ND TPS
General Description

ND-60.105.03

ND TPS
General Description

ND-60.105.03

NOTICE

The information in this document is subject to change without notice. Norsk Data A.S assumes no responsibility for any errors that may appear in this document. Norsk Data A.S assumes no responsibility for the use or reliability of its software on equipment that is not furnished or supported by Norsk Data A.S.

The information described in this document is protected by copyright. It may not be photocopied, reproduced or translated without the prior consent of Norsk Data A.S.

PRINTING RECORD

Printing	Notes
09/78	ORGINAL PRINTING
10/78	Revision A
10/81	Version 02
12/81	Revision A
	The following pages have been revised:
	v, vii, viii, 2–12, 2–13
12/82	Version 03

ND TPS-General Description
Publ.No. ND–60.105.03

NORSK DATA A.S
P.O. Box 4, Lindeberg gård
Oslo 10, Norway

Manuals can be updated in two ways, new versions and revisions. New versions consist of a complete new manual which replaces the old manual. New versions incorporate all revisions since the previous version. Revisions consist of one or more single pages to be merged into the manual by the user, each revised page being listed on the new printing record sent out with the revision. The old printing record should be replaced by the new one.

New versions and revisions are announced in the ND Bulletin and can be ordered as described below.

The reader's comments form at the back of this manual can be used both to report errors in the manual and to give an evaluation of the manual. Both detailed and general comments are welcome.

These forms, together with all types of inquiry and requests for documentation should be sent to the local ND office or (in Norway) to:

Documentation Department
Norsk Data A.S
P.O. Box 4, Lindeberg gård
Oslo 10

PREFACE

THE PRODUCT

This manual describes the ND Transaction Processing System, version D:

ND TPS ND-10101D

ND TPS is a general transaction processing system that initiates and controls transactions between various input/output devices and application programs. TPS provides facilities for handling terminals, data communications, data bases, etc.

THE READER

ND TPS - General Description is an introduction to the ND Transaction Processing System and should be of interest to both data processing managers, system analysts, programmers and operators. The system is described as it appears to a user, programmer or operator without too many details. This manual should therefore be read before the more detailed TPS manuals mentioned below.

PREREQUISITE KNOWLEDGE

No previous knowledge of ND TPS or transaction processing in general is presupposed in this manual. However, some knowledge of the SINTRAN III operating system is recommended. In order to gain a full understanding of TPS, the reader should also be familiar with the SIBAS data base system and FOCUS or NSHS, the ND screen handling system. General descriptions of these systems are found in:

Introduction to SINTRAN III, ND-60.125
SIBAS II Users Manual, Chapter 1, ND-60.127
The ND Screen Handling System, Chapter 1, ND-60.088
The FOCUS Screen Handling System, Chapter 1, ND-60.137

THE MANUAL

This manual is meant to be read from beginning to end since most sections assume knowledge of the previous sections. After an introduction in Chapter 1, the manual starts in Chapter 2 with a discussion of transactions and transaction processing systems and a general description of the main tasks of TPS. In Chapter 3, the internal structure of TPS is described in more detail in relation to the tasks performed. The processing of a single transaction, step by step, is described in Chapter 4. Chapter 5 discusses operating procedures, including a fairly detailed treatment of system backup and restart facilities. Performance considerations and hardware and software requirements are treated in Chapter 6.

RELATED MANUALS

Related manuals containing more detailed information about ND TPS are:

ND TPS User's Guide, ND-60.111
ND TPS System Supervisor's Guide, ND-30.006

TABLE OF CONTENTS

+ \+ \+

Section: *Page:*

1 INTRODUCTION

ND TPS is a transaction processing system for the ND family of computers. The system provides many advanced features and its modular and open-ended design makes it possible to tailor configurations for almost any type of user needs.

ND computers with the SINTRAN III operating system, SIBAS DBMS,FOCUS or NSHS and ND TPS provide all the tools needed for the following alternatives:

- a single computer system with a variety of peripheral devices running TPS to perform on-line transactions on a local data base

- a multi-CPU system with a single TPS system distributed over several CPUs (both ND-100s and ND-500s) for more efficient processing and faster response time

- a fully distributed system with several TPS systems geographically separated. Each branch office has the equipment necessary to perform local functions as desired. Non-local transactions will be routed to the correct site through a simple addressing system

- a distributed system with communication between ND computers running TPS and other computers, for example an IBM computer at a central site and ND computers at local sites. On line transactions may be run on both local and central data bases and transactions may be automatically routed to the central site if they cannot be served locally

Although the possibilities outlined above may give the impression that ND TPS is a complex system, this is not so. The product itself is fairly simple, with a clear cut separation of cooperating modules, a design which leaves open possibilities to add new features on demand.

Data base management is done through the SIBAS data base management system, a flexible DBMS that provides:

- most of the capabilities specified by the CODASYL committee

- extensive backup and logging facilities together with complete facilities for data base recovery and transaction restart

The ND Screen Handling Systems (FOCUS or NSHS) provide facilities for complete terminal control, including formatting output, reading input, cursor control, etc. FOCUS/NSHS, running under TPS, can be used for both:

- standard asynchronous terminals

- synchronous terminals

Special communication devices and networks are handled by a set of special input output modules. These are provided by ND TPS as an option, tailoring each system to its particular configuration.

Part of the design philosophy is to leave a number of the functions in the system to the operating system, SINTRAN III, reducing software complexity without losing control over the transaction processing as such. This does not cause a decrease in efficiency in TPS because the SINTRAN III real-time operating system is designed to handle real-time processing with maximum efficiency and minimum overhead.

Thus, the following functions are managed by SINTRAN III directly, with TPS supervision:

- communication with standard input/output devices
- priority handling
- paging of both application and system programs
- multithread task handling
- reentrancy control
- storage protection

Letting SINTRAN III manage these functions makes it possible to offer both advanced features and a fairly simple design in the same system.

A ND TPS feature that allows the user to exert greater control over the system is the so-called special applications. These special applications are activated at certain operational stages, such as:

- ND TPS initial start
- ND TPS close down
- abnormal end of a transaction
- synchronized checkpoint
- restart after failure

ND TPS combines the features of large mainframe computers with the advantages of minicomputers - distributed processing, fast response times, easy terminal handling, automatic recovery and restart, several high-level languages, an advanced data base system - features that can meet the requirements of almost all types of users.

2 TRANSACTION PROCESSING

This chapter discusses the concepts of a transaction and a transaction processing system. The steps in processing a transaction are listed and from them the main tasks of TPS are derived. The different ways of starting and terminating transactions with TPS are described, and communication with the external environment and with the data base is discussed.

2.1 WHAT IS ND TPS?

2.1.1 Transactions

The normal meaning of a *transaction* in the everyday world is a piece of business that takes place between individuals or companies, an interaction involving the receiving and processing of some form of information.

The result of the transaction is often recorded on a *file* of some type. The file will usually contain the results of many such transactions and thus serve as an important source of information.

A procedure for fast *updating* of the file will result in a file with more timely and correct information. In order to make optimum use of this information, a procedure for *retrieving* the information from the file quickly and easily is also important.

2.1.2 Transaction Processing Systems

A computerized on-line *transaction processing system* is a computer-based system that allows the user to process data and update a file as soon as the information arrives, and at the same time gives him the tools for retrieving the information as soon as he needs it.

The user will normally have a *terminal* available that is on-line (directly connected) to the file, or *data base*. He will enter the transaction input on the terminal, a program in the computer will process it and access the data base and the user will receive a response within seconds.

Thus, a transaction in a data processing system may be defined as an on-line, program-controlled interaction between an input/output device of some type, usually a terminal, and a data base on an on-line storage medium.

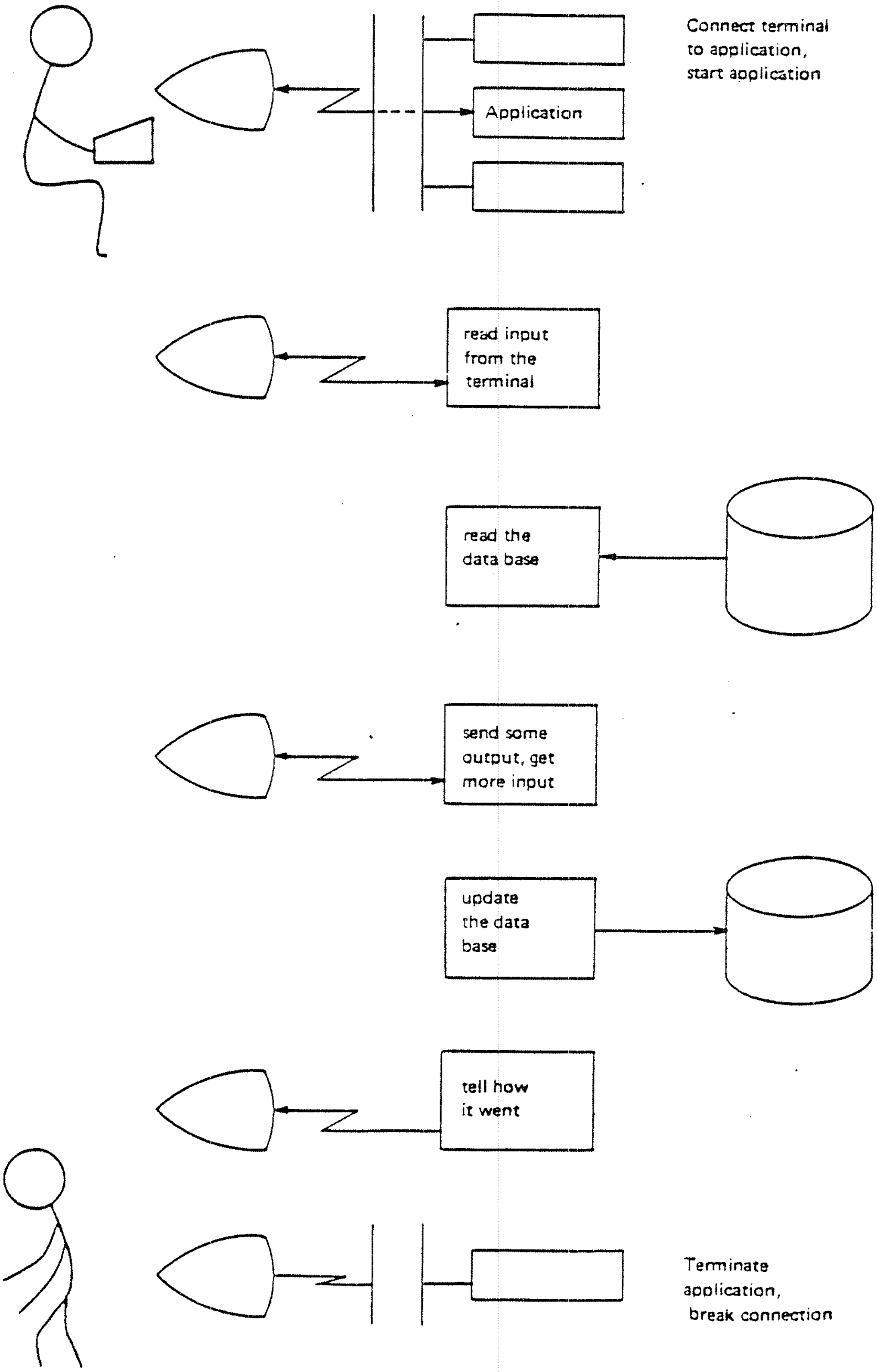


Figure 2.1: A TYPICAL TRANSACTION

2.1.3 Types of Transactions

A transaction may be an inquiry which only reads the data, formats it and sends it to the terminal (*inquiry* transaction). The transaction may update the data base, perhaps after a conversational interaction between the program and the user *(update* transaction). The transaction may gather in data interactively and store it in a temporary file for later batch updating (*data entry)*. A transaction may also generate a relatively large amount of output to a printer *(report generation*).

Most transactions are characterized by:

- a fairly small amount of input and output
- conversational interaction
- short duration
- fast response time

although none of these characters is absolute.

