CHD

Title:

TELEOP

A/S REGNECENTRALEN

RC SYSTEM LIBRARY: FALKONERALLE 1 DK-2000 COPENHAGEN F

RCSL No: 21-T002
Edition: December 1977
Author: Benny Hammarkvist

Keywords:

RC8000, TELEDATA, SYSTEM80, MULTI-ACCESS, ON LINE, REAL TIME TRANSACTION PROCESSING, REAL TIME DATA BASE SECURITY, PROCESS COMMUNICATION, DUETCOM, ALGOL, DUET, SODA.

Abstract:

TELEOP is an on line operating system, handling multi-access to a common data base from several terminals. It is an algol framework for the DUET-and SODA-systems, handling data base security by logging updated records. This manual describes the transaction handling, security system, communication via DUETCOM, operating instructions, error messages and procedure descriptions.

English edition: 86 pages

Printed by A/S Regnecentralen, Copenhagen

Table of contents

Indtroduction

TELEOP is an on-line operating system. It was developed for use in TELEDATA, and previous versions with the same principles have been in routine use since 1971.
The primary functions of TELEOP are:

- administration of terminals
- breakdown security by logging
- framework for the application systems

TELEOP administers multi-acce ss. One database can be accessed simultaneously from several terminals. Terminal specific areas may be accessed also.

TELEOP is used in TELEDATA which is a multi-user service centre system. This does not affect TELEOP which does not operate with the concept user.

TELEOP executes the following preventive security functions, on-line:

- logging of input transactions
- logging of updated/new record images and text blocks
- indication of the state of the database in a special area on the disc:
  being/not being updated.

Thus TELEOP gives the following possibilities for recovery of the database after a breakdown (though the last transaction is always lost):

- continuation of the run (if the database is not being updated)
- repetition of the record updating using the security copy of the database.
- reprocessing using the security copy of the transactions.

TELEOP is an Algol program and is activated under S in a process in RC8000. TELEOP will communicate, on-line, the transactions using another process which contains DUETCOM, this controls input/output to the terminals (and printers).

TELEOP can also be activated off-line, e.g. under the operating system BOSS (then the communication with DUETCOM is not included), this is a very powerful facility both on testing applications and in routine situations.

TELEOP itself does not execute applications. These are programmed in the DUET language. The DUET program is executed by a DUET interpreter which translates the code. The DUET interpreter is an independent part of the Algol program, TELEOP.

TELEOP does not undertake accesses to the database. These are carried out via the SODA system.

Access to the database by the application is defined in the SODA local data description (SODA-LD). The DB accesses are executed by a SODA interpreter, ~~wich translates~~ a SODA-LD. The SODA interpreter is an independent part of the Algol program, TELEOP. The database is described in DATABASE80 and stored on disc. It consists of:

- sequential files
- cfmaster files
- cflist files

Solely text files are also included, they are written using the PRIMULA system. This is included in the Algol program, TELEOP, as external procedures.

For each terminal TELEOP administrates:

- a sequential file

- a text file (replies to the terminal)

The TELEOP manual should be read from the beginning as the sections 2, 3, 4, 5, 6 and 7 are a thorough introduction to TELEOP.

Sections 8, 9, 10, 11, 12, 13, and 14 contain descriptions of individual parts, errors, operating instructions etc.

