds of processing units in a telephone exchange, each executing a process. In addition, on each processing unit, a number of processes may be *active*, i.e. be in different stages of execution. Each of these processes needs from time to time to get access to the processor and perform some of its assigned activities. An operating system must co-ordinate all the activities on each processing unit *and* on all the processing units taken together.

The term time-critical means that an activity must be performed within a prescribed time (deadline) or not at all. The count-down usually starts at the moment of request for such an activity and missing a deadline is considered a serious system error.

The need to provide a fault-tolerant execution environment for the application activities means that redundant resources must be available and the operating system must manage the replacement of failed resources.

In addition to the complications described above, the hardware and software resources of a system can be very extensive. For example, in a medium-size exchange, there are several hundred microprocessors that perform tasks ranging from DTMFservice reception to the running of operational and maintenance software.

A real-time, fault-tolerant operating system must support the following activities:

- process management(loading, creating, starting, terminating and destroying of processes).
- schedulingof processes (co-ordinating the assignment of processes to a processor so that all activities, and in particular all time-critical activities, are performed on time),
- interrupt handling (handling of hardware created signals),
- memory management (assignment of static and dynamic memory for process activities).
- I/O device management (handling of communication with peripheral devices, such as display units, magnetic tape and disk units, communication network units etc),
- inter-process communication (by means of messages, semaphores, critical regions etc.).
- file management(handling of mass-memory),
- time services (providing accurate time and calendar functions), and
- system integrityrecovery services.

These services may be delivered by one monolithic software package or, as is present day practice, in two layers with a kernel or nucleus of the operating system in which only a minimum of facilities is provided, and to which at the next level additional features can be added (where and when needed) by the application programmer writing in a high level programming language.

In practice this means that (1) each processor contains a local operating system i.e. the kernel and additional services that are required by the processes running on the processor and I/O devices connected to the processor, and (2) some processors also contain services necessary to co-ordinate all local activities - *global operating system* services. The local operating system is sometimes called a *module manager* and the global operating system a *system manager.*

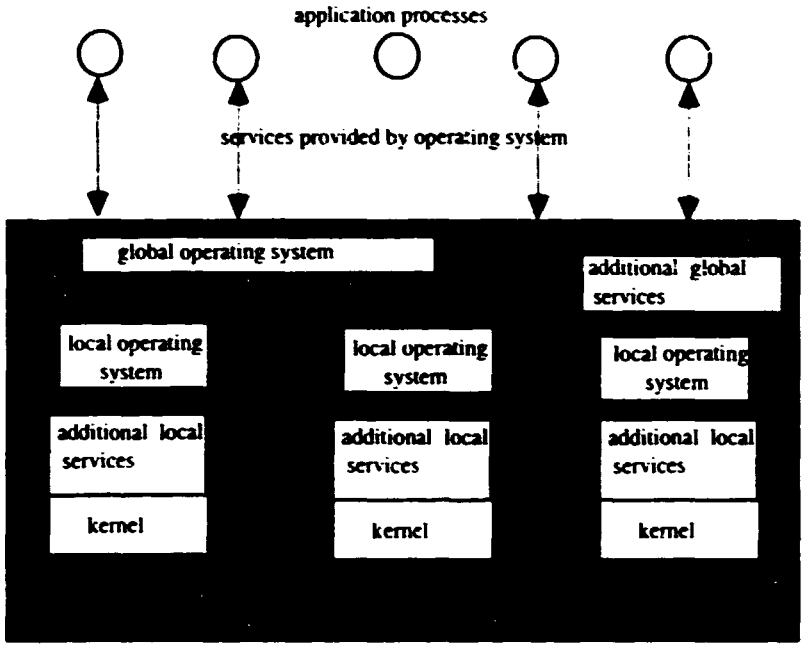

Figure 7: Operating system

The additional services can be provided by means of *kernel processes*. These processes differ from application processes in the sense that they enjoy privileges not available to 'normal' application processes (e.g. to execute at higher priority, be guaranteed memory space, etc.).

In the following sections we will describe the typical services provided by the different parts of the operating system. By means of examples, we will show possible services provided by the system. In different exchanges, the terminology used may differ both in semantics and syntax. The primitives we define are in some sense typical but certainly not the only possible ones.

3.2.1.1. Operating system kernel.

The operating system kernelprovides only the minimum functionality needed from the operating system. Usually, the kernel dynamically administers the data areas for the processes (assigns memory to processes, manages pools of buffers for communication). Another important function is the basic handling of internal exchange messages (management of queues by adding to and removing messages from a queue) and the actual transmission of messages to another processing unit. The kernel also provides facilities for the management of overload conditions (e.g. deciding to drop a message when a message queue exceeds a specified length, terminating a process if a certain deadline has passed, etc.) and the basic scheduling support (usually pre-

emptive, priority based and cyclic process schedulers are supported) . Interrupt handling, basic I/O management and management of local hardware timers are part of the kernel.

An operating system kernel may be an industry standard kernel (such as Motorola RMS68K) or a kernel developed specifically for support of telecommunication processes (CHILL-processes or FSMs).

3.2.1.2. Additional services

Process management

Local process managementprovides services related to actual process activities on the processing unit. The following operating system services are usually provided: *create_process* (process resources are allocated, the code of process is loaded, message queues are created, the process identification is set and made known, etc.), *destroy_process* (the process resources are freed), *start_process* (the process is made ready to run, i.e. scheduler is informed about ready-state of the process), and *terminate_process* (the process resources remain allocated e.g. for debugging purposes, but queued messages not received are deleted to recover memory space etc.).

Global process management controls the initial start-up of the system and any subsequent reconfiguration that may be required. It maintains a database of the programs and their properties. It also maintains a database of processes created from these programs and their properties, i.e. the processing units where they are executing, their priority, messages they can receive and their actual state. It can thus support the relocation of processes and dynamically control overload conditions.

Local process management and global process management co-operate in the scheduling of processes. Global process management allocates a processing unit for the execution of a process based on the current load of each processing unit and the availability of the resources required by the process. Local process management then actually performs the scheduling. Priority-based, round-robin and pre-emptive scheduling methods are usually employed.

Time services

Many processes require either periodic scheduling or time supervision. Both functions are provided by the time services which often include time-of-day and calendar functions. Local timing management measures all relative time periods and maintains the local absolute time using the timer services of the kernel. Global timing management synchronises the local

absolute times (using a particular clock synchronisation algorithm). It also performs additional time adjustments, such as changes for summer and winter times.

Input/output

The I/O systemsupports access to computer peripherals. It performs file management and device handling for physical access to the mass memory devices as well as to the visual display units. Usually it hides the details of the I/O interface behind a standard message-based interface. The I/O system is typically implemented as part of the local and global operating system. The division of the activities is very implementation specific.

Usually, logical and hierarchical file systems are implemented so that a process can access a file without needing to know the type of peripheral device on which the file resides. The I/O system translates between the logical file and the physical file characteristics and the location of the device. For example, a billing process outputs charging information to a billing file: depending on the exchange configuration, the information may be directed to a remote billing centre via a data link, or to a magnetic tape unit attached to the exchange.

The advantages of this type of I/O system are well known, e.g. from the industry-standard UNIX-oriented environment: it makes the software independent of particular peripheral hardware configurations.

The I/O system may also manage access to data files that are duplicated for security reasons, broadcast outputs to several devices (e.g. visual display and printer), and determine the fallback device in the event of a failure. Another major function of the I/O system is the management of man/machine interfaces.

System integrity control and recovery

The coherence of hardware and software is controlled and co-ordinated by functions (software and hardware) that are distributed through all the processing units. These functions must be able to (1) supervise and detect failures, (2) locate the cause o any failure (fault) , (3) isolate the fault and (4) detect and if possible repair the consequences of the failure. All this must be done without substantially degrading the functioning of the exchange.

The supervision is done either continuously or intermittently (upon request, or at intervals). Continuous supervision is done mostly by hardware implemented functions. Any device that is crucial to the exchange operation is usually equipped with specific circuits that continuously

analyse the device's condition. An anomaly tin the behaviour is reported to a *recovery* process (which may be a local or a global kernel). Recovery processes may be a part of global process management but often are implemented in a separate module.

The recovery processes consist of hardware and software fault diagnosis kernel processes. They guard the condition of the entrusted hardware and software and report occurrences of faults and errors to global processes that log error reports, analyse the faults and errors and when necessary reconfigure the hardware or make a process restart (corrective maintenance). Global recovery processes can also periodically start audit processes that examine software and detect, locate and attempt to correct certain classes of errors. An audit may also be requested by the exchange operator using the man/machine interface (e.g. for pre-emptive maintenance).

Examples of error conditions are invalid, dropped or lost messages, prematurely terminated processes, inconsistent call processing conditions, etc. The log reports include as many error details as possible. This information is used off-line for debugging or tuning the system

An important part of the recovery procedures is to control the consistency of data (e.g. call related data, semi-permanent data such as subscriber equipment number, directory number, class of service; or data containing the description of the system configuration). Such data is interrelated but is often distributed over a number of processing modules. The update may be requested from a man-machine interface, or internally invoked due to changes in hardware or software status, and must be updated on different processing units. A mechanism to control the concurrent updating of database and perform databases recovery operations is thus required.

The integrity of databases is guarded and provided by a combination of the following mechanisms:

- validity checks: before a change is performed, checks of security clearance, the meaningfulness of the request , etc are performed.
- roll-back mechanism: when data is changed, both the old and new values are stored in a secure file. Should a hardware failure occur during the update, the roll-back facility resets the data to its value at the start of update.
- roll-forward mechanism: periodically, a copy of the database is stored on a secure file to provide a historical record. If data is lost, the database is restored by updating the most recent historical record.

Inter-process communication

In a telephone exchange, and in particular in support of the call processing function, interprocess communication plays a very important role. A major part of the activities of an exchange consists of receiving and sending intra-exchange messages and, consequently, a large fraction of the processing power must be reserved for interprocess communication.

The resources required to receive and send messages and process their contents must be 'balanced'. What exactly is considered a balanced situation depends on detailed architectural considerations, but as a general rule an implementation of interprocess communication is well-balanced if equal resources are used to process messages and to communicate messages.

The operating system provides a number of interprocess communication services. Again, the type of messages, their semantics and their syntax may differ from exchange to exchange. In general the following services are provided: *send_message*, *receive_message*, *receivecase_message*. Usually the send and receive messages are *asynchronous*, i.e. the sending process does not wait until the receiver is ready to receive the message. The duration of *send_message* is time-bounded and a-priori, statically known. *receive_message* waits until the (specified) message can be received, and *receivecase_message* waits until one of a set of possible specified messages has been received.

Interprocess messages are transmitted as (variable or fixed length) packets. When a process intends to send a message, the kernel must make a contiguous memory area available to the sending process. For this purpose, the kernel manages pools of memory each of the size of the packets that will be transmitted.

Depending on the implementation, there may be separate pools for kernel processes and application processes, short messages and long messages etc. The particular choice is strongly dependent on very specific design decisions and thus is different for each type of exchange.

When the process obtains a memory area it fills it with the message. A messageusually contains a destination address which provides an unique identification of the receiver process. This address can be a logical name (a process id) or a physical port to which the message must be sent. The destination address is followed by a message body and usually some control data (such as identification of the sort of message that is being sent, the length f the message and an error check code).

When a process sends a message the packet is usually placed by the kernelin a queue. Depending on the implementation, there may be a number of queues: e.g. separate queues for outgoing messages and incoming messages, kernel process messages and application messages etc. The messages are then moved (or transmitted) by the kernel from time to time, from one queue to another. If the receiver process is located on another subsystem, physical transmission of the packet is required; otherwise the packets are copied from one queue to another, or , as is mostly the case, pointers to the packets are copied.

A process waiting for a message is notified by the kernel of its arrival (i.e. the process is scheduled for execution) and the contents of the message are then delivered.

As already stated, there may be many differences in the mechanisms provided for interprocess communication and in the semantics of the send and receive operations: these may be driven by the (often) strong preferences (and tastes) of the system implementors.

3.2.2. Application software

From the initial signal on the incoming line to the final release of both incoming and outgoing lines, each call is guided, controlled and administered by a complex set of interacting activities. Each of these activities is implemented by a number of processes that may run concurrently on different processors.

The exact responsibilities of each software process, as well as number of processors used and the distribution of processes over processors, can vary widely. The simplest is to have no distribution at all: all software runs on one processor (and if the processor is duplicated it is for reasons of reliability, not to divide functional responsibilities).

This centralised and monolithic architecture has some inherent advantages and disadvantages. On the one hand it provides substantial flexibility and cost saving because the software, once developed and tested, can be replicated at practically no cost at all. On the other hand, centralised control is quite inefficient in dealing with frequently occurring, time-critical peripheral events (e.g. line circuit control). This inefficiency may be partially reduced by employing hardware circuits (combinatorial or sequential) to deal with peripheral activities. However this strategy is successful only when one attempts to implement simple functions (like line signal processing on subscriber loops). Implementing the functions that are necessary for

complex exchange termination and common channel signalling using only hardware circuitry is both difficult and uneconomical.