2.1.4 The Main Tasks of TPS

The main steps in processing a transaction are (See Figure 2.1):

- setting up the connection between the terminal and the processing program and starting the transaction
- reading and processing input from the terminal
- reading and updating the data base
- preparing and sending output to the terminal
- terminating the transaction

To accomplish this, a transaction processing system must provide facilities for handling the following main tasks: (See Figure 2.2);

- starting, controlling and terminating transactions
- communicating with terminals and other I/O devices
- accessing the data base, including reading it and updating it

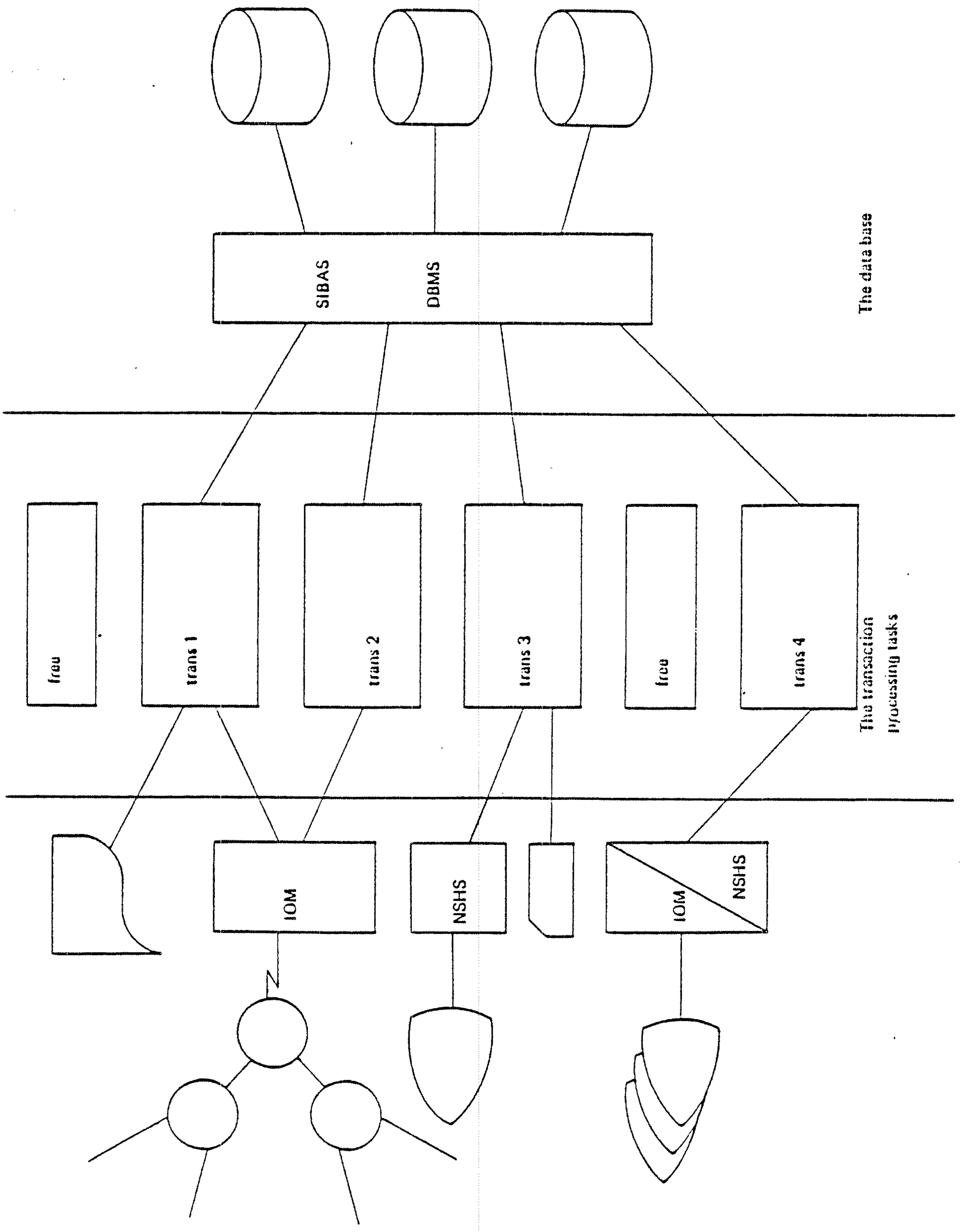


Figure 2.2: THE MAIN PARTS OF TPS

A short description of how these tasks are handled in the ND Transaction Processing System follows. In addition to the TPS modules themselves, the ND TPS system uses the facilities of the SINTRAN III operating system, the SIBAS data base system,and FOCUS or NSHS, the ND screen handling systems. In order to gain a full understanding of TPS, the reader should be familiar with these systems.

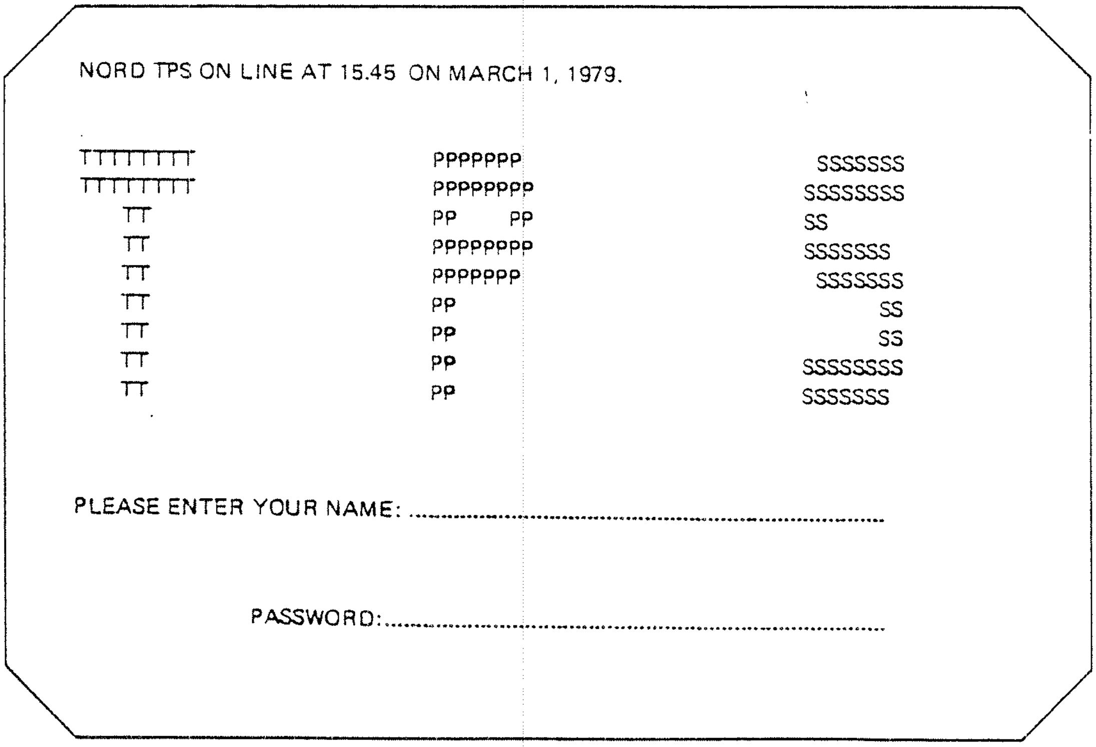
2.2 CONTROLLING TRANSACTIONS

There are two main ways of setting up a transaction, depending on whether the connection between the terminal and TPS is permanent or only lasts as long as a single transaction.

2.2.1 Type 1: Permanent Terminal Transactions

Permanently connected terminals are the simplest to handle and usually give faster response time, because the overhead in setting up the connection between the terminal and TPS is avoided. In addition, TPS provides some special applications (SIGNON and SELECT) designed mainly for permanently connected terminals.

SIGNON helps the terminal user to sign on to the system by writing a picture on the terminal asking for the user's name and password. A typical SIGNON picture is shown in figure 2.3. When the user has written his name and password, SIGNON will check them, and if they are accepted, control will be given to SELECT.

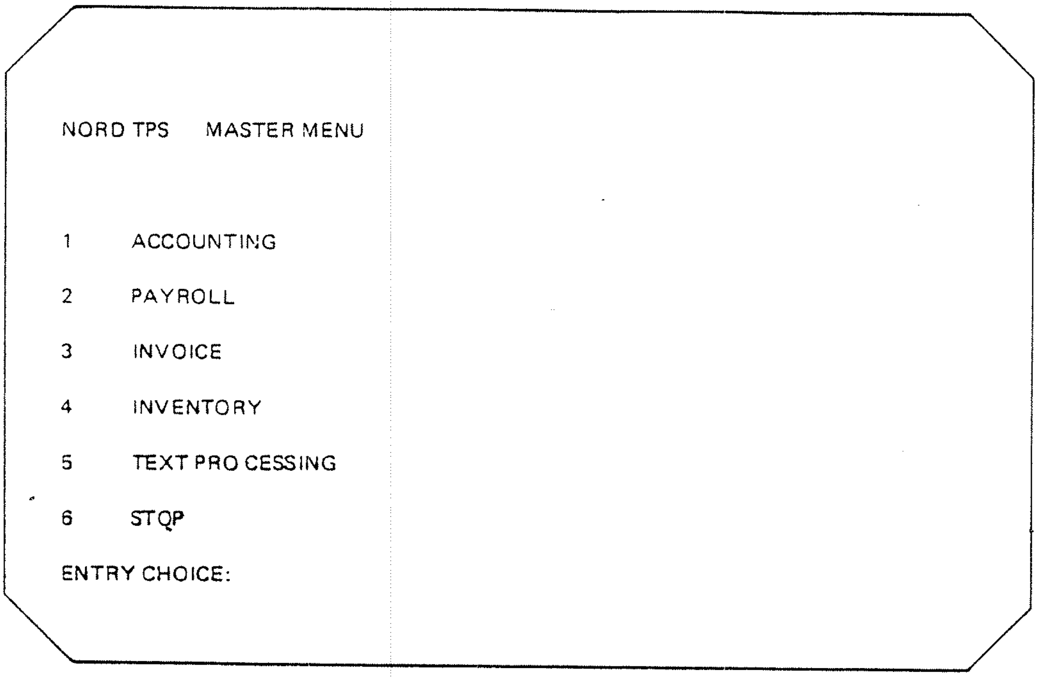


```
NORD TPS ON LINE AT 15.45 ON MARCH 1, 1979.

TTTTTTTT            PPPPPPP             SSSSSSS
TTTTTTTT            PPPPPPPP           SSSSSSSS
   TT               PP    PP           SS
   TT               PPPPPPPP           SSSSSSS
   TT               PPPPPPP             SSSSSSS
   TT               PP                       SS
   TT               PP                       SS
   TT               PP                 SSSSSSSS
   TT               PP                 SSSSSSS


PLEASE ENTER YOUR NAME: ...................................................


                 PASSWORD:.................................................
```

Figure 2.3: A SIGNON PICTURE

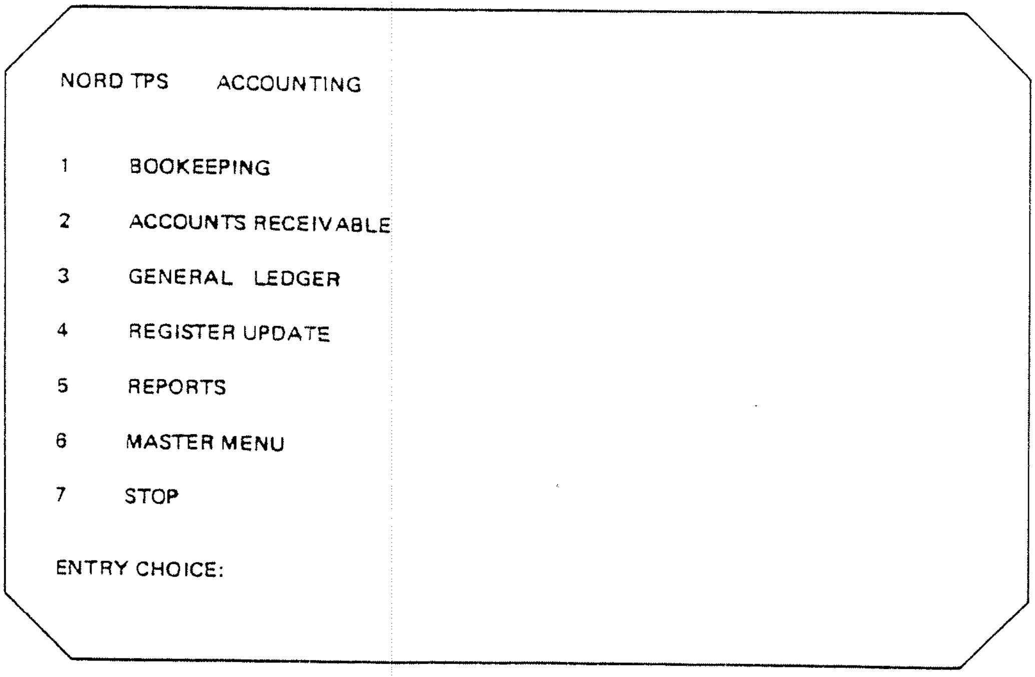
SELECT will help the user to start the transaction program (application) he wants by writing a menu picture on the terminal, as shown in figure 2.4 A. The user just has to select the item he wants and enter its number. The application will then be started.

It is also possible to have sub-menus, i.e. one menu choice will give a new menu, as shown in figure 2.4B. It is possible to have as many sub-levels of menus as desired. The user can also go from a sub-menu back to the master-menu, back to SIGNON, or even into SINTRAN as a timesharing user.



```
NORD TPS     MASTER MENU


1       ACCOUNTING

2       PAYROLL

3       INVOICE

4       INVENTORY

5       TEXT PROCESSING

6       STOP

ENTRY CHOICE:
```

Figure 2.4A: A MASTER MENU



```
NORD TPS      ACCOUNTING


1       BOOKEEPING

2       ACCOUNTS RECEIVABLE

3       GENERAL   LEDGER

4       REGISTER UPDATE

5       REPORTS

6       MASTER MENU

7       STOP

ENTRY CHOICE:
```

Figure 2.4B: A SUB-MENU

However, sooner or later a user application will be started. The user will then carry out his transaction, probably involving interaction with both terminal and data base.

When he is done, control will be given to another special application, SIGNOFF. The main task of this application is to find out what type of transaction it is, in this case a ''never-ending'' transaction, and to terminate it accordingly, in this case by giving control back to SIGNON.

The terminal is now ready to accept a new user and start a new transaction. If it is not necessary to go through the SIGNON procedure again, SIGNOFF could have given control directly to SELECT.

In both cases, the terminal will not be released from TPS, i.e. the Transaction Processing Task (TPT) in TPS controlling this terminal will not release the terminal. There is a permanent connection between the terminal and the TPT. There is a never-ending succession of applications running on this terminal (SIGNON, SELECT, user application, SIGNOFF, SIGNON, SELECT, user application, etc).

This is illustrated in figure 2.5, Type 1.