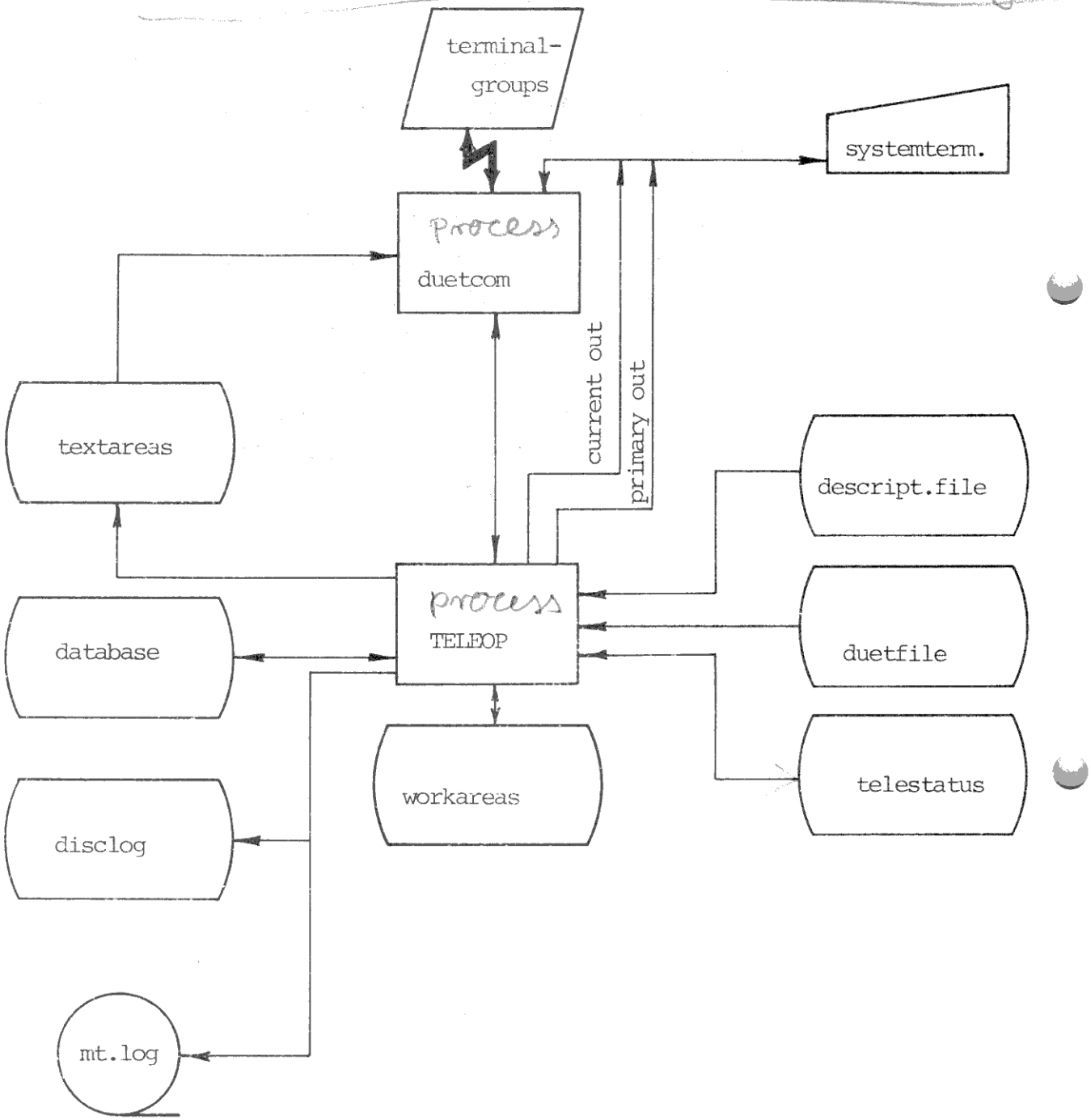
Systems which are used together with or within TELEOP are described in the following manuals:

SODA	RCSL 21-V019
DUET	RCSL 21-V02o
DUETCOM	RCSL 21-T003
PRIMULA	only documented internally
INTRODUCTIONS to TELEDATA	RCSL 21-T001
TELEDATA	RCSL 21-T004