Obviously the applicability of this kind of monolithic architecture is limited by the load the exchange has to handle. More precisely, it depends on the number of call attempts per unit of time with which the exchange must deal during peak hours, and. the processing load associated with operational and maintenance tasks. Consequently, centralised control, with non-intelligent peripheral controllers are economically interesting for small exchanges (less than 1000 lines) but for larger exchanges other approaches must be used.

Basically what is necessary is to distribute (or off-load) some of the software activities to other processing units. One such an activity that can be off loaded is the handling of telephony devices, such as subscribers, trunk lines, auxiliary devices etc. The processes at the 'main processor' then see each line as transceiving complete telephone signals consisting of well defined messages instead of electrical or binary versions of signals on the line. For PCM lines the device handlers can implement insertion or extraction of the associated signalling, test routines etc. For Common Signalling Channels, processing at the physical and link layer level can be easily distributed.

In addition to telephony devices, other parts of the hardware can be controlled by their own processing units. For example, the operation of the switching matrix can be controlled by a dedicated processing unit, creating a virtual matrix that performs connections and disconnections, diagnoses the proper behaviour of the matrix, etc.

Other activities (e.g. those related to operation and maintenance) can also be assigned to their own processor. For example, the operator interface which has the tasks of (1) analysing each person-machine interaction, including the identification of the action requested by the operator, (2) displaying information in a user-friendly way and (3) receiving messages indicating anomalous behaviour and presenting them at the proper time in the best possible way, etc., are all examples of functions that may be off-loaded to a separate processor(s).

The advantage of having a single processor to execute all the software activities is that all the logical and physical resources are administered at one location, making it possible for the run-time system to dynamically optimise on the use of the resources of the exchange. The potential disadvantage is that the growth o the exchange in terms of the traffic, the number of exchange lines, the number of trunks and additional services to the subscriber, etc. must be

accommodated by the same single controller and the central processor must be capable of dealing with the maximum capacity expected to be necessary at any time for that location. Thus the central processor will for most of the time be over-dimensioned, leading to a higher-than-necessary hardware cost for the whole exchange.

These considerations suggest that there are economic reasons for distributing control of the exchange's resources. However, there is an overhead associated with the communication and synchronisation associated with distribution, and this grows with the (physical) distance between the processors. For example, generally communication using shared variables by two processes running on the same processor has less overhead than communication by message passing between processes on different processors. Thus if the number of processing units (the distribution size) is to be increased, the communication mechanism between the processing units must be very efficient. In some architectures, in addition to the circuit switch for the end-user (speech) signal, there is a separate packet switch to handle message communication between the processing modules. In ISDN exchanges, this packet switch is sometimes used also to handle the end-user's packet traffic.

Another problem with distributed control is the fragmentation of resources: the processing capabilities of two processors are not equal in all respects to the processing capabilities of a processor of twice the processing power of a single processor. This means that the necessary total processing power (the sum of processing powers of the processing modules) must be larger than the processing power of a single processor to handle the same traffic. However, the *cost* of a large number of small processors is usually *smaller* than the cost of an equivalent large processor. In particular, when use of an industry standard processor (such as Intel's 80*86) is possible the hardware cost of strongly distributed exchanges is very acceptable.

Generally the problem of finding a good software architecture for a telephone exchange is dependent on finding a partition of software into sets of communicating processes with an acceptable trade-off between processing time and external communication time; processes in the same set may communicate more intensively with each other than with processes in another set.

The large variety in exchange architectures shows that many good distributed designs are possible. In general, the present trend is towards greater distribution, and more recent exchanges usually have more processors than older exchanges.

For good distribution the structure of the software should mirror the functional structure of the telephony activities. For example, in a telephone exchange there is usually one incoming and one outgoing circuitassociated with a call and a natural basis for the functional division of call handling is between the incoming and outgoing halves; there is also a clear distinction between call set-up and call supervision. As a result, call handling can be distributed among four different functional entities.

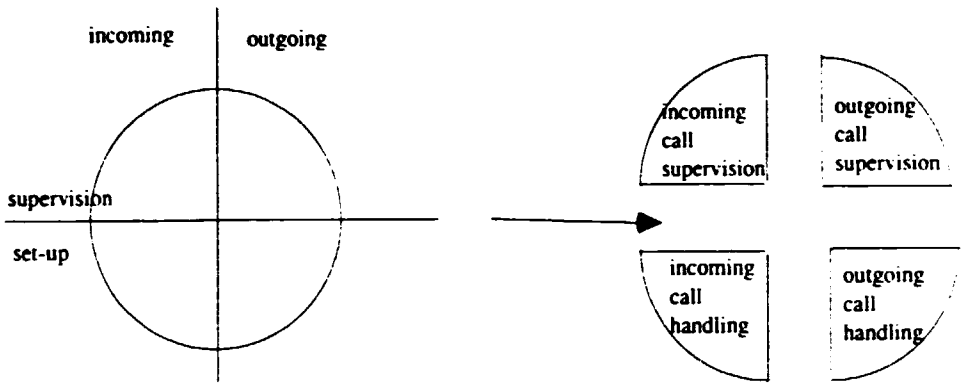

Figure 8: Call handling structure

These four functional units share operations on a common database containing common tables for number analysis, routing etc. on the switching network.

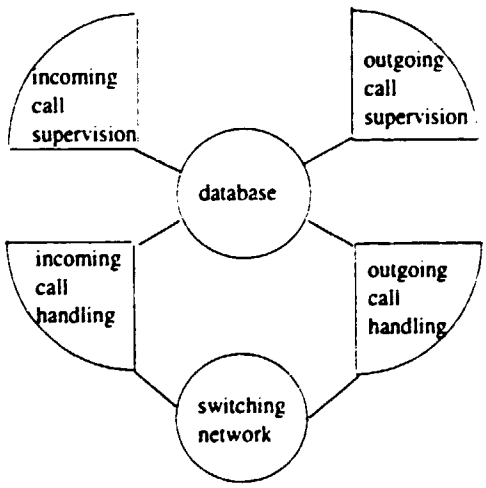

Figure 9: Call handling functional units

Thus a possible way to translate this functional view into a software architecture is to have sets of processes to (1) supervise incoming calls, (2) supervise outgoing calls, (3) handle incoming calls, (4) handle outgoing calls, (5) control the database, and (6) control the switch. The processes to supervise outgoing and incoming calls, and those to handle incoming and

outgoing calls, can be assigned to separate processors; the databases and circuit switching control can be assigned to the same process or as the handling of calls, or they can be assigned to additional processors.

The way in which these functional entities can be translated into processes and distributed over processors also varies widely. In ITT 1200, distribution has been pursued to the fullest: the system is organised around a self-routing switching matrix to which peripheral modules are connected. There are two kinds of peripheral module: those that control some telephony termination units and those that contain only a processing unit that may control a computer peripheral (but not a telephony peripheral). The switching network is controlled by the devices connected to the switching network. There is no centralised device to set up and release calls (in other words, the function that controls the switching matrix - usually called the marker - is distributed over the peripheral modules). The database is distributed over a large number of peripheral modules. The peripheral modules are fairly small, so in most cases an exchange is built up from a large number of modules.

In other architectures (5ESS) the system may be divided into a small number of large units that are connected by point-to-point lines (or through a small switch). Each unit can be considered a complete exchange, containing all the necessary functions to allow independent operation and to switch calls on connected lines. Each unit contains a switching network controlled by the marker and necessary data. In this type of exchange, only some functions are centralised (e.g. administration, operator interface).

Each exchange must deal concurrently with successful calls and with call attempts, i.e. unsuccessful calls that for some reason do not reach their destination. For each call, successful or unsuccessful, a *transaction* is defined in the exchange, and a *call record* associated with the call keeps all the necessary details. A call record is generated when a transaction starts and is updated during the call. After a call has been completed, the contents of the call record are processed (usually on a separate processor) and stored if necessary for later use (e.g. to generate billing data, or for statistical analysis of the operation of the exchange). Usually these activities take place on dedicated processing units.

We shall discuss the functionality and the implementation of the application layer in the following two chapters. The next chapter contains a discussion of software to manage call processing. During this phase of a call, the activities of telephonic devices have to be supervised and supported. These activities we will therefore call *Telephonic support* activities. Activities

that take place during the set-up (and are related to set-up) are at the heart of telephony and can therefore be called *Telephony applications*. The functionality of the marker and the databases, the administration and maintenance functions as well as the user interface functionality will be discussed as part of the telephony application layer.

3.2.2.1. Telephonic support

Telephonic support functions are a part of the application layer software and implement a part of subscriber and trunk lines interface.

A generic hardware architecture of a subscriber line interface is shown in the following picture

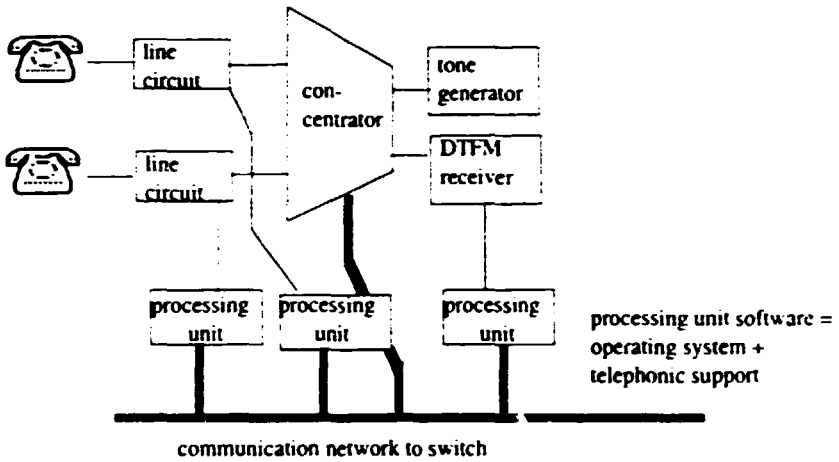

Figure 10: Subscriber line interface unit

The subscriber line circuits (analogue or digital) perform a number of interface functions commonly referred as BORSCHT(Battery feed, Overvoltage protection, Ringing current, Supervision, Conversion, Hybrid, and Testing) functions. They are often implemented using hybrid IC's and LSI devices such as SLICs (Subscriber Line Interface Circuits) and CODECs. Subscriber line traffic goes through concentrators which allocate time slots for signalling devices. The tone generator feeds the ringing current to the analogue line circuits. The DTFM receiver is responsible for the reception and decoding of DTFM code signals from push button receivers sets. During the selection phase, calling subscriber lines using DTFM signalling are switched on this receiver which detects, digit by digit, the frequency pair generated by the calling subscriber equipment. These processing units (called peripheral control units) consist of a small microprocessor (such as a Z-80), tens of kilobytes of memory and interface IC's for the communication network and for communication with the controlled device.

The architecture of trunk units is very similar: there are trunk circuits instead of line circuits, and trunk senders and receivers instead of DTFM receivers (e.g. R2senders and receivers).

The telephonic support functions are located on the processing units and are implemented as (application or kernel) processes, often as Finite State Machines (the processing of signals can usually be formulated as operations of a FSM). They run under the supervision of the operating system of the processing units and supervise and control the entrusted telephonic resources.

The basic task of the telephonic support layer functions is to abstract the widely ranging functions of telephonic devices into a homogeneous set of functions that can be used by call processing software. This abstraction is often realised in two software layers: a device handler layer, and a signalling layer For each hardware device, the device handler layer contains a device handler which translates electrical signals obtained by scanning the termination circuits into logical signals intended for the signalling layer or, in the opposite direction, drives termination circuits in accordance with logical signals received from signalling software.

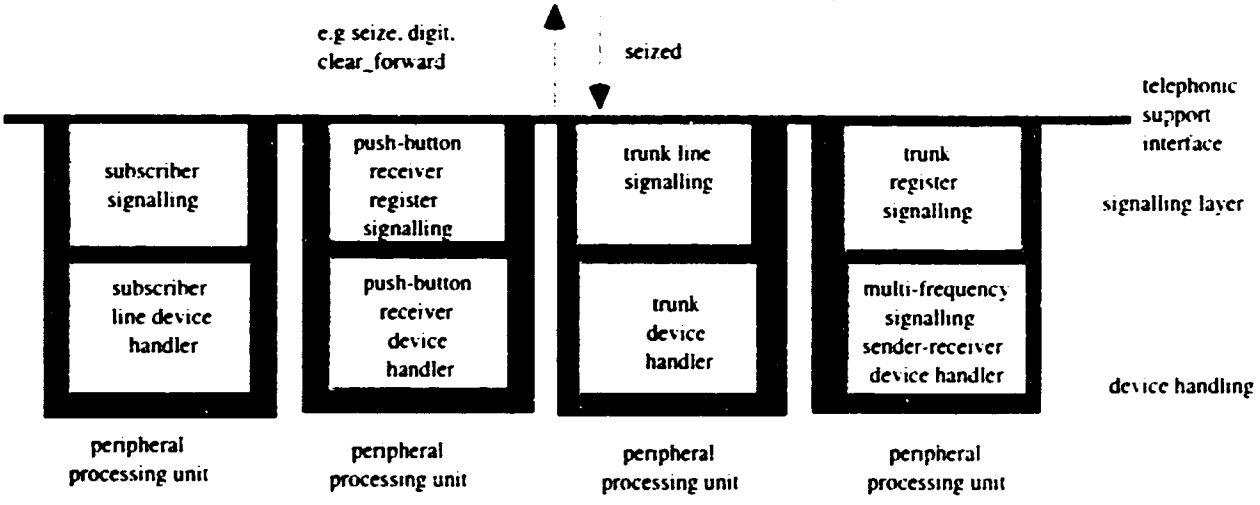

Figure 11: Telephonic support layer

The signallinglayer functions assign a telephonic meaning (a telephonic event) to logical signals from a device handler and, in the other direction, converts telephonic events into drive orders. Usually there is a different signalling function (a set of FSMs, or a package of programs) for each type of line employed. Thus there may be an Analogue Subscriber Line signalling function, an Analogue Trunk Line signalling function, and so on.