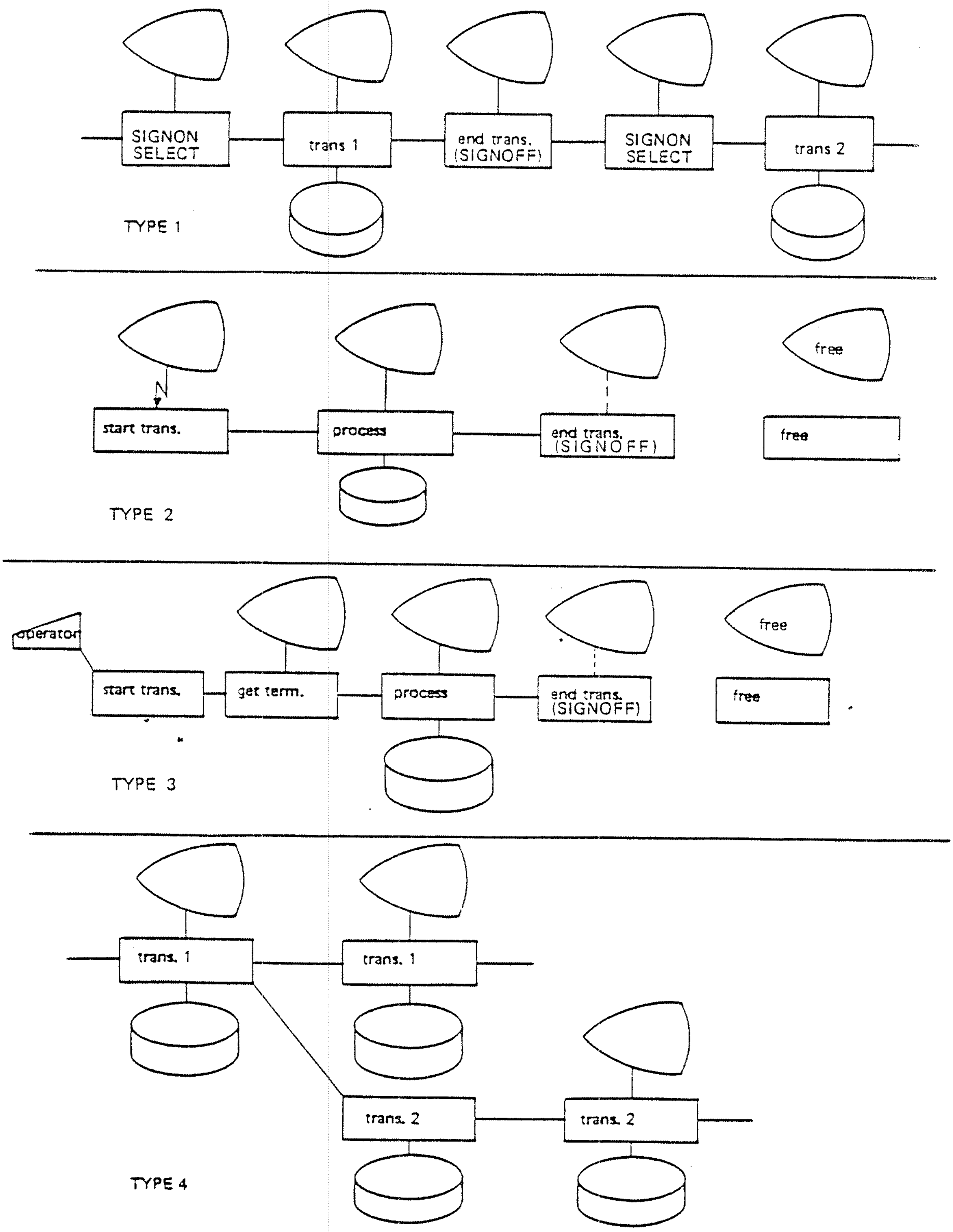


Figure 2.5: TYPES OF TPS TRANSACTIONS

2.2.2 Type 2: Short Terminal Transactions

The other main way of setting up a transaction is by connecting a terminal to TPS whenever a transaction is to be carried out on the terminal and releasing the terminal when the transaction is done. This frees the terminal for other use and, possibly even more importantly, also frees the Transaction Processing Task (TPT) in TPS for use with another terminal.

This method should be used if there are many terminals that are not in constant use. They may then share a limited number of TPTs among themselves. The disadvantage is that there is more overhead in starting up the transaction.

The procedure would be as follows:

A terminal connected to a network is inactive. The user presses a special function key that causes a message to be sent over the network to TPS. TPS allocates a TPT to the terminal and the TPT starts an application program that can converse with the terminal. A dialogue follows between the user and the application program, and the data base is read and updated. When the user is done, the application terminates and both the terminal and the TPT are freed.

2.2.3 Type 3: Short Local Terminal Transactions

A terminal is connected locally to the computer and can be used for both TPS processing and other processing. The terminal can be brought into the TPS system by issuing a command at the TPS *operator console*,a special terminal devoted to operational control of TPS (See Chapter 5). A TPT will be allocated and the application program indicated in the command started. When the application program terminates, both the TPT and the terminal will be freed.

2.2.4 Type 4: Concurrent Transactions

A transaction that is active can start another transaction to run ***concurrently***. A new TPT will be allocated and the new application started. Data can be sent from the mother task to the daughter task. The daughter task will not have a terminal - if it needs one, it must set up the connection itself (See Section 2.3).

2.2.5 Type 5: Future and Periodic Transactions

A transaction that is active can start another transaction at some time in the future, either at a given absolute time or periodically. When the time comes, the transaction will be started as for Type 4. The TPS operator can also use commands to start a single or periodic application at a specified time. TPS need only be informed once of a periodic application. It will then be started periodically at the correct times.

2.3 HANDLING INPUT/OUTPUT

TPS has facilities for handling input/output from many types of terminals and I/O devices and from data bases.

The data bases are controlled through the SIBAS data base management system.

Most terminal types can be handled through the ND screen handling systems, FOCUS and NSHS.

Other I/O devices can be divided into two main types:

- *standard devices* allocated to and controlled by a particular application program. Examples are standard SINTRAN terminals, printers, card readers, non-SIBAS files

- *special TPS devices* that are allocated to TPS modules, not individual applications. Examples are networks and synchronous terminals

2.3.1 Standard Devices

These devices are controlled by application programs directly through SINTRAN. Devices belonging to this group are generally available to all users of the computer system, also non-TPS users. They are allocated to an application program when the program requests them and released by the program when it no longer needs them. The application program will access the devices through the SINTRAN routines available in the language it is written in, such as OPEN, CLOSE, READ, WRITE, etc.

2.3.2 Special TPS Devices

Devices in this group are controlled by special programs called *Input/Output Modules* (IOMs). There is one IOM for each type of device, although each IOM may control many devices.

Terminal networks would be controlled by IOMs, as would non-standard terminals. All communication with distributed processing sites is done via IOMs.

Application programs communicate with these devices through sessions (see 3.2.3). Several IOMs are implemented at the moment:

- buffered terminals with the ISO 1745 or IBM 3270 line protocols
- X-25 permanent virtual channel for inter-system communication
- remote access to an IBM 360/370 central computer via 3270 emulation

The number and type of IOMs in a particular TPS system depends entirely on the user's needs.

2.3.3 The ND Screen Handling System

Terminals can be controlled by application programs through the *ND screen-handling system* (NSHS) which contains routines for formating output pictures, reading input, field definition, cursor control, etc.

Terminals controlled by NSHS may be either standard SINTRAN terminals or special devices controlled by input/output modules. The application program will use the same NSHS calls for both types of terminals.

2.3.4 The FOCUS Screen Handling System

The Screen Handling System, FOCUS, is a new system that can be used within one CPU or distributed to one or more Front End CPU's.

FOCUS is described in the "FOCUS Screen Handling System" manual. The distribution of the processing part of the screen handling is transparent to the user programs, and carried out by the runtime system.

2.3.5 The SIBAS Data Base Management System

The data base in a TPS system is controlled by the SIBAS *data base management system.*Other files/data bases may also be accessed by application programs, but these would be treated as standard devices. They would not be a part of the integrated functions of TPS and SIBAS, such as data base open/ close, checkpoint, rollback and recovery.

SIBAS is a DBMS that provides most of the capabilities specified by the CODASYL committee for a data base facility in COBOL. Similar facilities are available to FORTRAN and NPL programmers.

A SIBAS data base is accessed by application programs through the usual routine calls to SIBAS.

SIBAS allows direct and fast access to all data. It provides several methods of file organization and access, separation of physical and logical organization, concurrent or exclusive access and data independence. It has facilities for backup and restart to insure data integrity and privacy locks to prevent unauthorized access.

2.4 STANDARD TPS

TPS exists as basic modules plus a number of options. The basic modules consist of transaction control and service routines, the SIBAS data base system, the NSHS or FOCUS screen handling system, operator communication, message queuing and routing routines and a standard set of special applications. In the default version of TPS, all of these are run in a single CPU.

TPS options include multi-CPU systems, using both ND-100s and ND-500s, several transaction control modules and input/output modules for special devices, networks and distributed processing.

3 THE STRUCTURE OF TPS

In Chapter 2, the main tasks of TPS were discussed. These were starting and terminating transactions, controlling the input/output and accessing the data base. In addition, provision must be made for starting, stopping and controlling the TPS system itself. This is accomplished through the operator communication module. Also, facilities must be available for simple and efficient communication between the modules of TPS. This is done by a message routing and queuing system.

In summary, the TPS system is composed of the following types of modules (See Figure 3.1):

- transaction control and service routines
- SIBAS data base control routines (separate subsystem)
- FOCUS or NSHS screen handling (separate subsystem)
- input/output modules
- operator communication
- message routing and queuing routines

The application programs themselves are mainly user-written, but TPS includes a number of:

- special application programs

to carry out user-oriented system functions.

This chapter gives a short description of these seven types of modules.

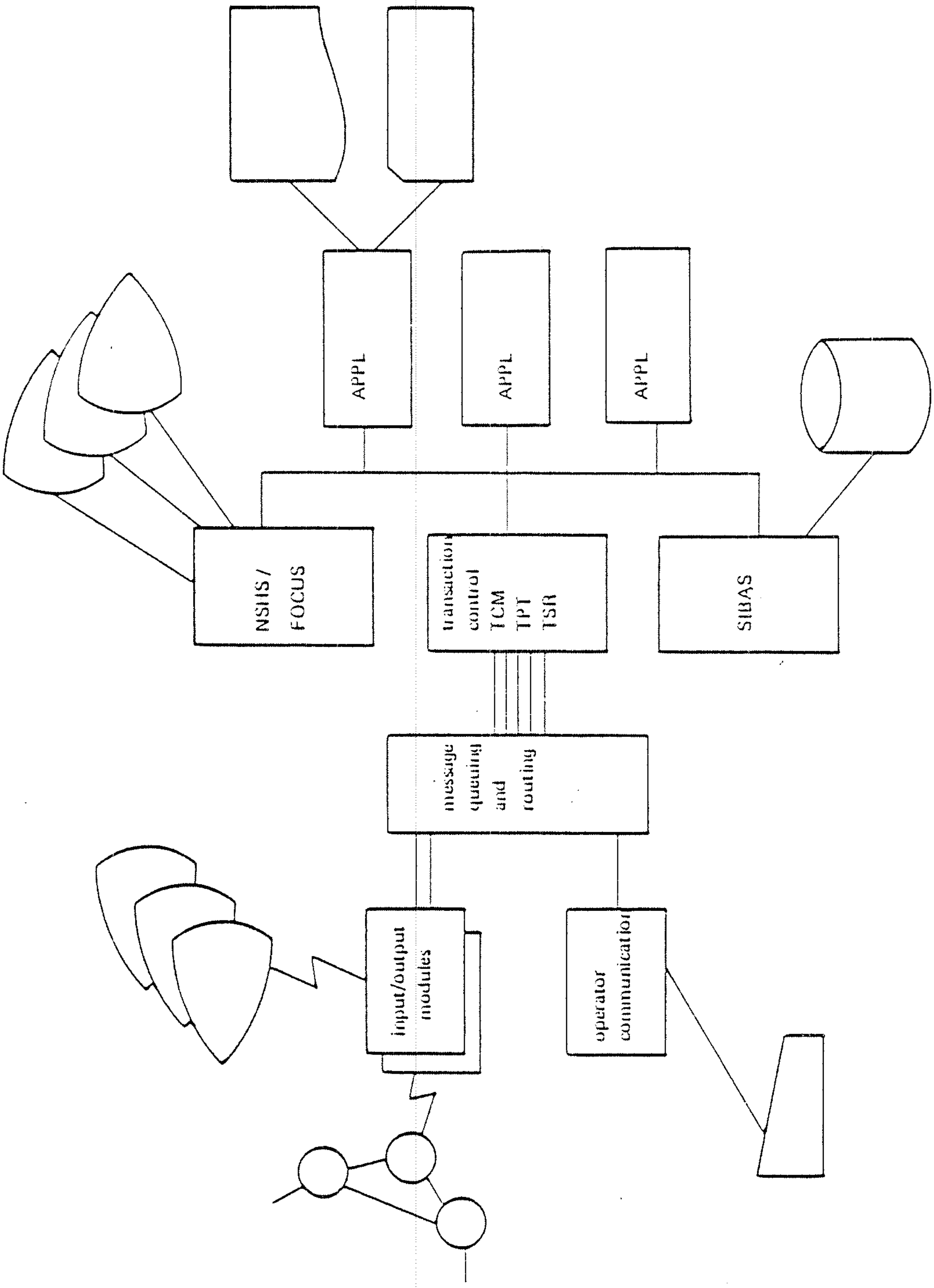


Figure 3.1: TPS MODULES

3.1 TRANSACTION CONTROL

3.1.1 Transaction Control Modules

Transaction control is handled by one or more **transaction control modules** (TCMs). The TCM supervises the application programs belonging to it and controls system functions such as start,stop and checkpoint.