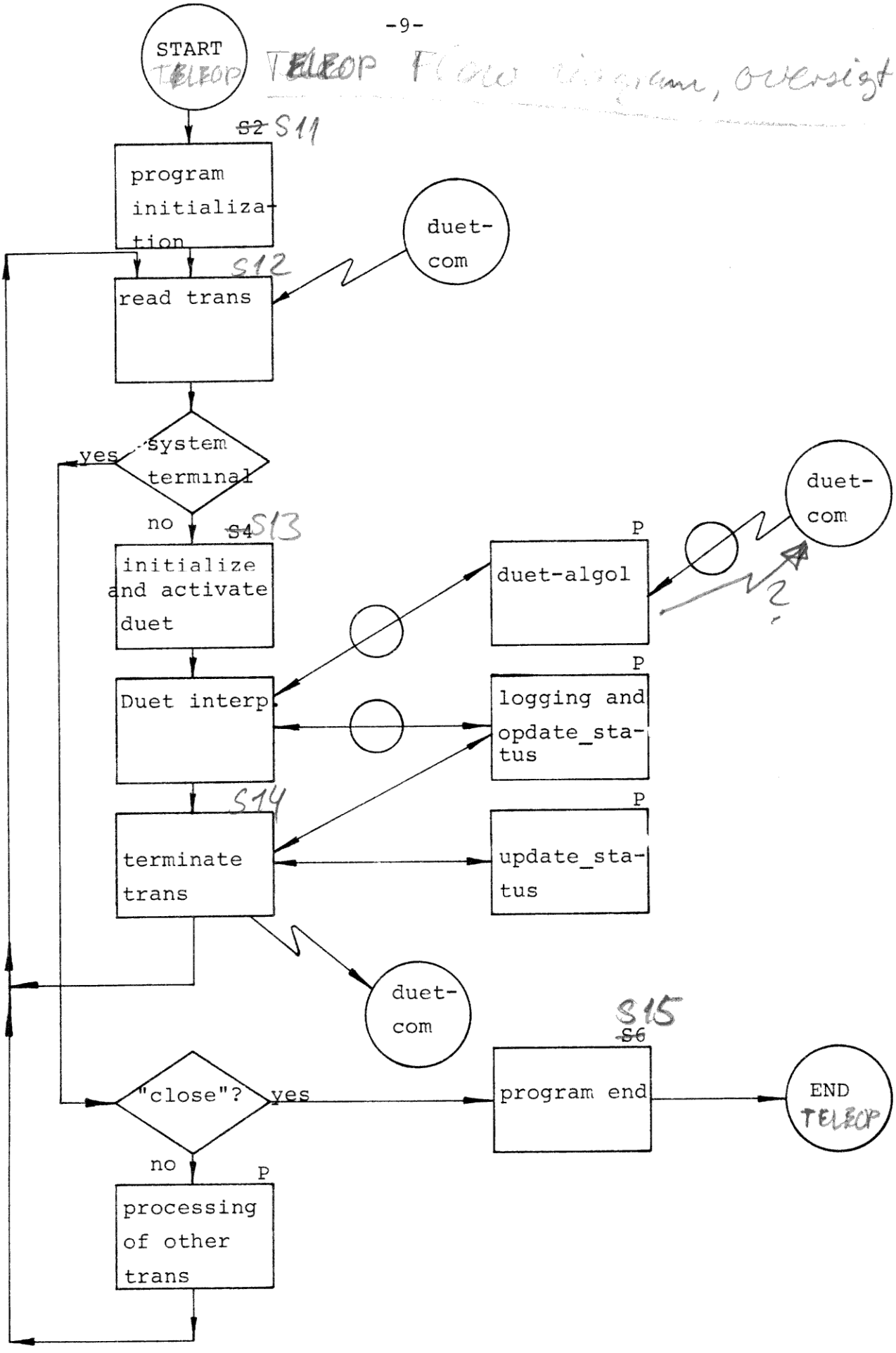
2. Primary functions

2.1 Flowcharts and System survey

Processes and diskareas, survey



TELEOP Flow diagram, oversight