The signalling functions usually cater for all major signalling systems. Each function taken in isolation is quite small, but the great variety in signalling is one of the causes of the large size of telecommunication software.

Each signalling function operates on a set of variables that exhaustively describe the current operating conditions and configuration profiles of each line, its telephone terminals at the subscriber premises and the operating conditions of the lines (in service, out of service, under test, etc.). These sets of variables constitute the reference model of the exchange and their updating is under the exclusive control of the telephonic support functions.

It is good design practice to define a generic interface between the telephonic support (i.e. signalling) and call control functions. This means that all sets of signalling functions work on the same (small) set of messages to call control. Such a generic interface helps to hide the complexities and idiosyncrasies of signalling systems, and separates the maintenance of telephonic hardware from the call processing software.

3.2.2.2. Telephony application

The telephony applications are divided into two groups: call processing and operation and maintenance applications.

Call Processing

The call processing function is carried out by the part of the application layer responsible for the execution of the functions necessary to set up, hold, supervise and release a connection between two subscribers. In our layered model, the call processing software uses the services of the telephonic support layer. It provides services to administration functions (such as charging).

A telephone call is made in an automatic exchange by connecting (at least) two pieces of terminal equipment. (In multi-partyor conference calls more than two pieces of terminal equipment are involved.) This equipment may consist of subscriber or trunk circuits, and one is the calling party and the other the called party.

Usually, the two pieces of terminal equipment are handled independently (asynchronous processing). The call can then be handled by two separate processes, each dealing with half of the call. Each half-call process deals with the recurring events and telephone messages associated with a call (off-hook, answer etc.) and initiates the appropriate sequence of actions. The functionality of the half-call processes is easily understood in terms of a possible sequence of activities for setting-up, supervising and terminating a call.

Call processing for an *automatic* call (without operator intervention) starts when a seizure signal is generated by a signalling process. The seizure signal contains a parameter for the co-ordinates of the line, i.e. an internal identification of the line. The seizure signal usually results in the dynamic creation of a process (by the operating system, or a special dispatcher process of the application layer) to handle the setting-up of the call on the incoming line (incoming half-call or IHC, process). If there is congestion, the operating system may reply to the seizure message with a negative acknowledgement and the call is terminated. Otherwise, the IHC process replies to the signalling process with a message containing its own identification (call index) so that the signalling process may communicate with it directly.

At the same time, the IHC process opens a call record and puts in it a reference to the call index and the co-ordinates. From that moment, the call record will be used to save all the important events in the history of the call.

The IHC process is then able to process other messages that may arrive from 'its' signalling process. When the IHC has processed a sufficient number of incoming digits, they are written in the call record and the string of digits is passed to another application process (or processes dedicated to call routing (CR). Based on this string, the CR process analyses the request and if there is a possible route provides the co-ordinates of the outgoing line on which the call must be forwarded. If call routing is not able to find a route (e.g. insufficient digits, or congestion) the IHC attempts to resolve this problem. If it does not succeed, it terminates the call and deletes itself.

Route selection is based on the dialled digits and other information, such as the class of incoming line used by the call. The outcome of the routing request is always one of the following decisions: 1) the number of available digits is insufficient to carry out the routing: more digits are needed, (2) the digits are inconsistent with the routing alternatives allowed in the exchange so the call attempt must be cleared, (3) the number received identifies a subscriber belonging to the exchange and provided with only one line: the call must be forwarded to that line, (4) the number received identifies a subscriber with multiple lines connected to that exchange, and (5) the number received identifies a special service offered by the operating company to its subscribers.

Routing is done by the routing process using a set of tables. These tables are semi-permanent, i.e. they change infrequently and usually only by action of the exchange operator. The tables are organised in a tree-like structure and each node corresponds to a digit in a valid telephone number. By parsing digits according to this tree, the routing process attempts to find out whether the number is meaningful, and if so, which subscriber line or trunk leads to it.

After choosing the outgoing line, IHC creates an outgoing half-call process, or OHC process, which mirrors the operations of IHC and controls the signalling on the outgoing line. For that purpose, the OHC must send a message to the signalling entity that handles the outgoing line. After the signalling (outgoing) process has been activated, the IHC and OHC co-operate closely by exchanging messages. OHC controls the activities of its signalling process.

It is possible that the incoming and outgoing lines use different signalling. In order to simplify the communication between different signalling systems, the interworking between the IHC and OHC uses a set of standard messages based on the CCITT Recommendation. For this purpose the telephone events are divided into two groups: forward interworking events and backward

interworking events. These events have been defined identically for all the signalling systems expected in a telephone exchange.

If an IHC and an OHC find that the telephone call cannot be completed, they release their lines according to procedures specified according to the signalling system. However, if everything proceeds successfully and the called subscriber answers, a connection on the circuit switch is requested. This is typically done by OHC which realises first that the called party has answered. OHC sends the request message to the routing process an requests a connection.

If the routing process (part of the marker) has found a route through the circuit switch and the call has been put through, the exchange may charge for the call. For that purpose IHC or OHC attempt to identify the charging rate. Upon request, one charging process (or more) provides charging information to the requester and writes it into the call record.

When either the subscriber hangs up, or the call ends for any reason, the IHC and OHC both request the marker to terminate the connection and they then carry out the release procedures. When the call ends, the call record is sent to the process(es) that prepare statistical and charging information.

Calls can be charged according to two different methods: by 'time and charges' and by 'pulse metering'. In the first case, the charging process writes to the call record the date, time (hour, minutes and seconds of connection start-up) and duration and any special service indication. In the case of pulse metering, the charging process must find out (just like the routing process) the destination of the call. Then, from an administrative table, the charging process derives the origin of the call and the charging rate function to be applied and records this in the call record. Additional messages to the process controlling the frequency of pulse generation may be required.

For semi-automatic calls, multi-party calls, supplementary facilities and other types of application activities such as packet switching in ISDN exchanges, there are similar sets of processes.

Of course, the software structure described above is not the only one possible For example, it may not be necessary to create a separate process for each call: one incoming call 'super-process' may be able to handle more than one call (perhaps upto 16 calls), so that on a seizure, the signalling process reports to this incoming call 'super-process' which then handles the call in

the usual way. If this 'super-process' reaches its maximum capacity then new 'super-process' is created to handle next group of calls. And another software structure could use a single process to handle *both* incoming and outgoing halves of the call (synchronous processing).

The application processes can be divided into two classes: those that control particular hardware or a part of the database located or controlled by a specific dedicated processing unit, and those that are not bound to any particular location. The processes that have to control a particular resource must be executed at a location from which the control is possible. There is thus little flexibility with respect to the allocation of these processes to the processing units. But for the unbound processes, there is an allocation flexibility that can be used o perform load sharing and/or to increase the availability of the exchange services. Such processes may be allocated at the moment of creation to a processing unit that is the least loaded one that is correctly functioning. In the case of load sharing, the allocation dynamically matches the overall traffic of the exchange to the available hardware configuration. For increasing availability, the allocationcan take account of available processing units, or in the case of failure, reallocate the processes to available processors.

Operation and maintenance

These functions cover a broad class of facilities necessary to provide reliable, economical and operator-friendly functioning of the exchange. They can be divided into the following groups: (1) Configuration control, (2) Performance measurement, (3) Maintenance and (4) Person-Machine interface.

The exchange software is informed of the hardware capabilities of its exchange by means of *configuration* parameters. Obviously, the value of these parameters must be kept continuously up-to-date and consistent and this is complicated by the fact that these values are used by processes running on different processing units: indiscriminate change of a value could lead to different values being observed by processes at different times, thus jeopardising the consistency of the software. No change must therefore be made without taking the necessary precautions. The configuration parameters are also administered by the operators (or computers, as part of maintenance activities) external to the exchange. These highly independent and sometimes conflicting activities must thus be co-ordinated.

Every modification request causes specific procedures to first check the consistency, feasibility and reasonableness of the request. If a request causes problems that cannot be resolved, it is denied without interruption of the exchange operation.

The updating of a parameter used by only one process is usually performed by that process and is relatively simple because it involves little co-ordination. Updating a parameter used by multiple processes is more complex and very often a procedure is specifically designed for this. In general, the processes using a parameter to be updated are warned and this causes them to complete their ongoing activities and then go to a standby mode. When all such processes are standing by, the parameter is updated and normal processing is resumed. In exchanges that consist of a large number of processing units, the configuration activities are therefore in general more complex than in a more centralised exchange.

Performance measurements are made of the traffic (e.g. the number and types of call requests) and the system resources used. Usually they are carried out by executing additional software commands (writing into a call record or into special counters, reading counters, starting and stopping timers, etc.) during the execution of the program. The performance measurements typically consume additional system resources and must be thus used only when requested and necessary.

The performance measurements are executed as a result of operator requests and deliver the results as files to be processed off-line to produce statistics.

For traffic measurements, an international general reference model has been defined based on (1) counters and state variables that are associated with a set of telephony objects, (2) specification techniques to specify the time at which a measurement should be performed, an (3) the type of measurement to be made. The operator may thus specify in a compact, clear and standard way the measurements to be taken.

The gathering of charging data can be seen as a particular type of a measurement: it is (continuously) executed by the call processing software and produces files that are further handled by a billing centre.

The management of system integrity is in principle a responsibility of the operating system. It is one of the most important tasks of the operating system to make the system appear to the application processes as a stable, error-free execution mechanism . The operating system must thus be able to detect and repair the consequences of a failure of a hardware or a software component.

Many of the functions to detect, isolate, and restore a service are still implemented as a part of application software (the *maintenance* software). To find an anomaly and repair the consequences very often requires understanding of the application software activity, and thus is best implemented at the application level.

Information about the functioning of the exchange is gathered at the system and application level. For example, signalling processes and device drivers deal with alarms generated by the hardware they control, and as a rule every switching activity checks for illegal events and inconsistent conditions, often using counter variables to detect whether thresholds have been exceeded etc.

Maintenance software is often divided into two modules: (1) the supervision and detection module, (2) the testing module. The supervision and detection module gathers all the reports concerning an anomalous behaviour, filters the reports and decides to do one of the following :

- request execution of more tests so that more information is available,
- request the operating system to take a software or hardware module out of action,
- file the report for later use, eventually changing the status of a module.

The test module organises execution of tests on a periodic basis. It may be invoked by the supervision and detection module or it can be called into action by the operator. The results of tests are reported to the requester and filed for later use.

An advanced exchange can only realise its full potential and efficiency if a comprehensive *person-machine interface* (PMI) is available, enabling operations and management personnel to control the full range of exchange facilities. The interface must satisfy a large number of requirements:

- the exchange will be used by both experienced and inexperienced users and operators familiar with similar systems should be able to operate the new system after only a short training period -- different input and output modes must thus be supported;
- the interface must provide support to users of very different technical backgrounds and degrees of sophistication: e.g. planning and traffic administration staff, maintenance engineers, installation personnel and others will need to use the interface;
- the outputs must guide personnel to perform actions: these actions are often very repetitive so only relevant information in the right form should be made available, and where possible, graphical presentation of information is desirable;

- the interface should be adaptable to the different organisational structures of operational and maintenance activities: e.g. flexible routing of outputs according to their functional content and the origin of the message is required in order to allow an administration to freely assign exchanges and functions to Operation and Maintenance centres (in general the system supervisors, technical specialists and managers are located at different locations and the PMI must accommodate such requirements);
- the interaction language should be identical for all types of operations on all types of exchanges (small, large, terminating, transit, etc.) so that the operation, maintenance, and training activities can be efficiently performed; and
- the format of dialogue with the operator must support natural interaction: a bad dialogue format can substantially affect the speed and efficiency of the interactions and ultimately can cause confusion and hostility towards the equipment.

To satisfy these requirements CCITT has defined the syntax of a dialogue language [CCITT Rec. Z.302, Z.312-Z.315, Z.200] and of the man-machine dialogue (Z. 317). The semantics are left to the implementer.