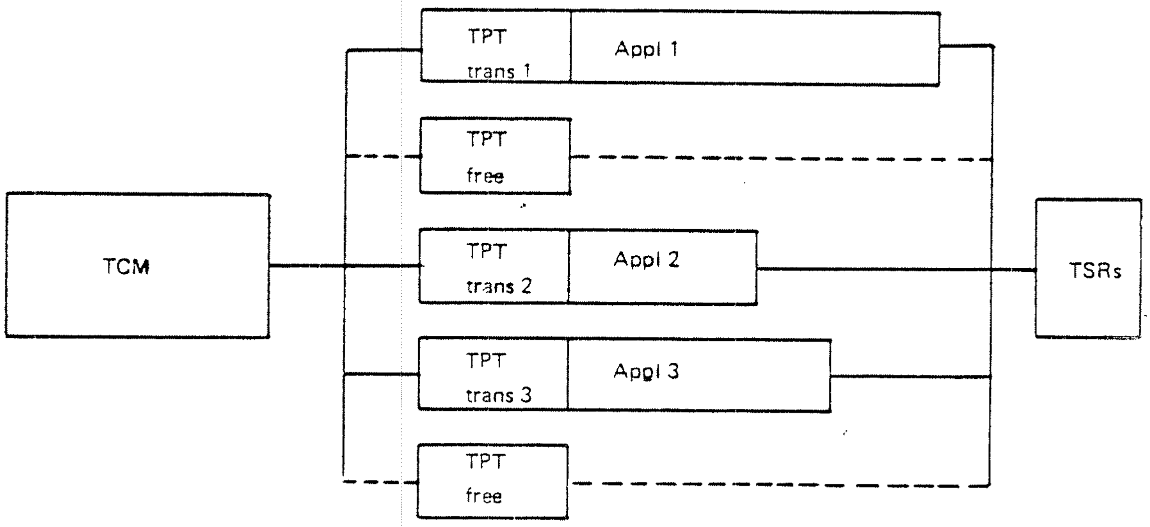


Figure 3.2: TRANSACTION CONTROL

3.1.2 Transaction Processing Tasks

Each TCM has a number of **transaction processing tasks** (TPTs) (See Figure 3.2). The TPTs are a set of identical programs belonging to a pool. When a transaction is started, TCM may allocate a free TPT to the transaction from the pool and start the TPT. When the transaction is finished, TCM may free the TPT. Some TPTs may be permanently allocated to terminals and can process many transactions in a row.

The TPT has several functions:

- to start the application program used by the transaction
- to terminate the application program when it is done and either switch to a new application program or terminate the transaction
- to provide the application program with data areas (all application programs are reentrant and thus may not be written into)
- to provide checkpoint/restart facilities for the application program in case of system failure

3.1.3 Transaction Service Routines

The **transaction service routines** are a set of routines supplied with TPS to assist the application programmer in performing functions such as administrating task control, communicating with I/O devices, and sending messages. These routines allow the programmer to concentrate on the applications as such, without having to be concerned with complex details of a real-time environment. The routines represent a clean and logical interface between application programs and ND TPS (See Figure 3.3).

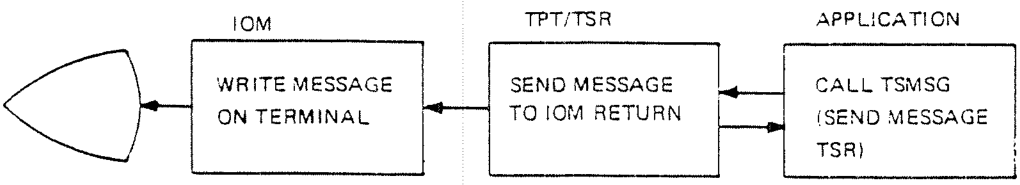

IOM
WRITE MESSAGE ON TERMINAL
TPT/TSR
SEND MESSAGE TO IOM RETURN
APPLICATION
CALL TSMSG (SEND MESSAGE TSR)

Figure 3.3: USING A TSR

TSRs may be arranged in groups as follows:

1. Administrative services.

   Switch control to another application.
   Activate concurrent application.
   Stop transaction.
   Set termination/abort/restart/close strategy.
   Set execution time/interval.

2. Session services.

   Read message.
   Send message.
   Open/close session.

3. Checkpoint control.

   Take checkpoint.
   Allow/prevent checkpoint.

4. Message services.

   Write message on operator console.
   Broadcast message to terminals.

5. Special application TSRs.

   Restart.
   Read status.
   Read configuration information.
   Operator functions.

3.1.4 Application Programs

The *application programs* do the actual processing of the transactions. They are mainly user-written, with the exception of a number of special applications supplied with the TPS system. They may be written in FORTRAN, COBOL, PLANC, NPL or MAC (assembly language), and they may use most of the facilities available to these languages.

Application programs are *reentrant* and may be used by several transactions simultaneously without having more than one copy. The TPTs are also mainly reentrant, with only a small non-reentrant part for each TPT plus the data areas. Thus, there may be many TPTs without taking more than a minimum of space.

The maximum size of application programs and data areas in the ND-100 are user-dependent configuration parameters. Typical sizes are 24K words for programs and 5K words for data.

In the ND-500, on the other hand, both programs and data areas may be much larger (up to 134 megabytes for each).

Application programs in the ND-100 may be tested as timesharing programs under SINTRAN before being run under TPS. A special set of TSR routines is available to simulate a real-time TPS environment. SIBAS can be accessed by both TPS, timesharing and batch programs at the same time.

Application programs in ND-500 may be tested by running the ND Symbolic Debugger in an ordinary TPS-run.

The processing of a transaction may involve the activation of several application programs, one at a time. When one program is done, it may switch to another program instead of terminating. The new program can be written in the same language as the previous one or in any other of the available languages.

The new program will have access to the data area of the old program if it is defined as belonging to the *common* area of the task's data area. In a FORTRAN program, this will be a FORTRAN COMMON-area. COBOL main programs must contain a section of the DATA DIVISION which is identical in all applications which may be executed within one task.

3.1.5 Special Applications

Certain functions of the TPS system are not handled by the internal modules of TPS but are carried out as applications. They are started and terminated as any other application, under the control of a TPT.

The main reason is that application programs are easy to write, modify and load. Users can easily tailor these programs to their own needs. Functions that are common to many users, but where the detailed processing may vary from user to user, such as sign on, transaction abend and restart, are carried out by special applications. Also, special system functions such as, checkpoint, rollback and recovery are controlled by these applications.

A complete set of special application programs is supplied with TPS and many users will find that their needs are fully satisfied by these standard versions. Other users will modify the standard versions, while some users may wish to write their own versions.

The main purpose of the following special applications is communication with SIBAS. They are called by only one TPT (a special system TPT) when the system uses them (See Figure 3.4).

- TPOPEN, called when TPS is initially started. This application may open the data base for general use. It may also start up transactions and broadcast a start message to terminals controlled by IOMs

- TPCLOSE, called when TPS is closed or abnormally ended. The application may close the data base

- CHECKPOINT, called when a synchronized checkpoint is taken. The application calls the SIBAS checkpoint routine

- ROLLBACK, called when a system failure has occurred and the system is to be rolled back to a synchronized checkpoint. The application supervises the SIBAS rollback routine

- RECOVER, called when a system failure has occurred to restore the system to its state at the latest transaction checkpoints. The application supervises the SIBAS recover routine

Additional special applications, activated for each individual TPT, are:

- SIGNON, called to check the terminal operator's status and password and to reserve the terminal

- SELECT, called to determine which processing application is to be given control

- TPMON, called when an ND-500 application is started. Thereafter TPMON monitors the ND-500 process. This application may not be modified by users.

- SIGNOFF, called when a transaction terminates

- ABEND, called when a transaction terminates abnormally due to an error situation in the program itself or an error return from a system routine

- RESTART, called after a rollback or recovery has been performed. The application is called by each active TPT for the purpose of restarting the TPT's application at the correct point

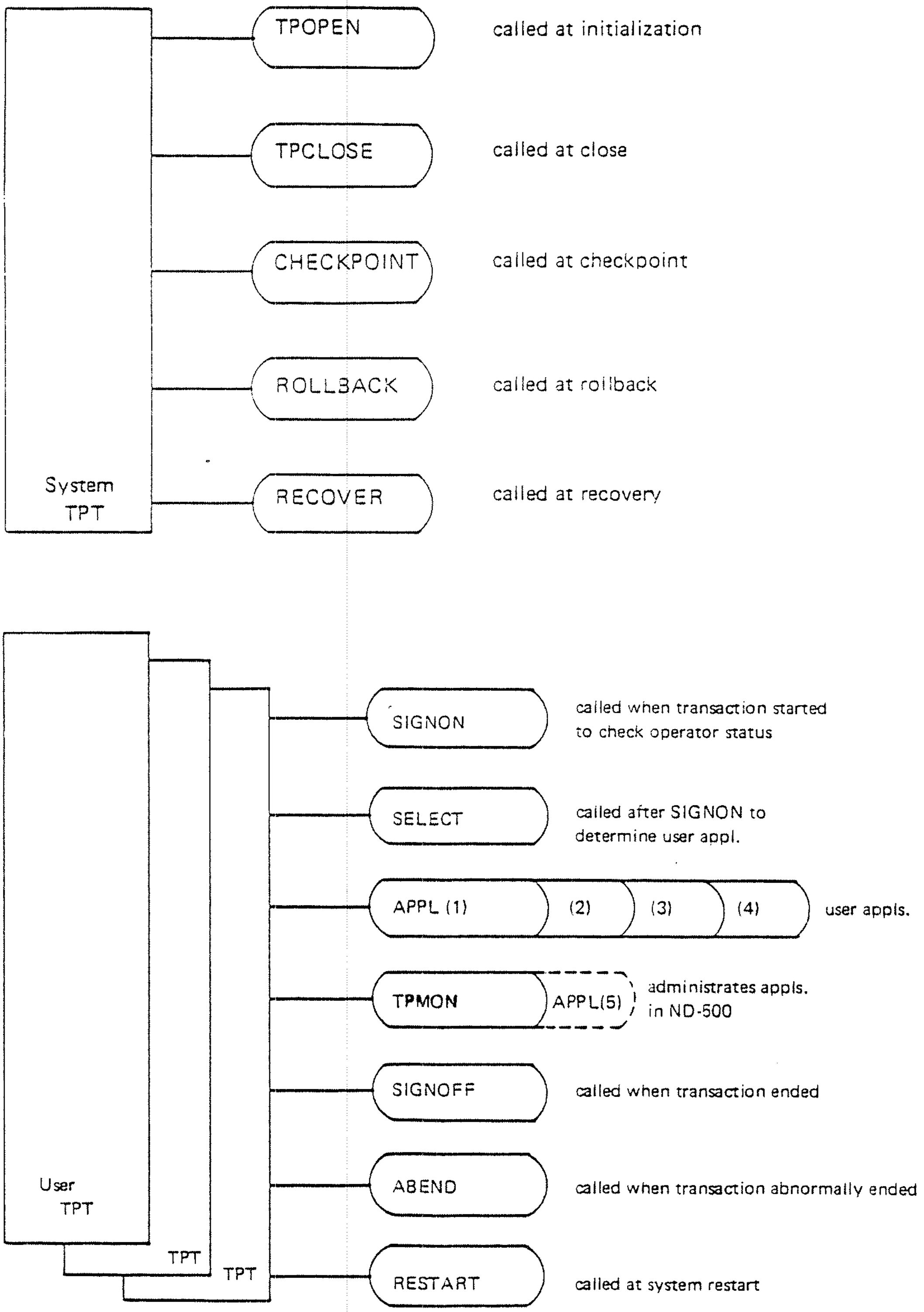


Figure 3.4: SPECIAL APPLICATIONS

3.2 INPUT/OUTPUT CONTROL

As mentioned in Section 2.3, input/output devices are divided into standard devices and special TPS devices. This section describes the way the special TPS devices are handled.

3.2.1 Input/Output Modules

Input/Output modules (IOMs) are used to control devices that cannot be controlled directly by an application program. This may be because their control is too difficult for the application program, for example, non-standard devices or devices with complicated communication protocols. However, the main use for IOMs is in connection with *networks,* and other types of multiplexed connections.

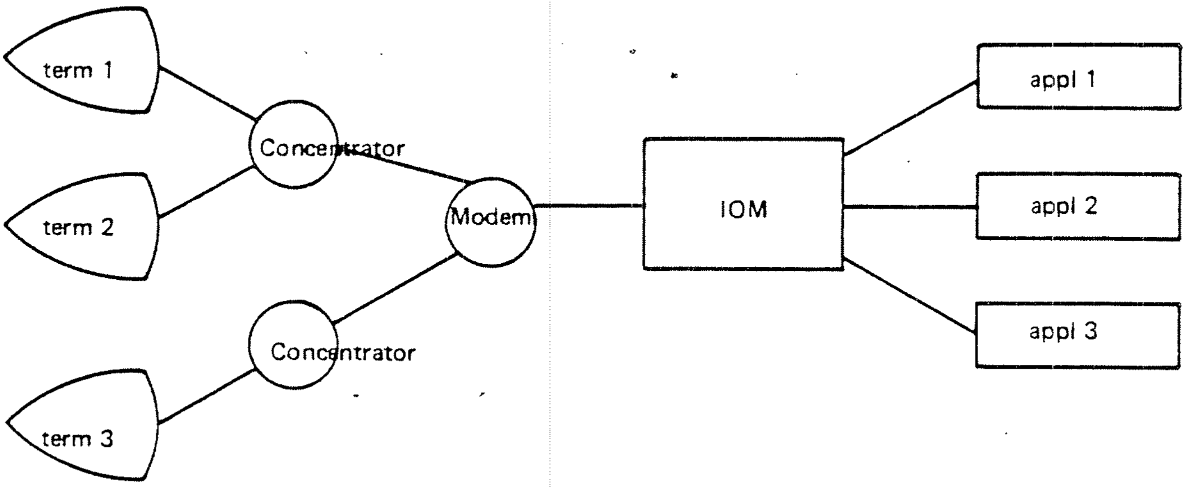


Figure 3.5: A TERMINAL NETWORK

3.2.2 Networks

A network does not belong to any one application program but may have many terminals connected to many application programs (See Figure 3.5). The connection between the terminal and the application program is not direct, but goes through modems, concentrators, etc. Another type of network connection may be the connection between two TPS systems at different processing sites (See Figure 3.6). Application programs at one site may communicate over the network with application programs at the other site.