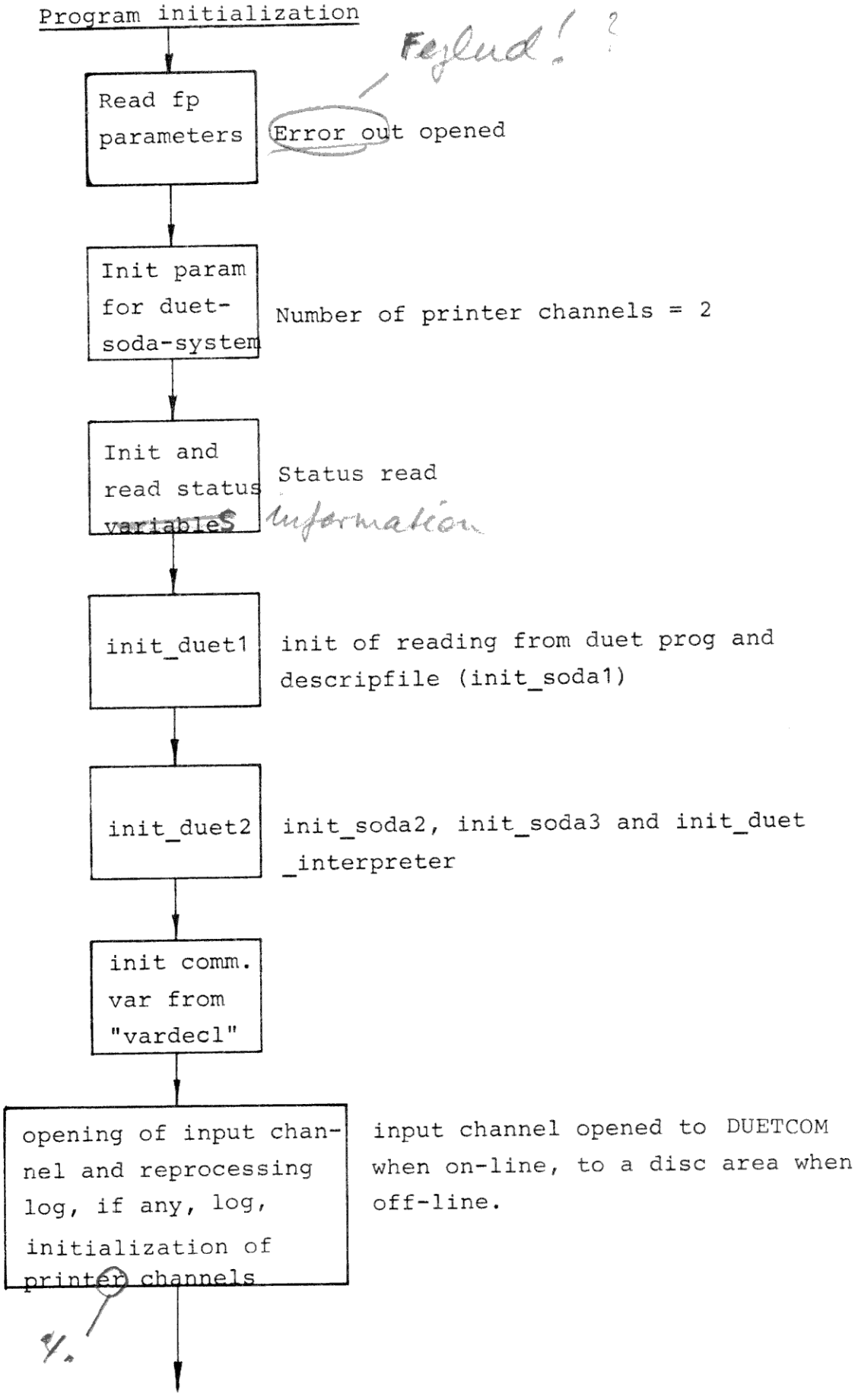


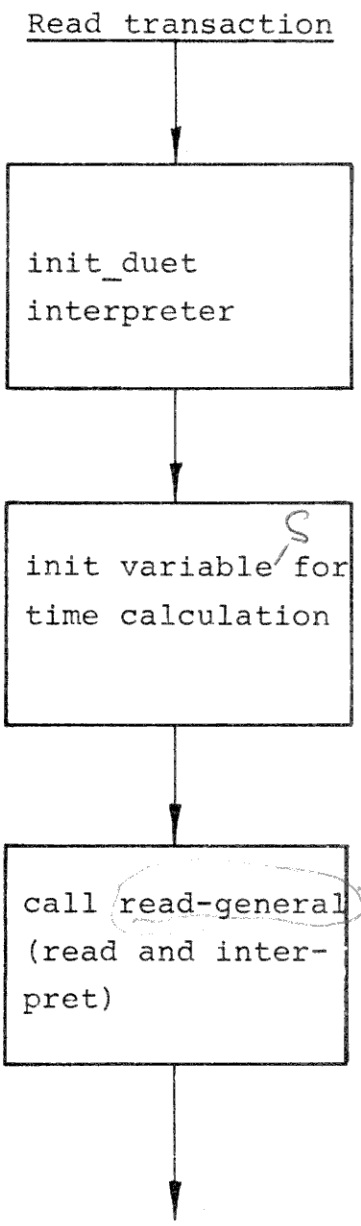
Comments to page 9 :

- a circle on the arrow denotes that activation is dependent on the DUET code being executed.