The syntax of the language (called Man-Machine Language - MML) and the dialogue is specified with diagrams. Input can be from any device that produces the codes of the characters of CCITT International Alphabet No.5. and output can be to any device accepting such characters.

The dialogue can take place in two modes. In the direct dialogue mode, the operator uses a single command to input all the data required to activate a function: the system does not provide any help so this mode of operation is provided mainly for experienced operators. In the system supported dialogue (continuation) mode, the system specifies the data it requires for the execution of a particular function. This mode is intended for inexperienced personnel or for complex and infrequently used operations.

MML specifies a mechanism for access control, and devices and techniques to alert the operator to situations that command immediate attention.

MML is widely used by telephone operating agencies and switching manufacturers. It has become a widely applied standard and practically no system is produced without providing a MML conformant interface. MML is used not only for exchange control, but also for subscriber administration (Rec. Z.334), routing administration (Z.335), covering functions that

are in charge of routing a call attempt towards its destination, traffic measurements administration (Rec. Z.336) and network management administration to control the flow of traffic through a network of exchanges. It is also used in many other applications such as private branch exchanges, videotext systems, mobile radio systems etc.

The PMI is usually realised as a part of the operating system. Often a number of processes that are executed cyclically and in parallel process the user inputs, perform parsing and validation and the request their execution. The processes interact with device drivers that provide an interface independent of the physical characteristics of the device.

3.3. Network planning systems

The purpose of network planning is to select the best architecture, design and time period to meet the telecommunication needs of a network operator. In this context, architecture refers to the structure of the network and the function of its components. For example, the selection of an hierarchical versus a non-hierarchical structure and the routing capabilities of a transit exchange are typical architectural choices required in network planning. The design of the network refers to the selection of specific facilities and equipment quantities to meet current and future needs.

Digital technologies expand the number of options available to the network operator, and they also offer the possibility of integration of previously separate networks or network components. Significant economical advantages can be gained if the network options and alternatives are fully exploited. As the number of networks (voice, packet switched data, circuit switched data) increase, telecommunication networks change and integration of voice and data networks becomes reality, computer-aided planning systems will become more and more necessary.

Technically, the process of network planning is an optimisation problem in which the selected design satisfies certain minimal criteria (such as cost), provided that certain constraints are satisfied (such as maintaining a certain specified grade of service under particular levels of call congestion). In practice, planners must analyse the given network problem and extract the significant parameters before the optimization can be performed. To reduce the problem to manageable proportions and still maintain sufficient model accuracy requires both insight and experience in network design.

To solve a particular planning problem the following strategy can be followed

- Identify the planning needs and constraints. The following factors must be established:
  - planning horizon
  - accuracy requirements (economic impact)
  - present network configuration
  - network objectives (network size, function, growth potential, traffic patterns, degree of network control)
  - technical constraints (distance limits, propagation delay)
  - minimality criterion: economical assumptions with respect to maintenance, equipment, leasing and other costs are used in formulating the cost function of a design alternative

- Develop Models. Sufficiently accurate models of the desired network must be developed to assure confidence in the results. These models, together with the planner's experience are used to develop design alternatives and analyse their performance.

  A model consists of modelling elements (network elements) and a specification of the interrelation between modelling elements. Modelling elements, such as subscribers that generate and receive traffic, switching nodes that receive subscriber traffic and switch the connections to the destination subscriber, transmission nodes, such as lines, line sections etc., are the basic elements f the system to be modelled. The interrelationships between modelling elements are basically the connection between the elements. Each modelling element can be characterised in a way that is typical for its function. For example, subscribers can be characterised by traffic density in Erlangs and variance coefficients of the traffic they generate or receive, their geographical location and so on.

  Sometimes simple models suffice, but for complex networks complex models are necessary. The model is used to formulate the problem and the domain of design alternatives. Algorithms will operate on this model to find a design alternative that meets some optimum criterion.

- Select the best alternatives. The design alternatives are evaluated according to stated criteria and the best alternative is selected. The selection is in fact a multidimensional optimisation problem.

To support these activities software tools (Computer Aided Planning - CAP system) can be developed. A CAP system is usually organised in modules accessing databases. The databases and modules operating on it are administered by a (database) management system.

The database contains traffic and trunk data, transmission facilities and evolution data, switching data etc. and the modules can perform different types of projections and evaluation (traffic demand projection, trunk route evaluation etc.) The databases contain highly interrelated information, so database management has to be tightly co-ordinated. The database is also an interface between the tools. New functional modules can be thus added without affecting the operation of existing modules.

Based on this generic structure, different applications (data + programs) can be developed to serve specific planning problems. Two types of applications exist: optimisation and simulation

applications. Optimisation applications attempt to derive an optimal solution for a specific set of constraints. Simulation applications attempt to predict the behaviour of the system under dynamic circumstances. Each application provides to the user a language to specify the model data, planning/simulation rules, an algorithm to compute the desired result and an output facility to present the information in a form best suited for the specific problem.

The trunking model application, which computes the trunk loads in large networks, is an example of a optimisation application. It provides a selection of algorithms to specify traffic loads and various types of switching and trunking (analogue and digital). It can size networks in the case of hierarchical (fixed) or other types of routing. As output it provides for example the number of circuits required for the high usage (direct and intermediate) and final circuits.

Another typical optimisation problem handled by CAP systems is to compute an optimal location for an exchange. Frequently it is necessary to introduce a new local exchange to an existing network and the problem is to find where it should be located and whether existing exchanges should remain in operation. The input data required by this program consists of existing and projected numbers of subscriber lines, the number of exchanges and their characteristics, the cost of subscriber lines as a function of distance and the geography of the area (detailing obstacles like rivers and mountains).

For large networks, the computational requirements can be quite taxing and large computers are usually used (mainframes). The algorithms are often heuristic in nature so they give only a near-optimal solution. However the computational costs are feasible. CAP systems usually provide usually extensive output, often in graphical form.

Network simulation allows planners to anticipate network problems and devise and test solutions to alleviate the possible difficulties. Simulation is used to model the performance of a network subject to the failure of certain switching nodes or transmission routes. A program can for example compute a call congestion profile between nodes as a result of failure of some nodes.

Other simulation programs may provide simulation of novel routing strategies in a network. Complex routing strategies using decentralised management often require a computer analysis to estimate performance/cost.

3.4. Maintenance systems

Maintenance systems manage an overall *network* of telecommunication equipment to achieve a high quality of service for its users and maximum resource utilisation of the network components. They provide support for provisioning, installation, maintenance, operation and administration of telecommunication networks.

Maintenance systems can vary in size from very simple systems that manage a single network node, to distributed systems that manage complex networks consisting of equipment of differing ages and origin. Each node of such a distributed management system may be specialised to perform a dedicated task (such as a billing centre) or it may perform complete management for a geographical area.

The telecommunication network consists of many types of equipment, such as

- transmission equipment: terminal multiplexers, add/drop multiplexers, local cross-connects, cross-connects, special units related to the transmission technology employed,
- switching equipment: local and transit voice exchanges, data-packet switches, and
- associated equipment such as service computers, to provide non-standard services

The total management activities can be separated into the following five management functional areas [Principles for a Telecommunications Management network, CCITT M.3010, R.5/1992]: fault, configuration, accounting, performance and security management.

In practice, the following tasks have to be accomplished:

- telephone exchanges, trunk circuits, subscriber lines etc. must be routinely and remotely tested and diagnosed and faults must be isolated; detailed test results and other reports must be filed for later analysis;
- new versions of network equipment software must be downloaded;
- traffic measurements have to be activated both on a demand and schedule basis and collected data processed and summary reports generated; system status such as equipment occupancy, trunk route congestion, etc. must periodically logged for later processing;
- alarms caused by malfunctioning equipment or by circumstances that require attention must be displayed in a central location, and messages from network nodes have to be processed;
- subscriber service orders (assignment of directory number, service class, scheduling of execution of services, etc.) and complaints have to be processed, subscriber line testing in answer to complaints must be initiated, subscriber lines must be barred for access etc.;

subscriber records must be updated and various statistics regarding subscription should be compiled;

- the overall network status, such as congestion and overload have to be displayed, and detailed information should be obtainable from dedicated displays; traffic diversion, route restriction, destination restriction, originating and incoming call restriction have to be performed; the occurrence and consequence of such activities must be filed and processed into statistics;
- charging data has to be collected from each exchange through data links on a scheduled basis and bill processing and statistical processing must be performed.

Such widely varying tasks are performed by a network of management centres, some of them specialised test centres such as the Operation and Maintenance Centre, Line Test and Subscriber Service Centre , Transmission Test Centre, Service Order Centre, Network Management Centre, and Billing Centre. In each of these functional units, the activities are supported by computerised equipment with similar or even identical system architecture and differing application software.

The implementation of the maintenance system must take into account the fact hat almost all network operators already have an extensive infrastructure consisting of operation systems, networks and telecommunication equipment. Thus interworking with existing systems is of crucial importance. The maintenance system also provides information that will be later evaluated by personnel and equipment that can vary widely in its function, experience and needs. The information can be evaluated by humans, electronic devices or other computers. Thus interfaces to equipment such as video-walls, post-processing systems, report generation systems, expert systems etc. must be available.

For such systems, flexibility is of central importance and standards are the key to achieve it. Intensive studies in the last few years appear to have finally resulted in workable tools for the description and management of networks. The methodology and the driving force behind the definition of standardised interfaces (both communication protocols and the information model) are laid out in CCITT Rec. M.3010. M.3020 identifies the management services that represent the management tasks and describes the methodology to be used in the specification of interfaces. M.3400 contains the list of telecommunication network management functions for different functional areas. M.3100 provides a generic network information model.

Basically, networks and their management are described in a layered hierarchical manner. All the activities that take place in a network can be allocated to various networks, such as the physical

transmission network, the logical transport network, the intelligent services network and the telecommunications management network. Resources are represented as Managed Objects in the information model and a management action corresponds to manipulation of an MO.

In the implementation of management systems, hardware, system software, communication protocols and users interfaces are all based on international standards. The hardware that is being used is usually an industry standard workstation/server, the operating system is usually UNIX, databases are SQL-compatible, communication is based on CCITT, OSI, Internet , SNA and DECnet protocols, and application and user interfaces conform to X/Open recommendations. The applications are usually programmed in a language that has a good level of portability (C, C++) and the user interface is highly graphical.

For example, PHAMOS[4] is based on HP9000 Series 8x7 computer systems, the UNIX operating system is XPG3 compatible and the Q-interface is used for the communication between Operations System and Network elements, and X.200, 400, 500, 600 and 700. FTAM, NFS and LU 6.2 are supported. OSF/Motif and X-Windows for user interfaces and Ingres databases (SQL compatible) are used.
The PHAMOS architecture is illustrated below:

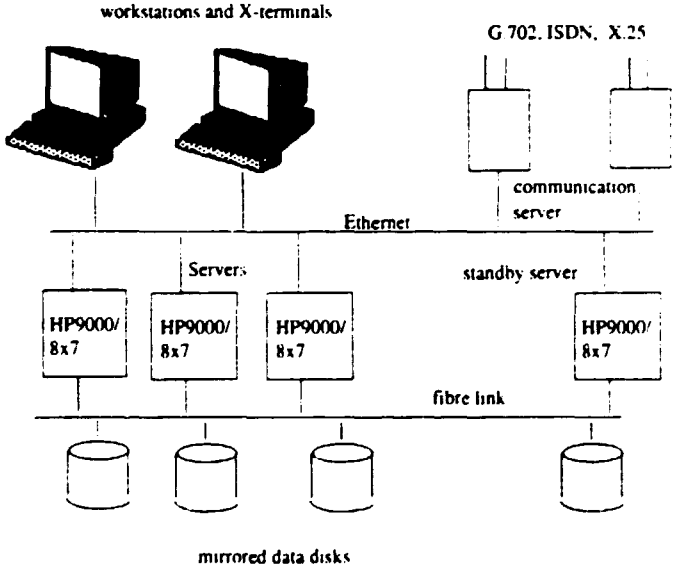

Figure 12: Phamos architecture

[4]Hermann, A. et all.: 'Phamos - Philips Advanced Management and Operations System - Functionality and architecture', Philips Telecommunication Review, Vol. 51, N0.1

The software architectureis based on an object-oriented hierarchical architecture that is strongly client-server based.

3.3. Billing and accounting packages

In Western countries, communication costs represent approximately 4% of a company's operating expenses[5] and they keep growing at approximately 9% each year. For most companies these are considerable costs, but until now companies could not obtain the data needed to manage them.

Telecommunication users have defined quality billing to include accuracy, flexibility, timeliness, and the ability to receive the data in an automated format for further analysis. Unfortunately, there are too few tools to implement such services. While bills have always been detailed and explicit, it was often difficult to associate the cots with the group of users incurring the cost and the specific task under which the costs were incurred. In short, the bill was designed for the service provider and not the subscriber.

Billing information systems should enable customers to review communication expenditure and usage, consider future needs and adjust their telecommunication profile to their business needs. Billing systems must be able to combine data for all services provided. Currently most billing reflects the physical architecture and sage of the network instead the logical communication usage.