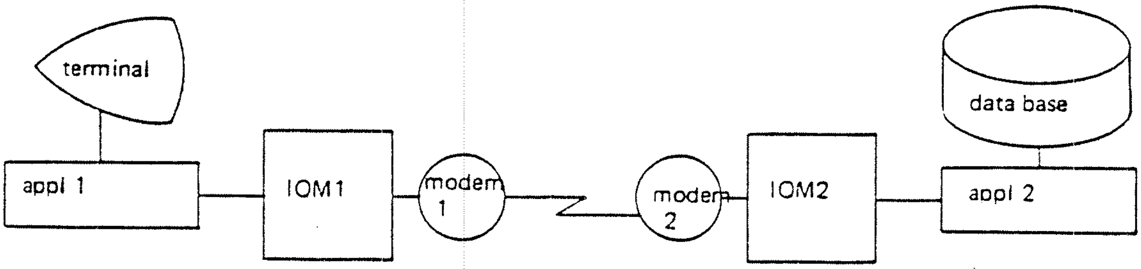


Figure 3.6: ACCESSING A REMOTE DATA BASE

3.2.3 Sessions

An application program communicates with these devices through a set of TSR routines provided by TPS. A *session* is established between the device and the application program, and the application program can then send messages to and receive messages from its session partner using these routines. Sessions may also be established between two application programs, either at the same processing site or across an external network.

Sessions may be broken and new sessions established. A transaction is only allowed to have one session at a time (but it may possess several standard devices, including standard terminals, at the same time).

3.3 DATA BASE CONTROL

3.3.1 Data Base Definition and Manipulation

The data base·is defined and created using the SIBAS data definition/redefinition language DRL. This is done independently of TPS.

The data base is accessed from application programs using the SIBAS data manipulation language DML. The DML used by a program running under TPS is the same as for a program running in a different environment, such as timesharing or batch.

It is, in fact, possible to access the same SIBAS data base from TPS, timesharing and batch programs at the same time. All SIBAS calls will go to a common SIBAS interface under the control of TPS. The timesharing or batch user will be unaware of TPS control over the data base unless a TPS *restart* should happen to change its contents.

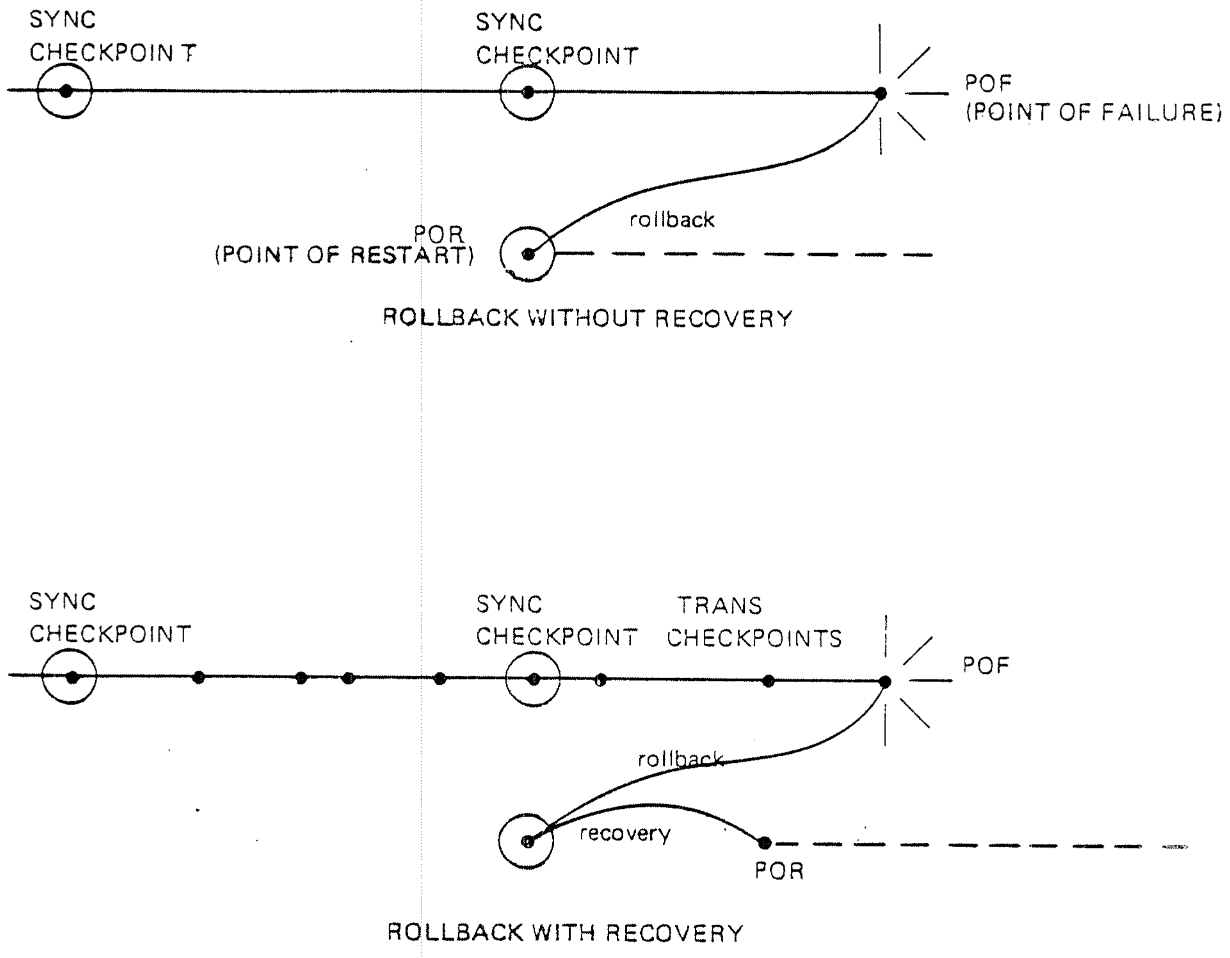
3.3.2 Checkpoint and Restart

An on-line transaction processing system should have adequate facilities for protection of the data base. If a system failure occurs in a *batch* system, a backup copy of the data base can be mounted and the whole job run once more without too much inconvenience or waste of time. If a failure occurs in an *on-line* transaction system, reentering and reprocessing the transactions may be very inconvenient, time-consuming or even impossible.

Facilities should be available to assure the integrity of the data base, to minimize the amount of data lost, and to restart the system automatically at a well-defined point.

TPS makes use of the extensive checkpoint/restart facilities of SIBAS. These are partly transparent to both the user and the application program, TPS itself controlling them. *Synchronized* checkpoints of the whole system are taken automatically at given time intervals and/or according to the load on the system. In addition, the application program can take individual *transaction* checkpoints.

If a system failure occurs, the data base can be restored to its status at a synchronized checkpoint (rollback - See Figure 3.7). Processing may continue immediately from there, in which case input data after the checkpoint must be resent. Alternatively, the data base may be updated automatically to its state at the latest transaction checkpoints (recovery) so that no input need be repeated. In both cases, those transactions which were active will be restarted automatically at the correct point.



ROLLBACK WITHOUT RECOVERY

ROLLBACK WITH RECOVERY

Figure 3.7: ROLLBACK AND RECOVERY

How often checkpoints are taken and what types of backup/restart facilities are used are system parameters controlled by the user. He must weigh the advantages of assuring the protection of data in the data base against the overhead needed to accomplish this.

Checkpoint and restart are discussed in more detail in Chapter 5.

3.4 OPERATOR COMMUNICATION

A special terminal, the operator console, is used for starting, stopping and controlling the TPS system. A set of commands is available for interaction with the whole system (system directives) and for interaction with individual modules.

System messages, both error and informative, will be written on the operator console. Application programs may also send messages to this console. In a multi-CPU system, those CPUs that do not have an operator console will have a log-writer terminal for special error messages.

3.4.1 System Directives

The following system directives are available to the operator:

- INITIATE-TPS starts the system from the beginning, opens the data base, initiates all data areas and files
- CLOSE-TPS stops the system as soon as all active transactions are done, closes the data base and files
- HALT-TPS stops the system temporarily as soon as possible
- ABEND-TPS stops the system immediately (abnormal end)
- CHECKPOINT-TPS takes a synchronized checkpoint of data base and TPS modules
- ROLLBACK-TPS restores the data base and system to the last checkpoint
- RECOVER-TPS brings the data base and system up to the point of failure
- CONTINUE-TPS starts the system after a halt, rollback or recover

3.4.2 Other Commands

Among the general commands available are commands to:

- start and stop individual TPS modules
- start transactions
- set periodic applications
- abort transactions
- broadcast messages
- read and alter the status of modules
- change system parameters
- print information about active transactions
- print information about the time queue and interval table
- start and stop the trace routine

3.5 MESSAGE ROUTING AND QUEUING

Communication between the individual TPS modules is done by messages using a buffer pool and queuing system controlled by the main dispatcher (MD). If the modules are spread across more than one CPU, MD gives the message to the correct CPU. TPS may thus make use of multiprocessing facilities in a single system.

Every TPS module has a queue for messages to that module. When a message is sent, it goes first to the main dispatcher. MD will put the message in the queue for that module and it will then start the module. The module will read its queue and process the message there. When it is done, it will usually read the queue again in case any messages have arrived in the meantime.

Routines are also available for putting messages that have been received into a waiting queue if they are not to be processed immediately. They can be read from the waiting queue later and processed.

Application programs do not have to concern themselves with these queues. They are controlled by the application program's TPT.

4 PROCESSING A TRANSACTION

This section describes in some detail the steps involved in processing a typical transaction using the standard version of TPS. It follows the transaction sequentially through the TPS system from its initiation by the user until it is terminated.

The transaction described is a transaction of Type 1, a permanently connected terminal transaction. A connection is established between the terminal and a TPT when SIGNON is first started by TPOPEN (*See Figure 4.1*). SIGNON checks the terminal operator's status and SELECT calls the correct user application. A conversation is carried on between the user and the application program and the data base is read and updated. When the user has no more input and has received all output, he indicates that the transaction is done. The application program will terminate and the SIGNOFF application will give control back to SIGNON to wait for the next transaction.

4.1 Starting the Transaction

The TPS system has been started initially and is in normal running state.

TPOPEN has started a number of transactions using the activate-task TSR. The first application to be given control is the SIGNON special application.

The terminal in our example is a standard terminal controlled by the NSHS screen handling system. The first time SIGNON is started, it will reserve the terminal. After that it will call NSHS routines to display a picture asking the user to enter his name, and then wait for an answer in the NSHS input routine (read-fields).

To start the transaction, the user will enter his name on the terminal and press the return key. SIGNON will be started up at read-fields with the reply in the input data area. It will check the user's identification and perhaps whether that user is allowed to use that particular terminal. SIGNON may then ask for a password, again using NSHS routines. When this has been checked, SIGNON will switch to SELECT, using the switch-application TSR. SELECT will display the user's master menu and wait for the user to enter the number of the entry he wants. This may result in displaying a sub-menu, going back to SIGNON or switching to a user application. If it is the latter, the transaction has been started.

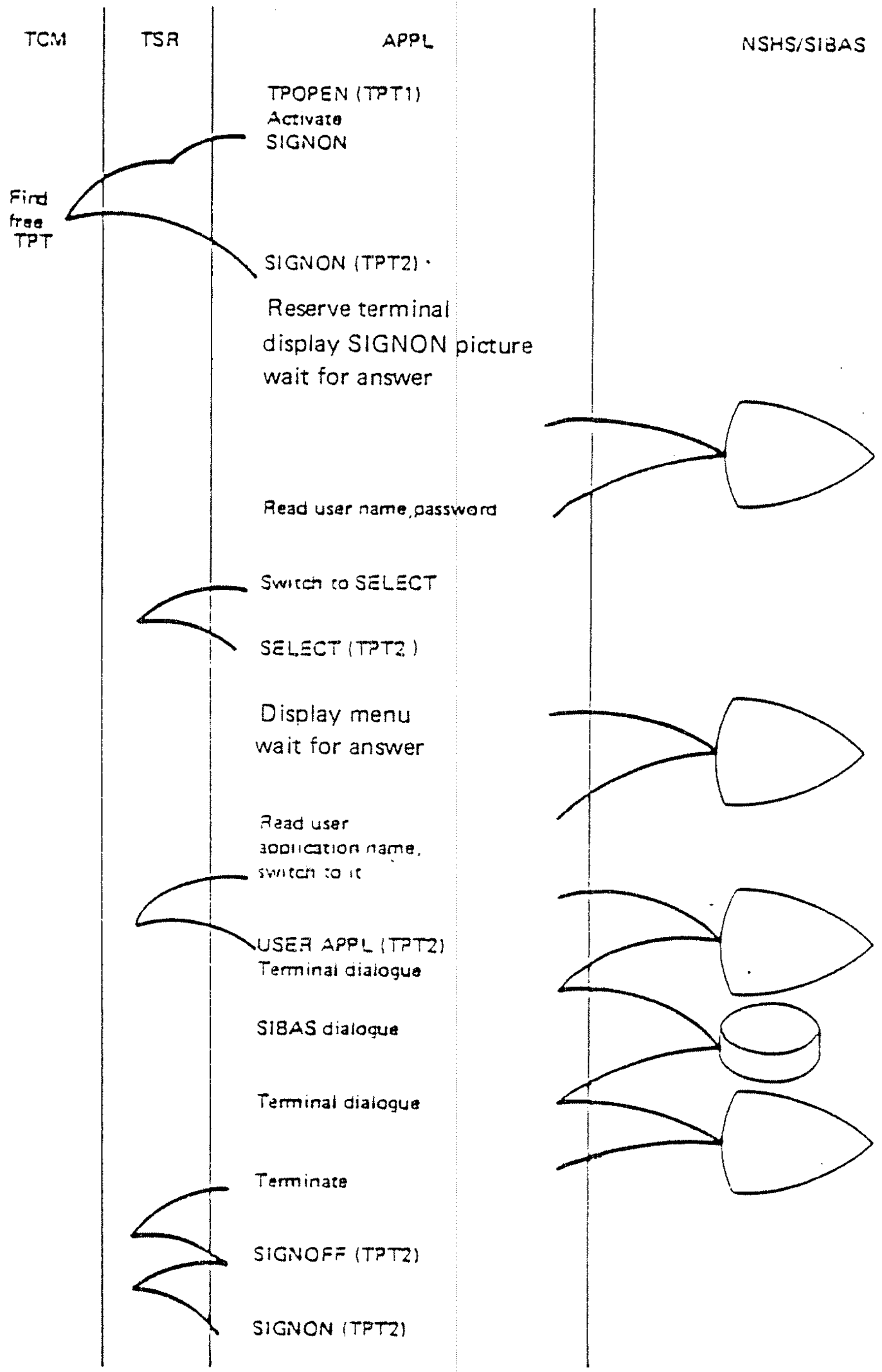


Figure 4.1: A TYPICAL TRANSACTION

4.2 Processing the Transaction

Normally, one may envisage the transaction processing in one or more sequences each consisting of a dialogue between the application program and the user, with some activity on the data base as the result of most dialogues.