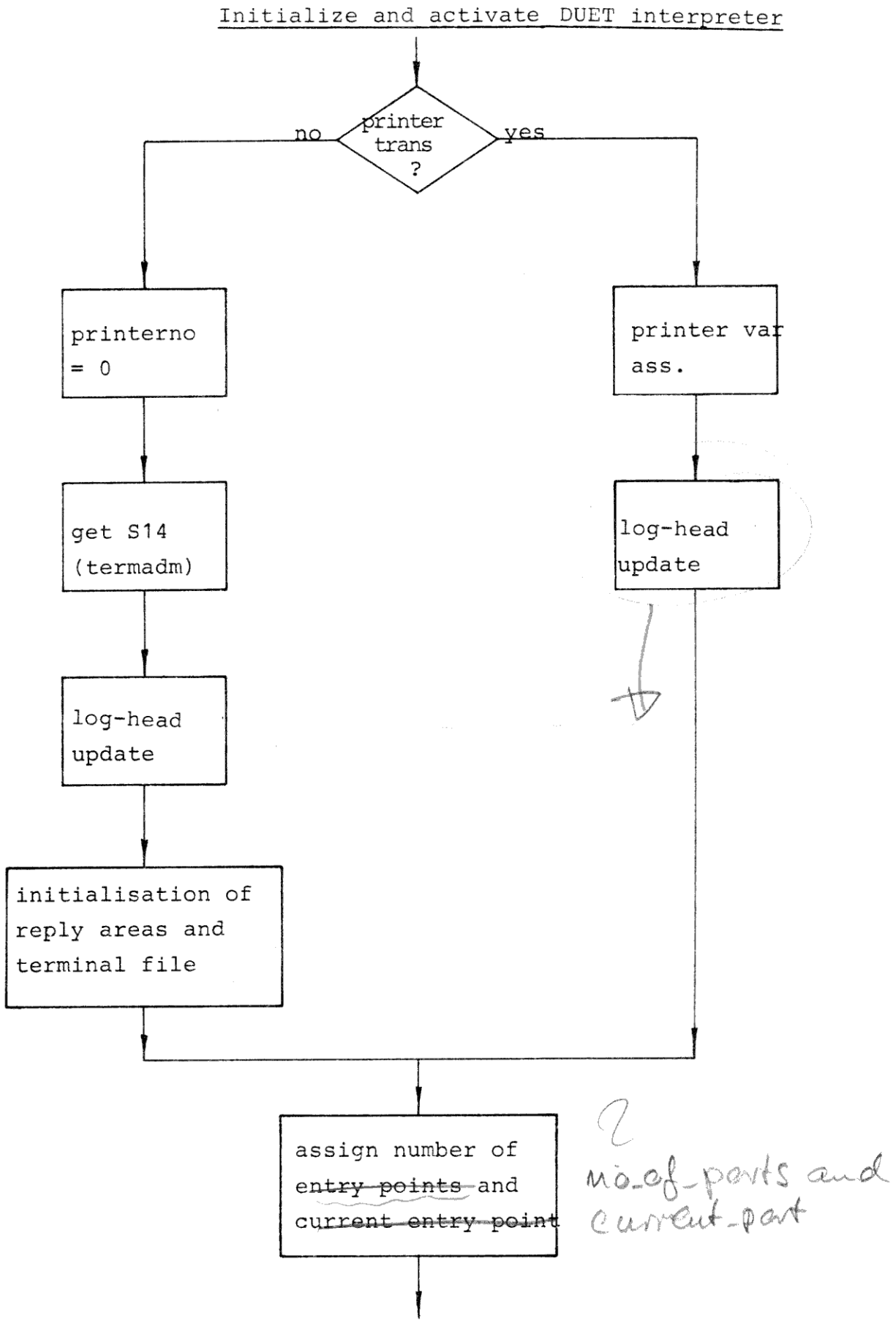
- the reference given beside the boxes either refers to the following pages with more detailed flow-charts, or to the procedure descriptions etc.