Billing should also provide information about actual performance delivered by the service provider. It should include information about busy hour reporting, blocked call rate etc. In this respect the billing and network management services interact strongly with network management systems.

The billing system is important in connection with customers queries about service requests and complaints and requires an on-line system to provide up-to-date information to customer service personnel.

---

[5]'Building a Financial Management Structure for Communications', Business Communication Review, August. 1989

Fortunately, the capabilities of technology makes it possible provide system support for the capture of billing data of the kind that is required by users. Basically two strategies are being used: (1) the service provider develops a billing system that is able to cater to all the specific needs of its customer, and (2) the service provider captures the raw billing data and hands it over to the customer. The service provider helps the customer to develop tools necessary to analyse its data, but often industry-standard PC based tools can be successfully used.

Itemised billing, consolidated report packages and multiple billing hierarchies, invoicing via magnetic tape, compact disc, or electronic data interchange are all attributes of this new approach.

A billing system that is able to provide modern billing services is basically a database application. It provides retrieval and update operations on a (usually very large) database. The billing data may be located on several computer systems and to provide a consolidated bill complex reconciliation procedures have to be executed. These consist of a batch-processing facility that deals with the capture of data, the production of bills, non-payment reminders, and on-line facilities that allow real-time access to the databases. These are used by customer service centres while answering customer enquiries. In some billing systems, the customer is also able to get on-line access to its own billing-related data The security and integrity control of such systems is of prime importance.

As an example, the billing system of British Telecom (Customer Service System - CCS) runs on IBM 3900s, uses MVS and is written in the COBOL programming language, using the IDMS database system. On-line transactions are serviced by CICS TP. CCS is logically a single system, but in order to accommodate the size of the operations, 29 separate images of the system are running nationally at different geographical locations. These individual systems are connected together for on-line access.

CCS supports 1400 gigabytes of data, 14 million screen exchanges, 60,000 connected terminals and has an average response of 2 seconds. It contains 5 million COBOL statements and more than 3,000 different screen presentations..

In 1991, BT's CCS produced 100 million telephony bills. It serviced 20,000 regular on-line users and an additional 20,000 users used it occasionally. The system was developed in the 1980s and due to its complexity was plagued by delays. Its integrate database provides many

benefits, such as rapid access to all data, but it also has its limitations. The size and complexity of the databases and applications operating on it make change very time consuming and complex. The type of database and the way the customers data is used does not allow easy reorganisation of the customer's data to reflect their organisational changes.

4. What is so special about TCS?

4.1. Introduction

In the previous chapters we have described the different types of telecommunication software. The first characteristics that makes the TCS special is its wide range of functionality. In fact, every type of software known to information technology industry finds its application. For example, software for

- *real-time highly-available , distributed* systems (operation and maintenance),
- *highly distributed control system* (network management),
- *(on-line) database* (billing and accounting),
- *number crunching* (network planning) etc.

is used.

These functionally very different systems have to interact almost continuously, resulting in a very complex system in which correct functioning of each subsystem is necessary. A telecommunication system is sometimes portrayed as the most complex system ever built by man, and correspondingly, telecommunication software as the most complex and extensive software ever designed.

Software risks

Many risks and difficulties encountered in developing telecommunication software are simply due to the fact that *software* is being developed. For example[6], 'inadequate personnel ... starting a major software project with inexperienced and inadequately trained personnel is a sure route to disaster , 'mistaken user needs ... many software projects have failed because the real user needs have not been captured adequately', 'unmastered complexity ... Software systems are complicated because of the inherent complexity of the problem domain and because of the extrinsic, needless complexity of the software design. This extrinsic complexity must be eliminated through the use of proper design methods ... Complicated software is expensive to build, difficult to verify and unreliable in its operation'.

In the remainder of this chapter we will concentrate on the operation and maintenance software of an exchange (SPC - Stored Program Control - software), mainly because a telephone

---

[6]H. Kopetz: 'Chances and Issues in Software Production in Developing Countries', UNIDO IPCT.144 (SPEC)

exchange is the central feature of a telecommunication network and operation and maintenance software is the basis of the exchange.

Real-time software

As already described in Chapter 3, the SPC software must operate within strict timing constraints and under conditions where a variety of faults may occur at any time. It is thus an example of software for real-time highly-available systems. For designing such systems, general software techniques have to be extended and adapted to support the following:

- specification and verification of real-time constraints

  The usual approach for the specification of computing systems is to describe the actions in which the system participates and the order in which these actions can take place. For real-time systems a technique must be added to describe the physical time constraints placed on the order of the actions. For example, the maximal or minimal duration of an action, the frequency of actions etc. must be specified. Once a real-time system has been specified and developed it must be demonstrated that not only the functionality but also time constrains formulated in the specification are realised. Formal proof and testing methods for conventional systems must be extended so that the temporal properties can be verified and tested.
- specification and verification of fault-tolerance

  A real-time system and its environment form a synergistic pair. The computer must provide services at an acceptable quality level even in presence of hardware and software errors, or it is perceived as having failed. The 'acceptable quality level' must be specified, implemented and validated.
- programming languages and design methodologies;

  Conventional programming languages do not provide the necessary support for the expression of timing constructs or for specific services that arise in the development of such systems, such as support for distributed programs, fault-tolerance constructs (such as exception handling) etc.
- operating systems;

  A real-time operating system must provide support for guaranteeing real-time constraints, supporting fault-tolerance and distribution etc.
- architectures

A real time system can be viewed as a network of subsystems where each node acts as a three-stage pipeline[7]: data acquisition, data processing and output to sensors and displays. In such systems the movement of data is of crucial importance. Thus the high I/O, the interconnection mechanism in nodes and in the network and fast, reliable and time-bounded communication mechanisms are usually different from conventional systems.

SPC software

To specify, implement and validate the common control software these characteristics of real-time highly-available systems have to be taken into account. In addition to the generic properties, common control software notable for

- the complexity of the software,
- its consequent size,
- the long working life required,
- the need for real-time operation,
- the stringent reliability and availability required,
- portability of software.

These properties influence the way in which telecommunication software is developed, tested and maintained. These issues are discussed in the following chapters.

4.2. Size and complexity of the software

In general, SPC software is complex with respect to :

- *the number of functions* that have to be provided. The complexity of SPC software is a reflection of the complexity of a telephone exchange: there is a multiplicity of (1) units connected to an exchange, (2) units within the exchange (3) requirements for services. There can be tens of thousands of subscriber lines connected to the exchange, each line may have dozens of attributes reflecting its service needs, hundreds of trunks for which there may be a great variety of signalling modes, dozens of data lines for maintenance and operation purposes etc. An exchange may consist of hundreds of processing units, dedicated hardware units and dozens of computer peripherals. For a modern telephone exchange the generic functional requirements are often contained in multi-volume sets.

---

[7]Krishna, C.M., Shin, K.G., and Bhandari, I.S.: "Processor Trade-offs in Distributed Real-Time Systems", IEEE Trans. Computers, Vol. C-36, No.9, September 1987

The non-functional requirements can be also very extensive. The operating agency can formulate additional requirements with regard to the administration of its exchanges, the collection of charging information etc.

- large *volumes of data* that have to be manipulated by the exchange: in general, data structures describing the exchange equipment and services, subscribers status etc. are complex.

Because the software is complex, the development of SPC software is very challenging. To take account of the complexities of SPC software, (1) the choices related to the system and software architecture, (2) the organisation of the development activities, and (3) the software techniques and tools employed have a strong effect on the achievement on the overall quality and the effectiveness of the development effort and these are distinguishing characteristics of telecommunication software.

4.2.1. The consistency of system and software architecture.

There are many ways in which a system can be distributed over its components. As we have already indicated there is no 'best way' to distribute the functions. Different distributions are possible and valid. However, the absence of a well-defined and clear architecture leads almost inevitably to development problems, friction between different development groups and ultimately to redesign of the system

The software architecture has to describe how the total software is distributed over its components and how these components are related. In particular the following must be defined:

- the partitioning of the software into modules structured into hierarchically layered systems with each layer running on a lower level of abstraction and implementing a higher level (of abstraction). The functionality of each layer and its modules must be clearly described.
- The locus of control has to be established. Usually it is centred in processes and message passing between processes as the preferred method of achieving functional separation and synchronisation; however other alternatives are feasible.
- the interfaces between modules must be clearly defined.
  There needs to be a clear separation of module implementation from module interfaces.
  Real-time intensive functions must be contained within well-defined processes or modules so that code optimisation activity can be focused and processor power can be appropriately directed.

- Logical software functions must be de-coupled from the features of physical hardware units so that it is possible to re-implement the hardware of the system in order to take advantage of technology advances without disrupting the software architecture.

4.2.2. Organisation of the development activities

To develop SPC software, the involvement of (several) hundred people, perhaps at different sites or in different countries over a period of several years is often necessary[8]. Good organisation is characterised by a good division of responsibilities and co-ordination of the activities so that the user's real requirement is implemented, the system and software architecture is not compromised during the development and all work can progress at the maximum possible speed. The overheads related to good organisation and enforcement of sound engineering practices are substantial, and this tempts almost everybody to economise on organisation and procedures, typically to overcome delays by postponing to tomorrow tasks that should be done today. The consequence of these sins is a substantial degradation in system and software quality.

An example of a good development organisation for a switching system development is taken from [9]:

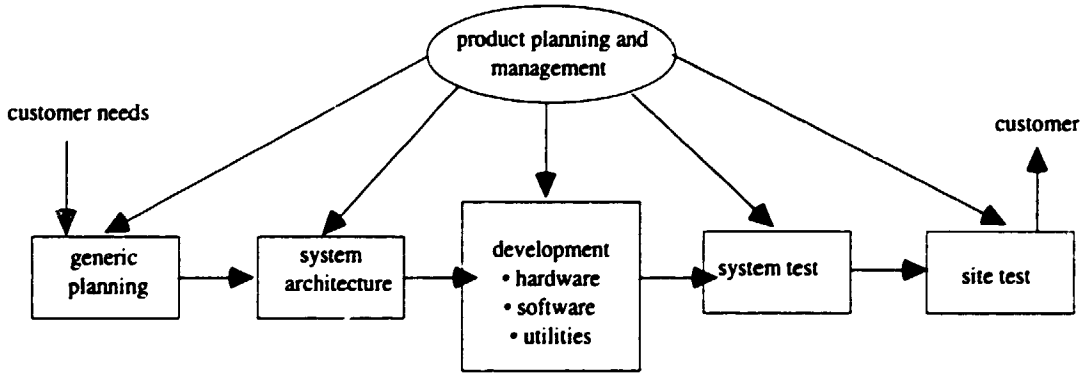

Figure 13: Structure of development organisation

The first step in the development consists of determining the customer's needs (list of features) and based on cost-benefit analysis deciding how these should be met. The result is a package of requirements sometimes called 'generic'.

---

[8]Goecke, D., Haake, G.: "A Software engineering approach applied to the complete design and production process of large communication system software", Session 13 C, Paper 1, ISS 84 Florence

[9]Bosco,H.L., Carney, D.L., Cicon, J.P. and Prell, E.M.: "Managing a very large software switching project- NO. 5 ESS",

There has to be a centralised architecture organisation that consists of a system architecture group and several planning groups. Each of the major development organisations (hardware, software, development support tools and utilities) has an internal planning group that co-ordinates and helps resolve internal design issues and represents their organisation to the system architecture group. The system architecture is responsible for co-ordinating high level requirements/design issues that have potential impact on the planned evolution of the system architecture and for ensuring that system requirements are met.

The sharing of architectural responsibilities is essential for focusing responsibility, maintaining control, obtaining developer acceptance and eliminating a serial path between system architecture and development. Acceptance and support by the developer organisations is ensured by having planning functions within their line organisations.

These effort may be supplemented by real-time modelling, methodology, training and quality groups/functions.

One decision that has to be made early in the project is to resist pressures to shorten the time spent on initial requirements and high-level design work.

Before the work on detailed requirements starts, teams from design organisations, planning groups and the architecture group have to establish high-level requirements and partition the total job. Software has to be split into modules that can be developed separately and for each module a person must be made responsible for its development .

The test organisation is separate from development and has the task of integrating the system. The process is a highly interactive but logically serial process and can take place while development of the modules takes place.

The final phase is site testing, and this is performed by a different organisational entity. The testing is performed on the customer's premises and it emphasises the external interfaces of the tested function.

The management styles must take into account the fact that the software industry is a know-how intensive industry employing professionals who as a rule are highly educated, qualified, independent and, if properly managed, easily to motivate. In decision taking, a certain level of

democracy with clear understanding of responsibility is necessary. An ethos must be cultivated in which people are not afraid of mistakes or potential schedule slips but view them as problems requiring resolution.

4.2.3. The software techniques and tools employed.

The software design usually starts with a comprehensive set of specifications based on world-wide study of telephone exchange requirements, including both basic and optional functions and facilities. Often, particular effort is made to catalogue all major existing signalling systems.