The transaction may start by asking for some information, for example an action type. When the user has answered, (for example register a new customer), a dialogue can follow, prompting the user to enter the details of the transaction (name, address, telephone number, account number, etc.). The data base will be updated and the user notified (the customer has been registered). The transaction may then start a new dialogue.

The data base is accessed through normal SIBAS calls in the application program. All of the common SIBAS data manipulation calls are available to TPS programs, such as OPEN DATA BASE, CLOSE DATA BASE, FIND, GET, MODIFY, STORE, ERASE, REMEMBER, FORGET, etc. It is possible to access records from outside the data base (out-of-the-blue access) in several ways, to conduct searches and to access records via their relationships to other records (relative accesses).

The transaction may be an inquiry transaction, an update transaction, data entry, report generation or any combination of these. It may use TSRs to create a session with an I/O device or application program. It may access standard devices by reserving them and communicating directly with them. It may start concurrent or future tasks or switch to another application, again using TSR routines.

4.3 Terminating the Transaction

The transaction will be terminated when the logical end of the program is reached, when the user indicates that there is no more processing to be done, etc. When the transaction terminates, the SIGNOFF application will be activated. SIGNOFF may gather some statistics or do other processing and will end by switching to the SIGNON application.

5 OPERATING THE SYSTEM

A TPS system should be supervised by an operator who has special responsibility for controlling and operating the system. In large systems. the TPS operator may have the supervision of TPS as his only task. In smaller systems, TPS may be controlled by a user or by the SINTRAN system supervisor.

5.1 THE RESPONSIBILITIES OF THE OPERATOR

The TPS operator is responsible for the daily operation of the TPS system. This includes the following:

- starting and stopping TPS
- detecting error situations and taking actions to recover from them
- taking backup copies of the data base and controlling logging facilities
- mounting and changing log tapes
- performing special actions such as starting or stopping TPS modules (I/O modules, TCMs, etc.), starting transactions, terminating transactions abnormally, altering system parameters, running utility programs
- informing users of special conditions and broadcasting messages

Information needed for the daily running of TPS is found in the TPS System Supervisor's Guide.

The operator may also be responsible for generating and installing the TPS system, loading applications and allocating the necessary resources to run TPS. These tasks are also described in the TPS System Supervisor's Guide.

5.2 OPERATOR COMMANDS AND UTILITIES

The *operator communication module* (OPCOM) provides the operator with a set of commands to be used in the daily running of TPS. The command set may be divided into the following groups:

— system start-stop

— system checkpoint-restart

— general commands

— message commands

— status commands

— timing commands

— system logging commands

The system start-stop and system checkpoint-restart commands are known as system directives. System directives consist of the basic functions used in connection with system start, stop, pause, checkpoint, and rollback/recovery. These commands serve vital functions both in connection with normal processing and in case of hardware or software malfunctioning.

The status and message commands are used to control particular transactions, applications, users or modules. These commands are, for instance, used to enable or disable certain applications, start or stop certain transactions, send messages to terminal users, or find out the status of a module, application, or unit (TPT or device).

Timing commands are used to start applications at some time in the future and to set periodic applications. The system logging commands provide an aid for debugging of new software modules in the system by the tracing of individual messages.

5.3 SYSTEM START-STOP

There are several ways of starting and stopping TPS (See Figure 5.1). The first time TPS is started after being loaded

INITIATE-TPS

must be used. This will cause the individual modules of TPS to open the checkpoint files and initiate data areas. The special application TPOPEN will be started to open the data bases and do any other processing desired by the user at this point. For example a message may be sent to all IOM-controlled terminals stating that the system is operational; never-ending transactions may be started. An initial checkpoint is then taken and the system is ready to start normal processing.

The system may be stopped in three ways.

CLOSE-TPS

will initiate a controlled stop sequence. Active transactions will continue until they are done, but no new transactions will be started. Users may be informed of the CLOSE situation by using the BROADCAST command. When all transactions are done, the special application TPCLOSE will be called to close the data bases. System files will then be closed and the system becomes passive.

The CLOSE command will be used when TPS is to be shut down for a while. This may be because TPS is only used for limited time intervals, or because of preventive maintenance such as taking a backup copy of the data base. The normal way to start again is by using the INITIATE command.

ABEND-TPS

will also close the data bases and TPS files and the system will become passive, but it will do this immediately, not waiting for active transactions to terminate. This command should only be used in serious error situations.

HALT-TPS

will stop the system, but the data bases and files will remain open. Transactions will be frozen with their data areas intact. Processing may be resumed again from this point with the command

CONTINUE-TPS

Transaction processing will be started up again without any loss of data.

The HALT and CONTINUE commands are used to temporarily stop the TPS system, possibly to correct a less serious error situation or to gain time to decide how to handle an unusual situation. The halt state should not last more than a few minutes without informing users of the situation.

Two other commands are available for restarting the system, ROLLBACK and RECOVER. Together with the CHECKPOINT command, they are described in the next section.

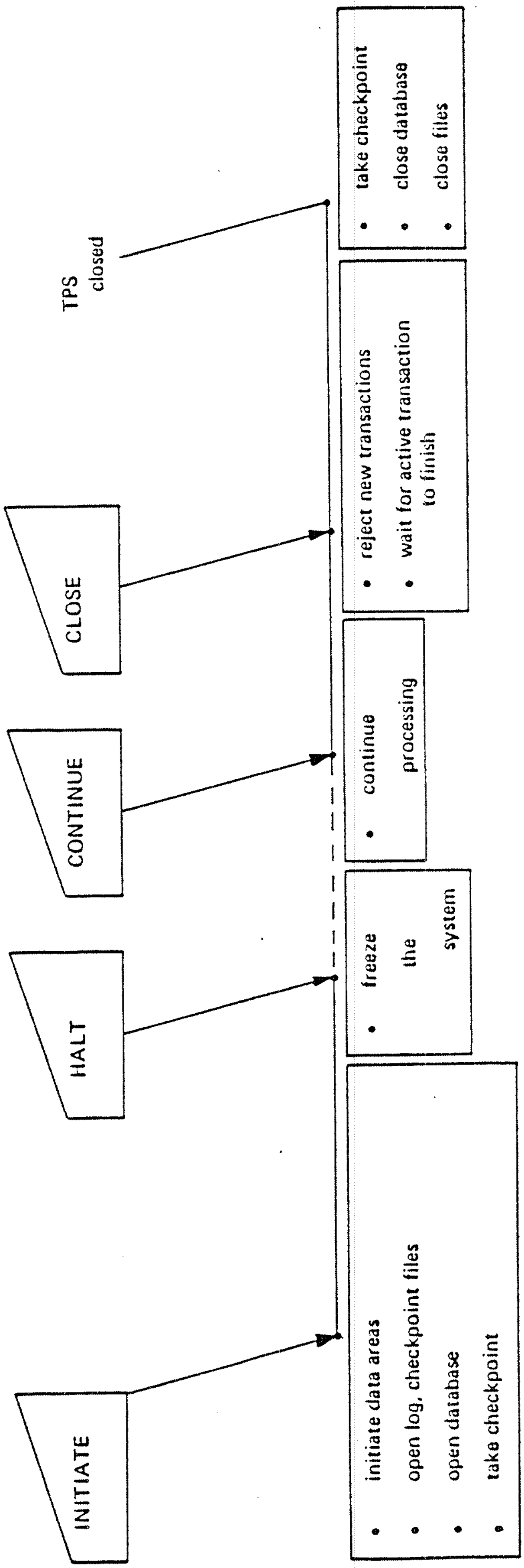


Figure 5.1: SYSTEM START-STOP

5.4 SYSTEM FAILURE-RESTART

It is important that a transaction processing system has the capability of recovering from system failure situations. The data base must not be destroyed beyond repair, as little data as possible should be lost, and transactions should be restarted automatically after data base repair.

The failure-restart features of TPS can be divided into two types, preventive facilities and restart actions.

5.4.1 Preventive Facilities

The preventive facilities of TPS consist of:

- utility routines to take a *backup* copy of the data base
- routines for taking *checkpoints* either automatically, by operator command or by application program
- facilities in SIBAS for updating on a special update file (delayed updating), logging old versions (before-image log) and logging SIBAS routine calls (routine log)

To what extent these facilities are used is up to the user. He decides how often backup is taken, how often checkpoints are taken and what sort of logging facilities SIBAS is to use. He must weigh the advantages of data protection and fast recovery against the increased overhead needed to accomplish them.

5.4.1.1 Backup

A backup copy of the data base is a complete copy of the files in the data base. This can be done with a COPY-FILE command while SINTRAN is running, and even while TPS is running if the data base itself is not being updated (for example if delayed updating is used).

It can also be done using a stand-alone disk utility program. If a stand-alone program is used, TPS should be closed (CLOSE-TPS) before SINTRAN is stopped and the copy program loaded. After this type of copy is made, TPS will normally be started again with INITIATE-TPS (See Figure 5.2).

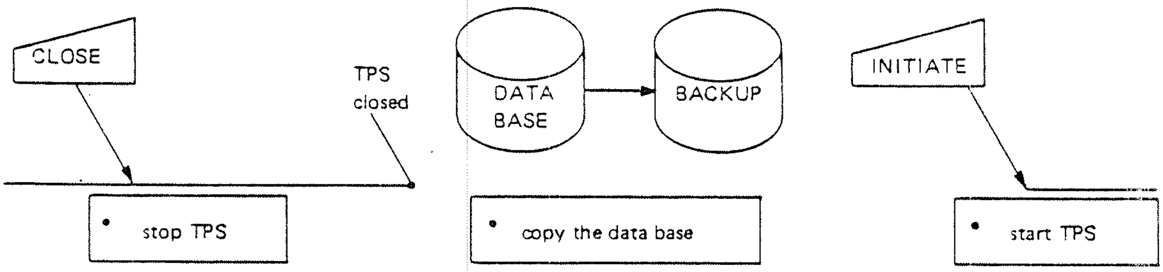


Figure 5.2: BACKUP

5.4.1.2 Checkpoints

Checkpoints can be taken automatically by TPS when the checkpoint load on the system has reached a certain value. Each transaction has a checkpoint weight. These weights are accumulated and when the sum has reached the checkpoint value, a checkpoint will be taken automatically.

When a checkpoint is taken, the individual transactions will be halted when they reach a suitable point (normally in a TSR). When all transactions have been halted, data needed to restore both application programs and system modules to the present state are recorded on the checkpoint files. The special application CHECKPOINT is then activated to send a checkpoint call to SIBAS. When SIBAS has completed taking its checkpoint, processing will be restarted automatically (See Figure 5.3). The whole sequence should not take more than 10-20 seconds.

The operator can also take a checkpoint with the command:

```
CHECKPOINT-TPS
```

This command can be given either during normal running of TPS or in the halted state.

The description above is of a *synchronized checkpoint,*taken by all programs in the TPS system and used in connection with system rollback (See Section 5.4.2.1). In addition, individual transactions can take *transaction checkpoints* at certain stages in transaction processing, such as starting the transaction, reading input and ending an update sequence of the data base. When a system recovery is done (See Section 5.4.2.2), the individual transactions will be started at their last transaction checkpoint.

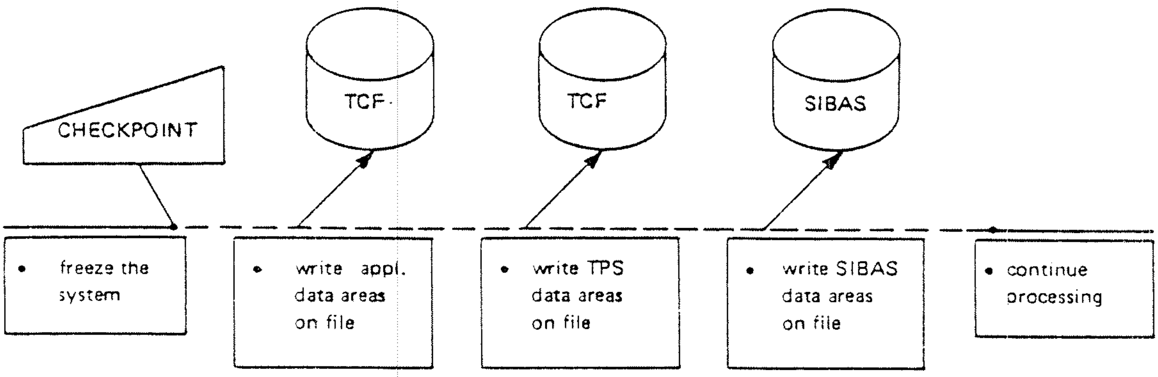

CHECKPOINT
TCF
TCF
SIBAS
• freeze the system
• write appl. data areas on file
• write TPS data areas on file
• write SIBAS data areas on file
• continue processing

Figure 5.3: CHECKPOINT

5.4.1.3 Data Base Logging

SIBAS has three logging facilities available (See figure 5.4):

— Delayed updating consists of the writing of all updated records on a special update file instead of on the data base itself. The data base is only read, not written on during normal processing. It is updated at suitable intervals.This decreases considerably the chance that it will be physically destroyed, it prevents bad data from being written before the actual updating takes place, and it gives a simple checkpoint/rollback mechanism.

— Before-image logging can be used instead of delayed updating. With this method, a page is logged on the before-image area in the SIBAS system realm before it is updated. The data base can then be rolled back by copying all before-images out to the data base.

— The routine log is a sequential disk file containing a record of all calls to SIBAS in the order they were originally received. This file may be used to update the data base from a backup copy or from a checkpoint without having to rerun the application programs.

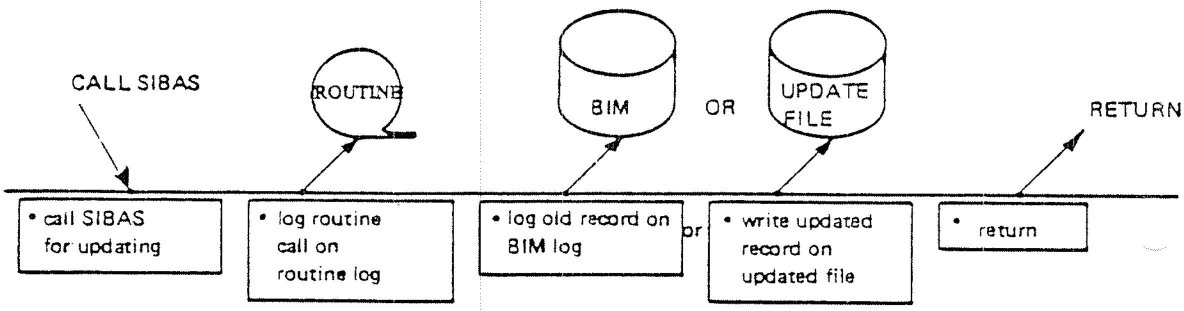

CALL SIBAS
ROUTINE
BIM
OR
UPDATE FILE
RETURN
• call SIBAS for updating
• log routine call on routine log
• log old record on BIM log
or
• write updated record on updated file
• return

Figure 5.4: DATA BASE LOGGING

5.4.2 Restart Facilities

When a serious error is detected, three steps must be taken:

- the cause of the error should be determined and the error corrected
- any damage to the data base should be repaired
- the system should be started again with as little inconvenience to the users as possible

5.4.2.1 Rollback

The operator has two commands available for repairing the data base after a serious error.

ROLLBACK-TPS

will restore the system to its state at the last synchronized checkpoint. SIBAS will execute the rollback either by using the update file or BIM to roll the data base back to the checkpoint or by using the routine log to roll forward to the checkpoint (See Figure 5.5). All programs will then restore their data areas at the checkpoint so their state corresponds to that of the data base.

5.4.2.2 Recovery

The other command

RECOVER-TPS

will restore the data base to its state at the latest transaction checkpoints. A rollback to the last synchronized checkpoint will first be performed, using any of the methods described for rollback. The routine log will then be used to update the data base to its state at the latest transaction checkpoints. Programs will also restore themselves to their transaction checkpoint.

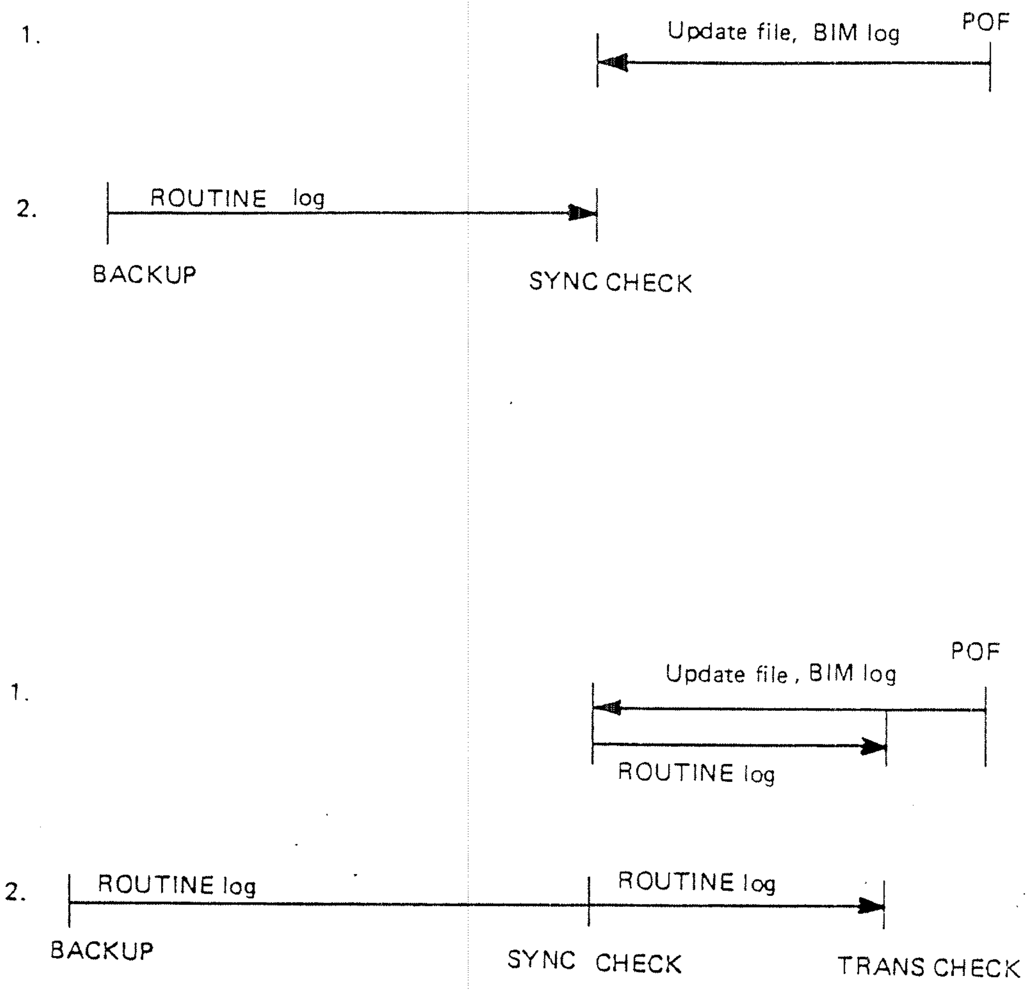


Figure 5.5: ROLLBACK/RECOVERY IN SIBAS

5.4.2.3 Restarting TPS

After rollback or recovery, the system must be restarted again with the CONTINUE command, This can be done automatically by the ROLLBACK or RECOVER special applications or by the operator. When it is restarted, transaction processing will continue as far as possible as though there had been no break.

To what extent this can be done depends mainly on the state of the connections between the application programs and the external environment. Both the data base and the application programs have been restored to their state at the appropriate checkpoint and are ready to continue processing from there. However, terminals and local devices may have been lost and sessions usually have been broken. After rollback, there is the additional problem that external connections at the synchronized checkpoint were probably different from those at the point of failure. The RESTART special application may try to restore broken connections.

Another important consideration at restart is that some transactions may not want to be restarted at the checkpoint. There are several ways of restarting transactions and each must be restarted according to the appropriate restart strategy. This is also done by the RESTART special application.

The RESTART special application and restart strategies are discussed in Chapter 4.

The possibility of defining application-dependent abend- and restart strategies is an important special feature of ND TPS.

Users may also choose to have no restart facilities at all. In this case, a destroyed data base would be restored from the backup copy and the system started with the INITIATE command.

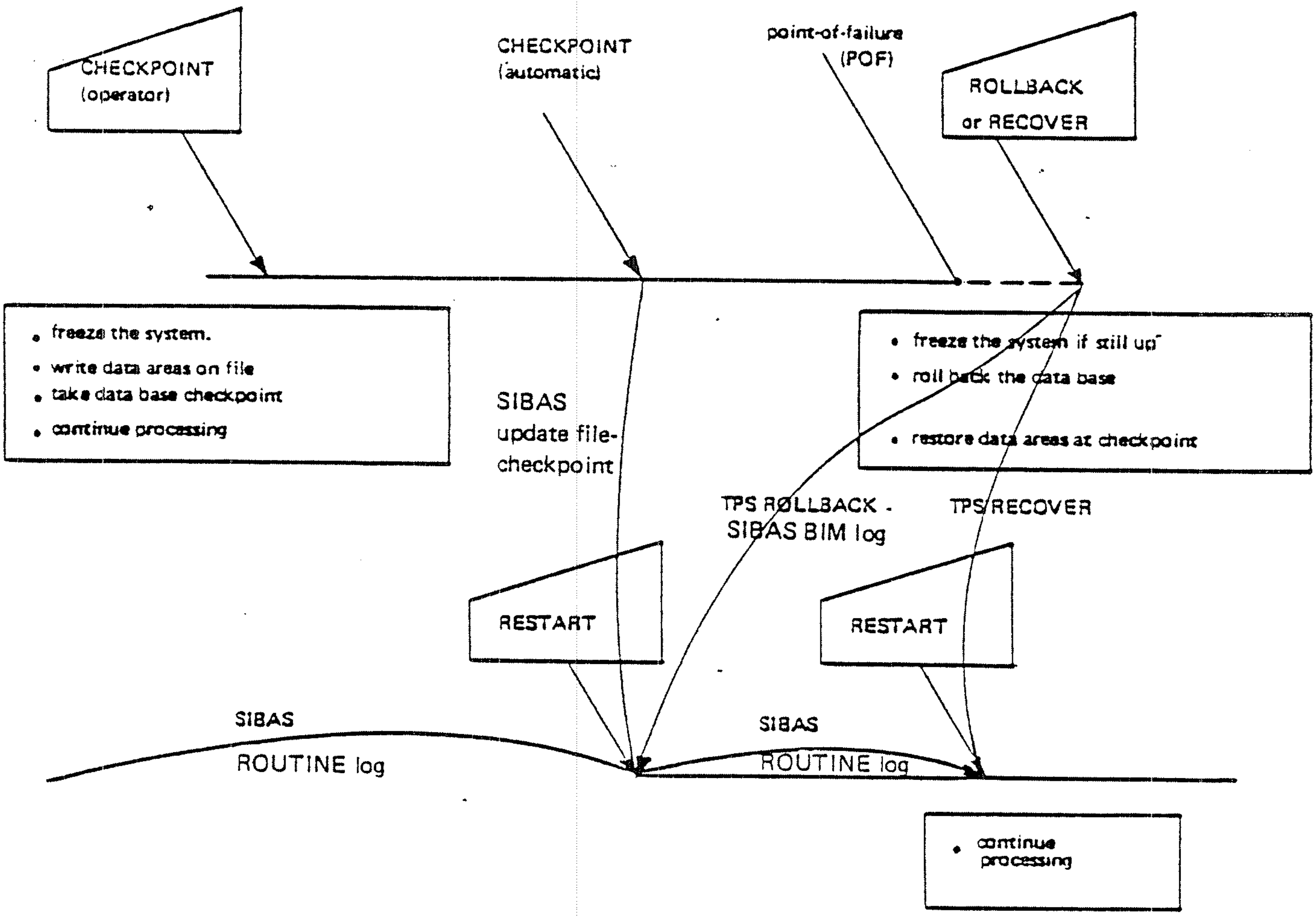


Figure 5.6: SYSTEM FAILURE-RESTART

6 PERFORMANCE AND REQUIREMENTS

6.1 PERFORMANCE CONSIDERATIONS

A number of factors must be taken into consideration when the performance of an on-line system is to be measured:

- average number of concurrent processing transactions
- average number of data base accesses per transaction
- CPU power and size of "working set" versus physical memory
- system software overhead

Usually the first two are determined as requirements from users or system planners at an early stage, while the last two will be chosen after closer study in order to satisfy these requirements.

Some significant characteristics of user applications should be taken into consideration when planning for a ND TPS system.

The following contribute to better performance:

- transaction processing is dominated by a limited number of applications (i.e. optimal use of reentrant code, less paging)
- most of the dominant transactions are of data base read-only type (less data base logging)

The following contribute to a reduced performance:

- large applications with a wide-spread usage pattern
- the most dominant transactions utilize a large data area (this will also influence the time used to make synchronized checkpoints)
- heavy I/O activity with complex formatting of screen pictures, etc.
- a large number of concurrently active transactions

In addition to the considerations above performance may be increased by sharing the load between two or more CPUs. The optimum sharing philosophy must be decided for each configuration, but the following modules could be run in separate CPUs:

- SIBAS DBMS (one data base per CPU)
- Application programs with TCM and TPTs (one TCM per CPU)
- I/O modules

6.2 HARDWARE AND SOFTWARE REQUIREMENTS

The hardware requirements for a TPS system vary greatly depending on the load on the system and the level of performance desired, whereas the software requirements are fairly independent of these factors.

The following is the *minimum* configuration needed to run TPS:

- ND 3200 or ND 3300 series systems consisting of ND 100 with 128K words of memory
- a floppy disk unit
- a 30M byte disk for system software, application programs and data base
- 2 terminals for transaction processing, system operation and program development
- SINTRAN III operating system with the TPS option
- ND TPS transaction processing system
- SIBAS data base management system (not mandatory)
- NSHS and/or FOCUS, the ND screen handling systems
- MAC assembler and NPL compiler
- FORTRAN or COBOL compiler

The configuration may be expanded by increasing the system in the following ways:

- additional ND-500 CPU with cache memory up to 128 kbytes.
- several ND—100 CPUs in one TPS system communicating through HDLC lines
- memory up to 32 Mbytes per CPU
- disk capacity up to 8 300 Mbyte per CPU
- an assortment of peripheral devices, including terminals, printers, card readers, floppy disks, drums, magnetic tape units, paper tape units, plotters, graphic displays
- data communication interfaces to terminal networks
- data communication interfaces to other computers
- data communication interfaces to ND TPS systems in other ND computers

6.3 SYSTEM REQUIREMENTS

The total maximum number of segments available for applications in the ND 100 depends partly on the SINTRAN III configuration, and partly on the other system and subsystems within the same CPU. The following formula gives the theoretical maximum segments available (A) for applications in ND 100 under ND TPS:

A = 254 — (S + OS + TPS)

where: S = number of segments used by SINTRAN

TPS = number of segments used by ND TPS including the TPTs

OS = number of segments used by other systems

(including SIBAS) within the same SINTRAN environment

Example: For S = 82, TPS = 23, OS = 5,

then: A = 254 — (82 + 23 + 5) = 144 segments

In ND 500—TPS, applications in the ND 500 do not require any segments in SINTRAN— III. ND—500 segments is limited by the number of free file—entries under user RT. Each ND 500-segment uses 3 files, and may contain up to 268 Mbytes of program code and data.