SDL

The specification is often written in CCITT's Specification and Design Language (SDL). SDL is a formalised method of presentation of the required behaviour and the internal processes necessary to implement the specification - the actual behaviour - in telecommunication systems. SDL was first specified by CCITT in 1976 to provide a graphical means of specifying the behaviour (known as SDL/GR). It was to be used as a software development tool and as a high-level documentation technique. In the early 1980s, in addition to SDL/GR an equivalent program-like non-graphical version of SDL, known as SDL/PR was also developed by CCITT[10].

SDL is well suited to a top-down design technique. It is oriented to demonstrate and solve problems related to process interactions in switching systems. Many automatic tools are available for creation, storage, updating and manipulation of SDL specifications[11]. Text books describing how the language can actually be used are also available[12].

Exchange functions are derived from these specifications and arranged hierarchically. The functions are broken into sub-functions, then these sub-functions are further divided until a basic set of functions has been identified, each of which can be performed by a single software module.

---

[10]CCITT Rec. Z.100: Specification and Description Language SDL, Blue Book, Volume X.1 (1988), ITU

[11]Faergemand, O., Reed, R. eds.: "SDL'91 Evolving Methods", Elsevier Science Publishers, ISBN: 0 444 88976 0, p.511-520

[12]Saracco, R., Smith, J. R. W., Reed, R.: "Telecommunications Systems Engineering using SDL", Elsevier Science Publishers, ISBN: 0 444 88984 4

A functional specification is performed for each of these modules. Interfaces to other modules are specified by listing the messages crossing the interface. The dynamic behaviour of the modules is documented using scenarios representing the information flow between the various modules engaged in a transaction, such as setting up a call or writing a file into a mass storage system.

CHILL

The design of each software module is based on its functional and operational scenario. The modules are coded and tested largely independently of other modules. The completed modules constitute a library which can be used for the production of exchange software. The language used to code the modules nowadays is usually a high level language such as C or CHILL. CHILL - the CCITT HIgh Level Language - is the language for programming SPC switching systems. Its requirements includes suitability for programming the following applications: call handling, test and maintenance, operating systems, on-line and off-line support, implementation of Man-Machine Language and acceptance testing. There are several CCITT publications on CHILL[13],[14],[15].

As a CCITT-standardised language, CHILL is very different in nature from SDL and MML. Whereas the latter are of chief interest to the telephone operating agencies and represent a mandatory requirement in the clause of contracts placed by manufacturers of switching equipment, the same does not apply to CHILL. The decision to use CHILL or not is basically a matter for the switching equipment manufacturer.

A number of switching manufacturers have adopted CHILL as their own programming language (Siemens, Alcatel, NTT[16], and Korean, Indian, and Brazilian manufacturers) but other languages, such as C are also used (e.g. by AT&T in the programming of ESS 5[17]). Initially, C was a general programming language designed for use in programming a time-sharing system in the Computer Department of Bell Labs. Nowadays it is widely used, particularly in conjuncture with the UNIX operating systemin applications ranging from text-processing to communication,

---

[13]CCITT Rec. Z.200, CHILL Language Definition, Blue Book (1989), Fasc. X.6.

[14]Introduction to CHILL

[15]Formal Definition of CHILL,

[16]Maruyama, K., Sato, N. and Konishi, K.: "NTT CHILL implementation aspects and its application experience", pp 191-195

[17]The UNIX System, a special issue of the AT&T-Bell Laboratories Technical Journal, Oct. 1984

process control and artificial intelligence. It is our impression that interest in developing languages specifically for programming of SPC exchanges is declining. General purpose programming languages, or producer-specific languages, have not disappeared and every manufacturer keeps to its adopted programming language.

The programming activities must be supported by a sufficient quality and quantity of tools. Editors, debuggers, compilers, configuration management tools, file management, and product-building tools, document processing facilities, communication networks, electronic mail etc. should be provided to all designers without complex request and evaluation procedures. The development environment is usually centred around a local area network to which workstations and minicomputers are connected. Using gateways, access to the target hardware can be obtained so the developed software can be loaded and tested on the target hardware. Other gateways provide access to external communication networks and thus external facilities.

As a rule, each developer has access to his/hers personal workstation and a number of minicomputers/servers are shared. Writing of specifications, programs and documentation takes usually place using the resources of the workstation. Compiling, archiving, building-up of the system usually takes place at the provided servers. Industry-standard workstations, servers, networks and communication gateways, operating systems, databases, and communication facilities are commonly used.

The techniques used in developing software - the software engineering techniques - do not make any unusual demands. Therefore, methods and planning tools adequate for normal process control software also apply to the development of telecommunication software. Special attention should be paid to documentation systems and planning.

Each document produced, from specification and design documents to user manuals, should be carefully conceived according to a well defined standard. For each document, the organisation in charge of writing, updating, checking and supervising it must be clearly defined. A hierarchical layered documentation plan has to be established at the beginning of the development project so that all produced documents can be archived and effective search and update procedures can be implemented.

A computer supported archiving system, implemented using industry standard minicomputers or workstations, running UNIX as operating system and having sufficient secondary memory, is necessary. Special attention should be paid to making regular and frequent back-ups and

storing these on a safe and physically distant location. Each member of the development team should have in principal on-line access to this electronic archive. The access, change and other prerogatives should be carefully defined and regularly checked.

The complexity of SPC software is so high that it is practically impossible to remove all design or implementation errors. In testing it is impossible to simulate every conceivable situation that an exchange may face, and this means that the best possible approach is to provide software that is tested against the most frequently occurring (and known) circumstances. Then the software is put in the field, in a normal environment, and made to work while repairing the residual errors. Present experience is that in the initial phase of a system's operation, it is possible to eliminate more than 95% of software faults through normal testing and diagnosis procedures.

4.4. Long working life required

In 1976 it was asserted[18] that the software for switching systems should have a life of up to 40 years. However, advances in technology may make a substantial reduction in this extremely long period. Even so, it can be expected that parts of telecommunication software will remain in use for at least 20 years. The life expectancy is quite exceptional compared with that of software for other major systems (where it does not exceed 10 - 15 years).

During the life of the switching system, the software is continually being developed and expanded due to

- the introduction of new services,
- the emergence of new requirements, in particular in respect to
- replacement of hardware components by higher performance components

One of the main design objectives for a modern stored program control switching system is to make the software future proof. Rapid advances in computer hardware technology, together with decreasing prices of VLSI make it necessary during the life time of an exchange to replace components such as memory chips and microprocessors with less expensive and more powerful units as they become available. Similarly, it becomes necessary and desirable to replace telephonic hardware with more advanced units.

---

[18]Bjurel, B.: "keynote address, ISS76, Kyoto

To be future proof, therefore, software must be as independent as possible of both the architecture and the hardware of the switching system, so that it can accommodate additions and changes to the hardware with little or no modification. Also it must be structured to permit the smooth introduction of new features with minimum design effort.

These design objectives are usually achieved by applying the following structuring concepts:

- The software is built-up in a layered manner. Each layer provides a set of well-defined services to the next layer. The services provided by one layer define an abstract, virtual machine.
- Software is structured in small, almost independent modules, with simple, standard procedures for intercommunication.
- Functions are grouped in a way that takes advantage of the characteristics of the virtual machineso that the points at which the software interacts directly with hardware are isolated from the software that controls exchange operation; as result, a change in hardware can affect only a small area of software (the device handler)
- Interfaces are generic rather than specific. This makes it possible not only, for example, to interwork with a number of signalling systems, but also to cater today for applications that will be provided in the future by incorporating in the software all the 'hooks' that may be required in anticipated applications. This effectively minimises the need to modify existing modules when a new facility is implemented.

4.5. Real-time operation

SPC software implements services for which the response times are of the utmost importance. For example the following maximum response times have to be realised by SPC software:

for signalling activities	10...100 ms
call processing activities	100...1000 ms
person-machine	1...3 s
O&M transactions	1...10 s

Switching software has to deal not only with each call in isolation, but with a large number of simultaneous activities that all require these response times. For a local exchange with 10,000 subscribers there can be the many simultaneous activities taking place: for example,

- 1000 calls in conversation
- 200 calls in set up/clear phase
- 20 Operation and Maintenance transactions

To meet these performance requirements, the specification language must provide facilities to state the temporal requirements. The programming language must support operations that will guarantee certain temporal behaviour and the operating system must be able to schedule activities on the basis of their temporal requirements [19].

SPC software has to interact intensively with dedicated hardware, and for a high degree of parallelism, multiprogramming and multiprocessing must be employed. The software must be predictable in its temporal behaviour, which implies a certain simplicity and perspicuity of structure. It is performance sensitive, and so the executable code must be efficient .

In real-timesystems more than 50% of the resources required for the development are spent on testing. Testability is thus an important criterion for the selection of the system and software architecture. Modularity of design, and clear and well-defined interfaces are a prerequisite for a testable design.

4.6. Stringent reliability and availability requirements

Conventional systems may be stopped at certain times without serious consequences. There may be regular interruptions for maintenance, or interruptions for unexpected reasons. For telephone exchange the requirements are totally different. Failure of a telephone exchange for a period of even 15 minutes is so rare that it is reported in the newspapers and the operating company is questioned about the causes of the interruption. The availability standards are thus very stringent.

The following availability figures are CCITT standards[20]
Unavailability :

- of the entire system $< 1.5*10^{-5}$
- of a subscriber line $< 10^{-4}$
- of an inter-exchange circuit $< 10^{-4}$
- for emergency calls $< 1.5*10^{-5}$

---

[19]H. Kopetz: 'Chances and Issues in Software Production in Developing Countries', UNIDO IPCT.144 (SPEC)

[20]CCITT Handbook "Economic and Technical Aspects of the Choice of a telephone Switching Systems", ITU publication, Geneva

- for basic telephone service $< 10^{-4}$
- for supplementary telephone service $< 10^{-3}$
- for charging $< 10^{-4}$
- for traffic measurements $< 10^{-3}$
- for administrative operations $< 10^{-2}$

To satisfy such rigorous requirements, a whole series of design arrangements are necessary:

- hardware units must be replicated. Depending on the function of the unit, some units are duplicated, triplicated or even quadruplicated.
- the functioning of the hardware and software components must be guarded so that faults can be detected and the faulty unit taken out of operation.
- in dealing with unpredictable situations the software must not crash but must be able to overcome any adverse condition, possibly losing calls already in conversation, or as last resort via an automatic resumption of the entire exchange, and with or without reloading of the software.
- the maintenance procedures must be defined in such a way that no interruption of normal service is necessary. A partial degradation of service is acceptable, but the system must be able to provide normal services even when the complete software of the exchange is being reloaded.

5. Prerequisites for entering the market

Organisation

An important component of the organisation of a new supplier is technologicai -- the tools and computers needed -- but an even more important component is skilled and trained people and the technical and management structures in which they can co-operate o produce high quality software. This requires the adoption (and often the development) of new methods for individual and group work with a variety of different communication modes. Software design at this level of complexity is capable of attracting highly motivated people and one of the tasks of a good management is to be able to stimulate creativity while monitoring progress and ensuring that targets are met. During the system design stage, creativity is important and can often require only limited management; later, responsive and sensitive management is required to convert these achievements into prototype and products.

If a new supplier has no experience of large-scale software development, these skills should be slowly and systematically built. Design is very stimulating and challenging, and management must understand the complexities of such system design and the demands it places on the designers.

Prerequisite: a creative environment for skilled and motivated university educated personnel, providing contact between industry and university, and able to draw on a technological tradition. The organisation must have available a high degree of technological support in the form of computers and software tools. To successfully design, manufacture, market and maintain a system is not only a matter of personnel and hardware and software resources: to a high degree it is also a matter of the optimum use of the resources during different phases of system's life.

Thus an organisational culture is needed in which individuals can work together in an efficient and purposeful way. This places demands on good communication and co-ordination among technical personnel, and between management and technical personnel.

There are many values of such a culture:

1) continuity of knowledge, continued development (even a 'finished' system is never done), transfer of knowledge and experience between individuals, and in documentation, and technical continuity in terms of work practices and work ethos;

2) development of both the organisation and the individual, management of the development of creative individuals, management of production-oriented people for more group work, and development of work standards;
3) encouraging responsibility: for technical challenges and for the market and the end-user,
4) to be able to design individual components and integrate them into a system, and on time, making the best use of people with different skills in different phases of the work.

The phases consist of the technique- and technology-oriented activities (basic development, system concept definition, component design, choice of methods and design aids) as well as organisation- and production-oriented activities that require individuality and team work.

Supporting Information Technology

The development of new equipment of this degree of complexity requires the use of sophisticated high-technology products that are capable of supporting design and production activities.

Prerequisite: Research and development centres where feasibility studies can be made, basic designs proposed and evaluated, and basic technologies developed (e.g. real-time distributed operating systems).

These centres can be set up in co-operation with manufacturers and service providers, perhaps initially with staff seconded for this work but eventually able to make staff available to migrate to manufacturers, service providers and universities in order o transfer knowledge and technology.

The centres must be provided with equipment for design and analysis: e.g. workstations able to support both specialised software such as graphical versions of SDL as well as more standard packages for configuration, change and software integration management. Industry-standard equipment is essential, to enable each centre to draw on a wide range of internationally available software developed to common standard.