Other limitations to the size of a TPS system are:

- The total number of RT-programs (real-time programs) in a SINTRAN III system may be limited if the configuration involves a large number of devices and options. This may reduce the maximum number of concurrent transactions, since each transaction (TPT) uses an RT program description.
- The application may freely utilize the file system, but no recovery will be provided for files opened in addition to the SIBAS files. The maximum number of files that may be concurrently opened for RT-programs running in a SINTRAN III system is limited to 30 files. The exact number depends on whether the files are opened as sequential or random files. SIBAS files as well as spooling files (used by the SINTRAN spooling system) are included in this number.

Memory requirements for a ND TPS installation are listed below. All numbers represent kilo words (1000 words-Kw). Memory resident modules need space in main memory, whereas the others only need space on the disk (plus approximately 5-10% paging space in main memory).

MODULES	*NON-RESIDENT* *(disk)*	*RESIDENT* *(main memory)*
TPS		
RT common with buffer pool		4-20K
Reentrant routines (TSRs, NSHS, etc.)	12-28K	
TPTs and application data areas	3-20K per TPT	
Applications and subroutines	2-30K per appl.	
TPS system modules	20-34K	
Sum TPS	50K+	4-20K
SIBAS		
Data manipulation	40K	
Administrator and initiator	100K	
Work area		10-32K per data base
Sum SIBAS	140K	10-32K

SINTRAN		
Monitor,file system, etc.	500K	40-50K
Background processors (timesharing and batch)	68K per processor	
Sum SINTRAN	500K→	40-50K
Sum of modules	700K→	54-102K
Paging space (5-10% of non-resident)		36K→
Minimum space needed	700K→	70K→

FILES

TPS Modules, etc.	700-2000K
TPS (TCF)	250-2000K
SIBAS data base	10K→
SIBAS logging	0K→
SINTRAN scratch files	64K per background processor
SINTRAN subsystems	0K→

6.4 TPS SEGMENT STRUCTURE IN ND—100

In the drawing, the dark line shows how the 64K address space is used when application programs are running. The application program goes from 0 to P2, the TPT plus user data area from P2 to P3, the reentrant segment from P3 to P1.

The other lines show the segments used for:

— reentrant segments, from 1 to 9 (R = 1 in example)

— TPT segments (T = 6 in example)

— special and test application segments (S = 5 in example)

— user application segments (U = 7 in example)

— system segment (1), OPCOM segment (1) and scratch segments (2)

Total number of segments:

4 + R + T + S + U (23 in example)

In addition, a TPT segment is shown in more detail.

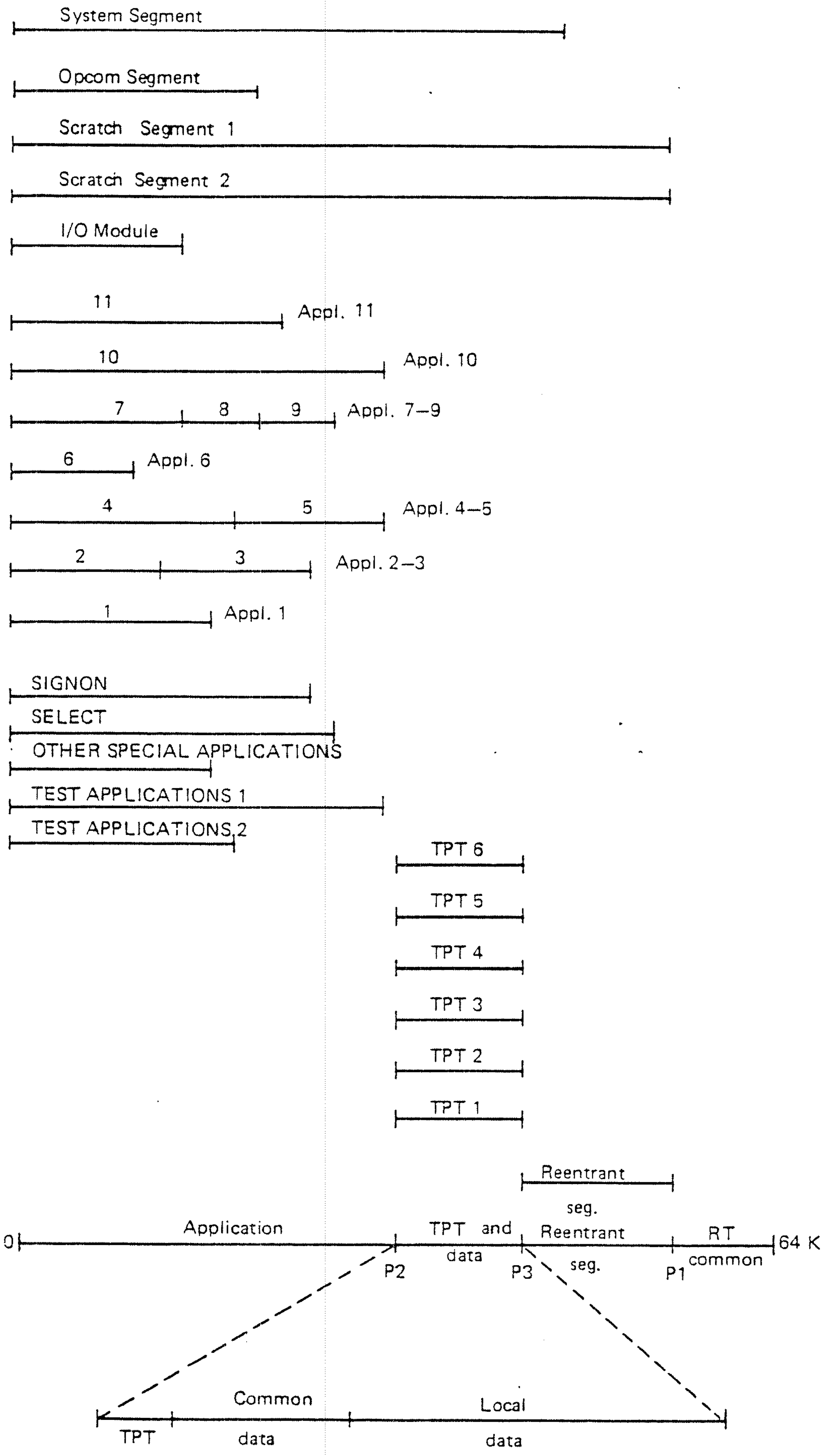


Figure 6.1: TPS SEGMENT STRUCTURE

GLOSSARY

application - a program run under the control of TPS to do the actual processing of a transaction

application, special — see special application

backup — a copy of the data base, taken regularly and used to restore the data base if it is destroyed

batch — the processing of data that has been collected over a period of time to be processed later in a single run of the application program - done in SINTRAN by using the SINTRAN batch processor

BIM log — SIBAS before-image log, a log of SIBAS records that have been changed, logged before the change is made, in order to be able to roll the data base back to a previous state

checkpoint — the saving on a file of all data used by a program in order to later be able to restore the program to its state when the checkpoint was taken

conversational — a program with the ability to carry on a dialogue (see dialogue)

delayed updating — the writing of updated records on a special update file, thus preserving old versions and giving the possibility of restoring the data base to a previous state - done by SIBAS

devices,standard — devices with which an application program may communicate directly through SINTRAN

devices, special — devices with which an application program must communicate through a TPS input/output module

dialogue — the exchange of messages between an application program and a user at a terminal, each message depending on the answer received from the partner

DML — the SIBAS data manipulation language used by an application program

DRL — the SIBAS data definition/redefinition language

failure, system — see system failure

FOCUS — the ND screen handling system (replaces NSHS)

input/output module — a TPS module that communicates with devices such as networks and special terminals externally and TPS modules internally, an interface between the devices and application programs

interactive — a direct connection between a user and a program so that immediate interactions are possible (see on-line, dialogue)

IOM — see input/output module

MD — the TPS main dispatcher module used in message switching between TPS modules

network — a group of communication devices, such as terminals and concentrators, connected by communication lines, and connected to a computer through a modem device

NSHS — the NORD screen handling system used to control display terminals

on-line — a direct and usually permanent connection between a computer and a peripheral device allowing the device to be used without any operator intervention

OPCOM — the operator communication module of TPS containing commands for controlling TPS and routines for sending messages to the operator from TPS modules

point-of-failure — the state of the TPS system and application programs when a system failure occurs. See system failure.

point-of-restart — the state of the TPS system and application programs after a system restart procedure has been carried out. See system restart.

recovery — the process of restoring the system after a system failure by rapid updating from a checkpoint or backup copy. The point-of-restart after recovery is at the last transaction checkpoint

reentrant — the facility of a program to be used by several users concurrently. Each user has his own data area, but only one copy of the program itself is needed.

restart, system — see system restart

rollback — the process of restoring the system to its state at a checkpoint, by undoing the updating after the checkpoint. The point-of-restart after rollback is at the last synchronized checkpoint

routine log — log of all SIBAS routine calls, used after system failure for data base recovery

SELECT application — a special application used by a never-ending TPS task to determine which user application is to be used and switch to the application

session — a connection between an application program and another application program or a device controlled by an IOM. Communication between them is through the TPS message routing system

SIBAS — the data base management system used by TPS

SIGNON application — a special application used to start a transaction by controlling the terminal user's identity

SIGNOFF application — a special application that is given control when a transaction terminates

SINTRAN — the NORD operating system, supporting real time, timesharing and local and remote batch processing

special application — application programs supplied with TPS and used to perform standard functions, such as SIBAS system calls, signon and transaction restart after a system failure

synchronized checkpoint — a checkpoint taken by all TPS and application programs at the same time

system directives — a set of TPS operator commands to perform system control functions such as start, stop, checkpoint, rollback, recovery

system failure — an error situation resulting in the inability of TPS to continue normal processing. The data base may or may not be intact, but transaction processing will be interrupted in both cases

system restart — the process of starting the TPS system after a system failure, including repairing damage to the data base, restoring the data base to a consistent state and restarting transaction processing

TCM — the transaction control module of TPS used to control the allocation of TPTs

timesharing — the use of a computer by several users at the same time by giving each user in turn control over the computer for a certain length of time — done in SINTRAN through the SINTRAN background processor

TPT — see transaction processing task

transaction — an interaction between one or more I/O devices and an on-line data base, usually involving a dialogue between a user at a terminal and an application program, resulting in some activity on the data base and a response to the user

transaction checkpoint — a checkpoint taken by an individual application program at suitable points in processing

transaction processing system — an on-line computer system providing the facilities needed for the immediate access to a data base either to update the data base or to retrieve information from it transaction processing task — a TPS unit allocated to a transaction when it is started to control the application program and provide it with a data area and an interface to TPS

transaction service routines — TPS routines called by an application program to perform functions such as communicating with I/O devices, and administrating task control

TSR — see transaction service routine

update file — a file that SIBAS writes updated records on instead of updating the data base directly. The data base itself is updated at suitable intervals

ALPHABETICAL INDEX

* * * * * * * * * * SEND US YOUR COMMENTS!!! * * * * * * * * * *

Are you frustrated because of unclear information in this manual? Do you have trouble finding things? Why don't you join the Reader's Club and send us a note? You will receive a membership card - and an answer to your comments.

Please let us know if you

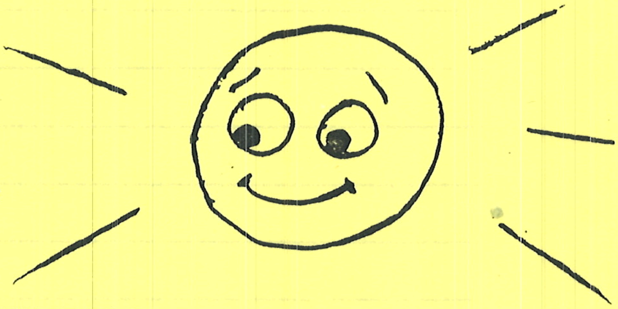
* find errors

* cannot understand information

* cannot find information

* find needless information

Do you think we could improve the manual by rearranging the contents? You could also tell us if you like the manual!!



* * * * * * * * * HELP YOURSELF BY HELPING US!! * * * * * * * * *

Manual name: ND TPS Genral Description

Manual number: ND-60.105.03

What problems do you have? (use extra pages if needed) ____________________

____

____

____

____

____

____

____

____

____

Do you have suggestions for improving this manual? ____________________

____

____

____

____

____

____

____

Your name: ____________________ Date: __________

Company: ____________________ Position: __________

Address: ____________________

____

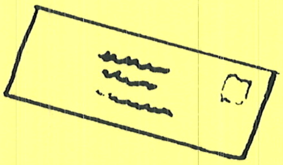
What are you using this manual for? ____________________

____

____

Send to:

Norsk Data A.S.
Documentation Department
P.O. Box 4, Lindeberg Gård
Oslo 10, Norway



Norsk Data's answer will be found on reverse side

Answer from Norsk Data ______________________________

____

____

____

____

____

____

____

____

____

____

____

____

Answered by ______________________________ Date ______________

- -

Norsk Data A.S
Documentation Department
P.O. Box 25, Bogerud
0621 Oslo6, Norway

Systems that put people first

NEC