For example, a small research centre for the development of different levels of switching systems may have a staff of 20-50 design engineers working on switching software alone, supported by 10-20 workstations connected by a LAN and connected in turn to external networks such as Internet through gateways. Such a centre must have tools to support software design activities as well as software management. Procedures must be established for component specification, design and testing, and subsystem and system integration and testing,

with version and change control enforced to keep track of development versions and to build up software design and implementation documentation for later use during productionisation. Where specialised hardware is required (e.g. for interfaces), this must be available for on-line use with test and monitoring software.

Prerequisite: standardisation of development methods, concentrating on CCITT standards, using SDL, CHILLand MMLas the basis for the development of tools for specification and implementation on industry standards development platforms (UNIX workstations, PCs, UNIX Servers, Ethernet LANs, graphical tools with industry standard interfaces X-Windows, and conformance to X-Open's XPG4 where possible).

It must be expected that considerable effort will be required to support such tools, and to adapt them to particular requirements. This system support activity must be done in-house, in order to preserve independence from particular manufacturers or suppliers.

Given the extent of development that is to be undertaken, and the time scale in which it is to be done, all possibilities for sharing development with other organisations should be explored. For example, it may be possible to use software components developed in other countries involved in similar telecommunications system development. This is possible provided the software is designed to international standards such as Signalling System No.7 and developed for execution on a generic platform. And whether or not the development effort is shared across different organisations, conformance to standards is highly desirable and makes it possible for conformance testing to be standardised.

Prerequisite: conformance testing facilities, including hardware and software, for shared and local development of software components.

Entering the Market

The development of SPC technology requires substantial capital investment and a long period of experience in the field. Because of this, the international telecommunications market tends to be dominated by a small number of major suppliers. Within any country, it is unusual to find more than one such supplier (e.g. in the UK., market forces have resulted in the switching system interests of several major manufacturing companies being merged into one large company), and the considerable development costs involved may be partly offset by direct or indirect government support.

Prerequisite: switching system development requires a concentration of technology. If the technology is available but is dispersed in different organisations, realignment of these technical skills is necessary to ensure that they are concentrated in a single organisation. But this does not require a merger of computer and communication companies into the large information utilities that were predicted in the past. In the isolated instances where this direction was followed (e.g. IBM's acquisition of ROLM, and AT&T's venture into computer design) it was not accompanied by much success. Both industries have a long tradition and very different market perceptions, and this makes it hard for a combined industry to preserve economic viability and market focus. Nevertheless, countries with a significant telecommunication industry also have a computer industry and developing countries have usually promoted both industries (e.g. Brazil, Korea, Taiwan, India).

For historical reasons, and sometimes to provide an economic barrier, many countries have specific local telecommunication standardsand operating procedures and these may make it difficult for foreign suppliers to enter a market, thereby providing some protection to local suppliers. But this is of limited long term value as the major suppliers usually design their equipment to make it possible to accommodate local requirements (e.g. different signalling protocols) without requiring changes to the overall software architecture. Moreover, international telecommunication standards are designed to ensure that global telecommunication services can operate without difficulty across national barriers and local suppliers who do not conform to these standards will eventually be limited to a very small segment of even their local market.

Prerequisite: Conformance to international switching standards (e.g. Signalling System No.7, ISDN) is important to obtain components and know-how at competitive rates and to take advantage of world-wide design experience.

Thus apart from the need for conformity, there are many positive reasons for local suppliers to adopt international standards:

- they represent many decades of experience of different national agencies,
- they provide a good starting point for the design of new equipment, and
- they make it possible for a local supplier to enter the market by undertaking system integration using local equipment complemented by equipment purchased from a competitive international market.
- Recent developments in switching and communication technology, and in subscriber equipment, are likely to have far-reaching effects on global telecommunications services

and it is important for local suppliers to have a sufficient degree of compatibility with international standards for them to participate in and make use of these new developments.

National and international providers of telecommunication services are generally required to purchase equipment from suppliers with proven experience, and they require a long-term commitment for support right through the life of an exchange (10-20 years). This makes it difficult for a new supplier to enter the market for large local and trunk exchanges and the associated administrative processing equipment. Moreover, the entry cost at this level is so substantial as to be uneconomical for any supplier who does not have relatively sound assurances of a large local market. But there are ways in which local suppliers can enter the market and some of these are listed below.

- To gain experience of the design and manufacture of telecommunications equipment, a new supplier can begin with the development of in-house equipment (e.g. PABX's) that have low capacity but which are required in large volumes and then move on to small exchanges. For example, Tropico of Brazil first developed small exchanges (Tropico R for up to 4000 subscribers, and the largest version Tropico RA with 16000 lines) and then undertook the development of larger versions at the Telebras Research and Development Centre with the active co-operation of the Brazilian switching industry and Telebras Operating Companies. At Campinas (where there is also a well-known university), there is a specialised centre with 1500 highly qualified professionals, making it one of the most important R&D centres in Brazil.

- Another viable entry point is to develop exchanges for specialised requirements, e.g. rural branch exchanges for low capacity use by highly dispersed subscribers using a mix of terrestrial and radio/satellite links. The Indian Centre for the Development of Telematics (CDOT) has gone on from the development and commercialisation of PABX's to rural remote switches for zonal switching centres. At the same time, they have an active programme for the development of large switches for eventual use in urban areas

- Experience of the very important service aspect of the telecommunication industry can be acquired by providing software, hardware and systems support for equipment supplied by other vendors, and this can be used later to set up support operations for products of local manufacture.

Many other alternatives are possible and it may well be appropriate for a new organisation to undertake work on a variety of such entry-level options in order to build up a breadth of experience.

Prerequisite: There must be an integrated framework for systematic development, starting with subsystems and smaller exchanges and including design, manufacture and system support. It must have long-term political support and be the responsibility of a high-level government policy- making body headed by a senior technical administrator (Technology Secretary). It must also be endowed with substantial funds and will require a 15 year investment horizon before any profits are visible.

Education

For the development of new SPC equipment a crucial requirement to be met is the education and training of manpower at different levels and with different skills. With the rapid developments in technology, such education must be viewed with long-range objectives so that trained staff are able to keep up with emerging technologies. This is particularly important in telecommunications as techniques from many different fields are rapidly being integrated into systems in order to solve problems or to provide new kinds of services (e.g. fields as wide apart as image processing, database technology and process algebra, to take just three examples, can all be seen to have a role).

Unfortunately, there is relatively little by way of published information (either as detailed articles or as text books) on the actual design of SPC equipment, even if most of the basic techniques in use are well known. This knowledge gap makes it necessary for a new supplier to undertake learning exercises at different levels -- design, implementation, production, operation, support and maintenance -- and to plan these for staff with different skills and backgrounds.

The first requirement is for university level graduates with qualifications and knowledge in areas such as communication, telecommunication and computer science who will have the depth of knowledge required to understand modern SPC design techniques and the ability to contribute their own specialised knowledge. People with such backgrounds have the added advantage of not facing 'knowledge obsolescence' in their future career. But in many cases, t[illegible]ing will have to be supplemented by specialised courses and followed up by subs[illegible]ed courses.

Prerequisite: A wide range of educational material must be introduced into conventional tertiary education courses in computer science and engineering. This must be complemented by more specialised short and long term courses targeted at new entrants, and for retraining those already in the industry.

In addition, a new supplier will need people with a range of skills in design, manufacture and management. Their knowledge and trainingneeds can be met in different ways and a suitable choice will depend both on the product strategy chosen and the level of available manpower.

There is thus a role for courses at different levels:

- General courses on telephony, SPC design, operation and maintenance to familiarise people with one set of specialised skills about related areas;
- Specific courses on standards;
- Specialised courses on software architectures for SPC systems;
- Advanced courses on large-scale software system design techniques.

The training needs are therefore very large and a good plan is for training and knowledge to be shared on an international basis. In addition to local experts, lecturers may be sought from other countries for their specialised knowledge and local staff ca be sent to work at other establishments. For example, selected staff can work for short periods at similar establishments in other countries, or attend specialised training courses at internationally supported institutions.

Prerequisite: University education including the following content:

Telecommunication education foundations;
Communication principles, transmission systems, principle of switching systems;
Probability, principles of propagation; radio propagation;
Additional attention to PCM transmission, data transmission,
Satellite communication systems, digital signal processing;
Communication services, private switching systems, data networks, traffic and queuing;
Operating systems, user interfaces, real-time systems, software engineering.
Computer science education according to some standard (e.g. ACM curriculum)

Participation in Standardisation

Telecommunications has a long tradition, and it has developed many specialised skills and branches of knowledge. The history behind some accepted knowledge or design is often very important and may not be evident from present descriptions. To understand certain issues, involvement in their formulation is thus often necessary, and this may take place through informal contact, or an informal network of human resources.

Participation in standardisation activities is a very good and ultimately an inexpensive way to achieve this kind of contact as standardisation meetings (e.g. CCITT working groups) bring together regulators of services, providers of services, suppliers and people from R&D establishments, and help all of them to understand and formulate standards and policies. Similarly, participation in scientific conferences is also important as they provide a perspective on future developments. Other useful interactions are bilateral exchange visits with related organisations in other countries.

Prerequisite: Financial and organisational support for participation in these activities, and for the later dissemination of the knowledge.

Market Support

The initial development of national telecommunications industries has invariably been with market protection and a high degree of market support. There is every indication that this is still a necessary precondition. Development is costly, and must be accompanied by guarantees/ agreements about the evaluation of the product and the reservation of a segment of the market if it is to lead to a commercially viable industry.

However, market protection can also have the negative effect of reducing the incentive for the new industry to achieve high standards and so there must also be sufficient independent evaluation and regulation to ensure that the technological trajectory of the industry is leading to clearly specified goals. This may be achieved by dividing the market into a segment serviced by a single national supplier and another by supplies obtained from the international market, by having two competing national suppliers, or by other chosen routes that lead t high product quality and a reliable telecommunications service. The choices will usually depend on the present level of development of the national telecommunications industry, the present level of the telecommunication services and the growth plan for these services.

Prerequisite: Market regulation is almost essential unless a relatively high level of expenditure can be sustained for a long period.

Marketing, Support and Long-term Growth

Just as there is a large and often underestimated gap between product development and commercial production, there is a large gap between product availability and successful market take-up. Part of this gap can be bridged by market protection or reservation: the other part requires well-organised customer support covering hardware, software and operations, and a phased plan for product enhancement. A potential customer requires assurances about a product both when it is delivered and over the life-cycle (10-20 years) of an exchange.

There are many reasons why support requires particular emphasis in the telecommunications industry. First, telecommunication services are highly visible and widely accessed and a provider must therefore make a service available with a high degree of initial reliability and a well-understood phased program for future development. The provider will therefore make very stringent demands on suppliers for assurances of reliability and on procedures for hardware and software fault detection, isolation and rectification.

Second, these systems have very large software components and these are often distributed over the network, making it difficult either to re-create fault scenarios or to test solutions. Thus the supplier must have a range of test jigs that in laboratory conditions are able to recreate network level problems with a high degree of verisimilitude. Software change, testing and release procedures must be carefully managed so that there is always incremental improvement to the product and the service.

Thirdly, the operating environment is under constant change, due to the introduction of new products at the subscriber or network levels, because of new operating standards or conventions and to meet the need for new services that may not have been planned for when the exchange was first installed.

Therefore, being the single source, a new supplier must expect to provide a wide range of support over the long product life cycle and this will require a very substantial investment in people and skills. And it must also keep to a realistic financial plan, providing a return on the investment within 15-20 years and building up sufficient reserves to support new product development.

Prerequisite: Long term finances to help develop the necessary support infrastructure.

6. Annex

CCITT Communications Standards

The technical standards recommendations of the CCITT are designed according to the letters of the alphabet, from Series A to Z.

- Series A to C cover the organisation of CCITT itself.
- D-series cover leased lines, data networks, international public telegram, telex, facsimile, photo transmission, maritime, transfer account, telephone and image transmission
- E-series cover operation, routing, network management, and quality of service of fixed and mobile telephone and telegraph networks, and ISDN
- F-series cover operation, definition, and quality of service of telegraph, mobile, telematic, data transmission, teleconference, and message handling and directory services
- G-series cover the general characteristics of international telephone analogue and digital connections and circuits, carrier systems and transmission media
- H&J series cover line transmission of non-telephone signals and the transmission of sound programs and TV signals
- I-series cover general structure and service capabilities , network aspects and functions, interworking principles and maintenance of ISDN
- K&L series cover protection against interference, construction, installation and protection of cable and other equipment
- M-series cover the general maintenance principles of international transmission systems and public and leased circuits
- N-series cover maintenance of international sound and television transmission circuits
- 0-series cover specification for measuring equipment
- P- series cover transmission quality
- Q-series cover general telephone switching and signalling, information flows in the ISDN, specification of signalling systems No. 4,5,6,7 and R1 and R2, digital local, transit, combined and international exchanges in integrated digital networks and mixed analogue and digital networks, interworking of signalling systems, digital access signalling systems, network layer and user-network management, public land mobile, application part and interfaces and interworking with satellite mobile systems
- S-series cover telegraph transmission and terminal equipment
- T-series cover terminal equipment and protocols for telematic services and conformance testing for teletex
- V-series cover data communication over telephone network

- X-series cover data communication networks: definition of services, facilities, interfaces, transmission, signalling, switching, maintenance, administration, OSI-model etc
- Z-series cover user guidelines, formal definition, static and dynamic semantics of SDL, criteria for using Formal Description Techniques; CHILL, MML

Glossary

A

access to obtain data from memory, a peripheral, or another system

accounting rate charge per traffic unit, which can be a unit of time or information content

A/D analogue/digital

address group of digits that makes up a telephone number; in software, a location that can be specifically referred to in a program

analogue referring to a signalling technique in which the amplitude of the carrier is varied in accordance with the value of the modulating signal

ASCII American Standard Code for Information Interchange; eight-bit code for representation of characters

associated signalling method of signalling in which a signalling link is routed in parallel with each transmission route between signalling points

asynchronous occurring without a regular or predictable time relation to a specified event

B

B-channel (Basic Rate Access) provides access to two 64Kbps data channels as defined in CCITT's Rec. I.420 as 2B

background processing the automatic execution of lower priority computer programs when higher priority programs are not using the system resources

blocking inability to interconnect two lines in a network because all possible paths between them are already in use

bps bits per second

broadcast a transmission to multiple receiving locations simultaneously

byte a small group of units, usually eight, that is handled as a unit

C

C language AT&T's high-level programming language

C++ AT&T's programming language supporting object-oriented programming

call any demand to set up a connection; used as a unit of telephone traffic

call processing sequence of operations to be performed by a switching system from the acceptance of an incoming call through the final disposition of the call

call record all recorded data pertaining to a single call

card the individual board that carries the necessary circuits for particular function

carrier signalling any of the signalling techniques used in multichannel carrier signalling. The most commonly used techniques are in-band, out-of-band, and separate channel signalling

CCITT Comité Consultatief International de Télephone et de Télégraphie. An advisory committee to the International Communication Union whose recommendations cover telephony , telegraphy and data communication areas

channel a transmission path between two or more points provided by a carrier

circuit (1) means of two-way communication between two or more points; (2) a group of electrical/electronic components connected to perform a specific function

circuit switching temporary direct connection of two or more channels between two or more points in order to provide the user with exclusive use of a channel to exchange information. In circuit switching a physical path is set up between incoming and outgoing lines, in contrast to message or packet switching, in which no such physical path is established

common carrier an organisation in the business of providing regulated communication services

common channel signalling system number 7 a CCITT-specified signalling protocol to provide communication between intelligent network nodes

common control automatic switching arrangement in which the control equipment necessary for the establishment of connections is shared

communication transmission of intelligence between points of origin and reception, without alteration of the content

concurrent processing the simultaneous processing of more than one program

D

data link layer the logical entity in the OSI model concerned with transmission of data between adjacent network nodes

delta modulation method of representing an analogue signal in which the successive bits represent increments of the signal

device driver a local object providing access to, or control over , a local physical resource. A device driver may be used to access a number of devices.

DTMF Dual Tone Multifrequency Signalling: a method of signalling in which a matrix combination of two frequencies (out of four) is used to transmit numerical address information

E

E&M Signalling arrangement that uses a separate path for signalling and voice signals. The M lead (derived from mouth) transmits ground or battery to the distant end of the circuit, while incoming signals are perceived as either a grounded or open condition on the E (derived from ear) lead. E&M is the traditional inband, send and receive signalling method that is frequently replaced by out-of-band signalling

EPABX see PABX

Erlang unit of traffic intensity. One Erlang is the intensity at which one path would be continuously occupied

Ethernet a widely used local area network, developed by XEROX Corporation

exchange assembly of equipment to control the connection of incoming and outgoing lines. It includes the necessary signalling and supervisory functions

F

facsimile system for transmission of scanned images

fault tolerance the ability of a system to operate properly even if a failure occurs

full-duplex ability of system to transmit simultaneously in both directions

G

gateway a network station that serves to interconnect two otherwise incompatible networks

I

IDN Integrated Digital Network: network employing both digital switches and digital transmission

ISDN integrated services digital network: a fully digital communications facility designed to provide transparent end-to-end transmission of voice, data, video, and still images across the public switched telephone network

integrity preservation of program of data for their intended purpose

interface boundary between two pieces of equipment across which all the signals that pass are carefully defined

ISO International Organisation for Standardisation: Technical agency of the United Nations concerned with international standards

K

kernel that part of the operating system which interconnects with the hardware

L

LAN Local Area Network: a system for communication over a relatively small geographic area

load sharing distribution of tasks among a number of processing units

layer in the OSI reference model, referring to a collection of related network-processing functions that comprise one level of a hierarchy of functions

local loop a line interconnecting a customer's telephone equipment with the local telephone company exchange

LU 6.2 IBM's proprietary standard for transaction processing

M

message sequence of characters used to convey information. In data communications, messages are usually in agreed format with a heading, which establishes the destination, and the body of the message that contains the information being sent

modular a design technique that permits a system to be assembled from interchangeable components (modules)

module a hardware or software component that is discrete and identifiable

MTP Message Transfer Part: necessary mechanism to ensure reliable signal transmission (ISDN term, corresponds to the first three layers of the OSI reference model

multiplexing division of a transmission facility into two or more channels

multiprocessing simultaneous application of more than one processor to the execution of a program

N

network a series of points connected by communication channels; a switched network is a network of telephone lines used for dialled telephone calls

NFS Network File System: a set of file transfer protocols developed by Sun Microsystems to support transfer of files over heterogeneous networks

NMC Network Management Centre: centre used for control of a network. May provide traffic analysis, call detail recording, configuration control, fault detection and maintenance

node a termination point for two or more communications link

nucleus see kernel

O

octet an 8-bit-byte

on-hook telephone set not in use

OSI Open Systems Interconnection: the reference model of OSI is a logical structure for network operations. It consist of a seven-layer network architecture for the definition of standardised network protocols

OSF/Motif a standard for how UNIX applications should appear on screen in terms of icons, windows etc.

overflow excess traffic that is offered to another, alternate, route

P

PABX Private Automatic Branch Exchange: Dial telephone exchange that provides private telephone service to an organisation while allowing users access to both private and public switches outside the organisation. The terms PBX, EPABX and PABX are used interchangeably

PBX see PABX

PCM Pulse Code Modulation: a technique to convert an analogue signal into a digital bitstream for transmission. It involves sampling of analogue signal at regular intervals and coding the measured amplitude value into a series of binary values

protocol a set of strict procedures required to establish, maintain, and control communications

PSDN Public Switched Data Network

PSTN Public Switched Telephone Network

Q

Q-Sig a digital signalling standard based on CCITT Rec. Q.931 to allow private and public signalling systems for voice and data networks to co-operate

R

redundancy provision of duplicate, back-up equipment to immediately take over the function of equipment that fails

register equipment in a common control exchange that receives address information in the form of dialled pulses of DTFM signals and stores it for possible conversion or transmission

resource any means available to system users, including computational power, programs, storage capacity or communication means

roll-back a programmed return to a prior checkpoint

S

signal a physical, time dependent energy value used for the purpose of conveying information
signalling process by which a caller or equipment at the transmitting end informs the receiving end that a message is to be communicated
source code a program usually written in a high-level language
subscriber line telephone line connecting the exchange to the subscriber's station
synchronous having a constant time interval between successive bits, characters etc.

T

tariff the published rate for the use of specific unit of equipment or service
telecommunication any process that permits the passage of information from the sender to one or more receivers
trunk transmission path used to interconnect exchanges
twisted pair two insulated wires twisted together

INDEX

5ESS 35
Accounting Management 7
additional services 23
Alcatel 63
allocation 42
Application programs 23
archiving system 64
availability figures 67
billing 53
Billing Centre 51
billing system 54
BORSCHT 36
Brazil 4; 72; 73
C 52; 63
C++ 52
C11 11
C2 11
Cailho 11
call handling 34
call processing 39; 66
call record 35
call routing 40
CAP 48
CCITT R5 11
CCITT Signalling System Number 7 12
CDOT 73
centralised and monolithic architecture 31
CEPT-L1 11
charging data 43
CHILL 21; 63; 71
CODEC 36
common channel signalling 13
common channel signalling channel 13
common channel signalling system 12
common control 14; 15
common database 34
Common Signalling Channels 32
communication protocol 10
communication services 30
complexity of SPC software 58
Computer Aided Planning 48
conference call 39
configuration 16
Configuration control 42
Configuration Management 7
conformance testing 71
conversion 5; 9
CR 40
DDI 11
DECnet 52
delta modulation. 9
development organisation 61
device handler layer 37
dialogue language 45
Digital Subscriber Signalling 11
distributed control 33
DSS 11
DTFM 36
DTMF 10; 25
Dual Tone Multi-Frequency 10
E&M 11
ESS 2
ESS 5 63
exchange architecture 33
Fault Management 7
fault-tolerance 57
file management 25
finite state machine 20; 21; 37
FSM 21
FTAM 52
future proof 66
generic interface 38
generic package 15
global operating system 25
Global process management 27
HP9000 52
I/O device management 25
I/O system 28
IHC process 39
incoming half-call 39
India 4; 63; 72
Ingres database 52
integrity of databases 29
inter-process communication 25
Internet 52
interrupt handling 25
ISDN 5; 11; 12; 33; 41

ISDN subscriber lines 13
ITT 1200 35
ITT 1240 23
kernel 23; 26; 31; 37
kernel processes 26
Korea 63; 72
layer 66
layer interface 20
layers 20
Line Test and Subscriber Service Centre 51
local 13
local operating system 25
Local process management 27
LU 6.2 52
Maintenance 48
maintenance software 44
maintenance activities 16
Maintenance system 50
man-machine dialogue 45
Man-Machine Language 45; 63
management style 61
management system 17
marker 35; 36; 41
maximum response times 66
memory management 25
message 21; 30
message transfer part 12
MML 45; 63; 71
module 22; 63; 66
MTP 12
multi-party 39
Network design and capacity planning 7
network element 17; 48
network elements 18
network management 18
Network Management Centre 18; 51
network management functions 17
network planning 47
NFS 52
NMC 18
NTT 63
number analysis 34
O&M 66
OHC 40
OMC 18
Open Systems Interconnection 11
operating system 23; 24; 39; 43; 46; 52; 63; 64; 67; 70; 75
operating system 63
operation and maintenance activities 15
Operation and Maintenance Centre 51
Operation and Maintenance Centres 18
operation software 23
Organisation 69
organisational culture 69
OSF/Motif 52
OSI 19; 52
outgoing circuit 34
outgoing half-call process 40
package 21
packet switch 33
PCM 9; 13; 75
Performance Management 7
Performance measurement 42
periodic scheduling 27
peripheral circuits 14
peripheral control units 14
person-machine 66
person-machine interaction 32
Person-Machine interface 42
personnel 69
PHAMOS 52
PMI 44
pools of memory 30
process 20; 21; 24; 25; 27
process management 25; 27
processing unit 22
program 21
programming language 19; 21; 55; 63
programming languages 57
public telephony systems 5
pulse code modulation 9
pulse metering 40
push button 9
queues 31
R2 37
real-time 24; 65; 69; 75
real-time constraints 65
real-time operating system 65
Real-time software 65
real-time, fault-tolerant operating system 33
reconfiguration 24
recovery processes 37

roll-back mechanism 29
roll-forward mechanism 29
rotary dial 9
routing 34
scheduling 25
SDL 62; 71
SDL/GR 62
SDL/PR 62
Security Management 7
Service Order Centre 51
Siemens 63
signal 21
signal conversion 6
signalling 5; 6; 10; 37; 66
signalling layer 37
signalling system 66
Signalling System No. 7 12
Signalling System No.7 71; 72
SLIC 36
SNA 52
software architecture 33; 53
software engineering techniques 64
Software risks 56
solicitation 21
SPC 2; 56
Specification and Design Language 62
SQL 19; 52
signalling function 38
standardisation 76
subscriber interface 14
subscriber line interface 36
Supporting Information Technology 70
switching 5; 6; 13
switching activities 15
switching matrix 14; 15; 35
switching network 34
System 12 3
system integrity 25; 28; 43
system manager. 25
System No. 7 13
System X 3
tasks 20
telecommunication 5
telecommunication standards 72
telephonic support 23; 35
telephony application 23
test organisation 61
time and charges 41
time services 25; 27
time-critical 24
traffic measurements 43
training 61; 75
transaction 35
transmission 5; 6; 13
Transmission Test Centre 51
Tropico 73
trunks 13
ttelephony devices 32
TUP 13
UNIX 52; 63; 64; 71
UP 12
user interface 6; 9
user parts 12
validity checks 29
virtual machine 66
volumes of data 59
workstation 64
X-Open' 71
X-Windows 52
X/Open 52
XPG3 52
XPG4 71