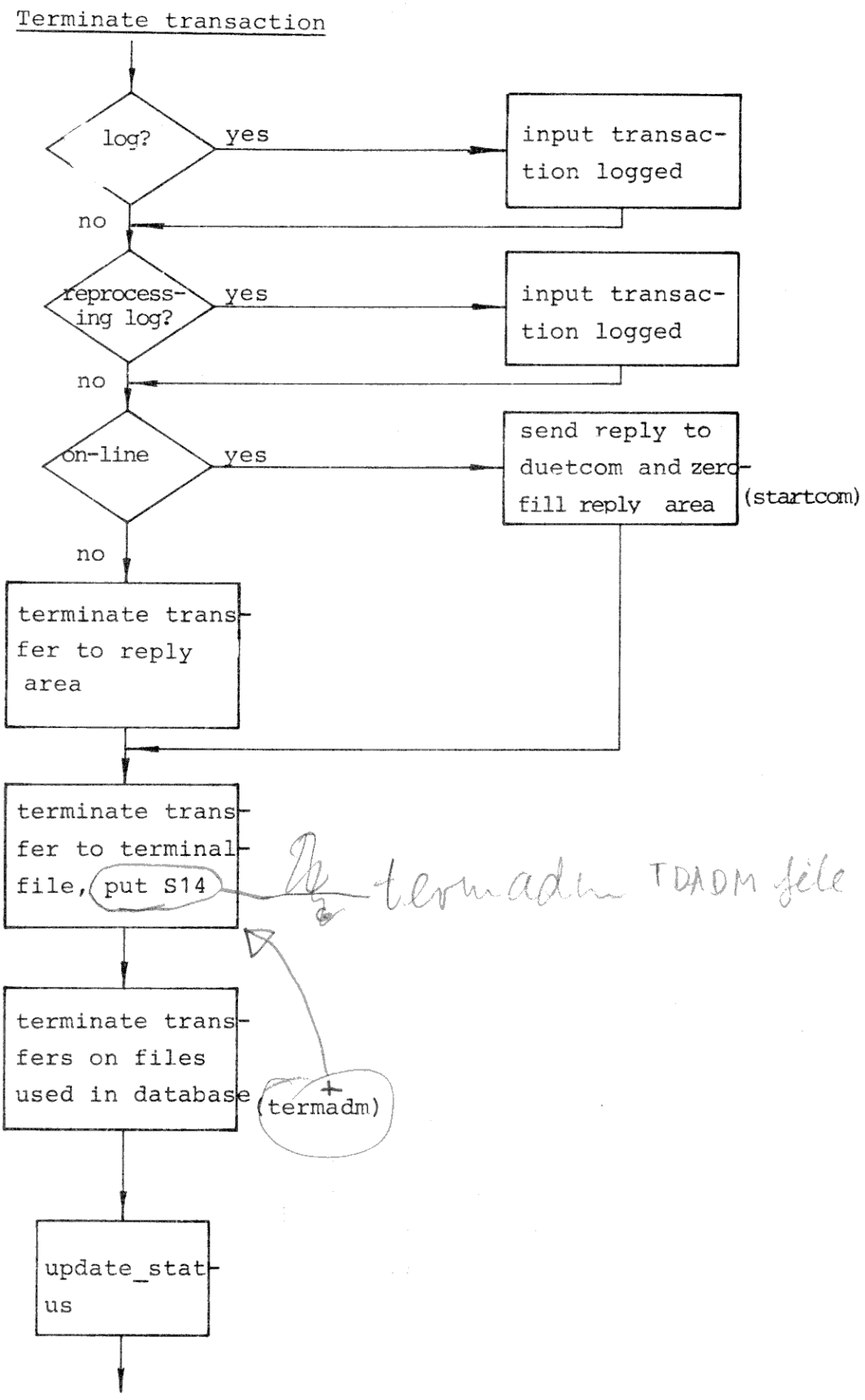
- the flowchart mainly shows the functions on-line. When off-line there is no communication with DUET-COM, and the reply areas etc. are processed in a different way.

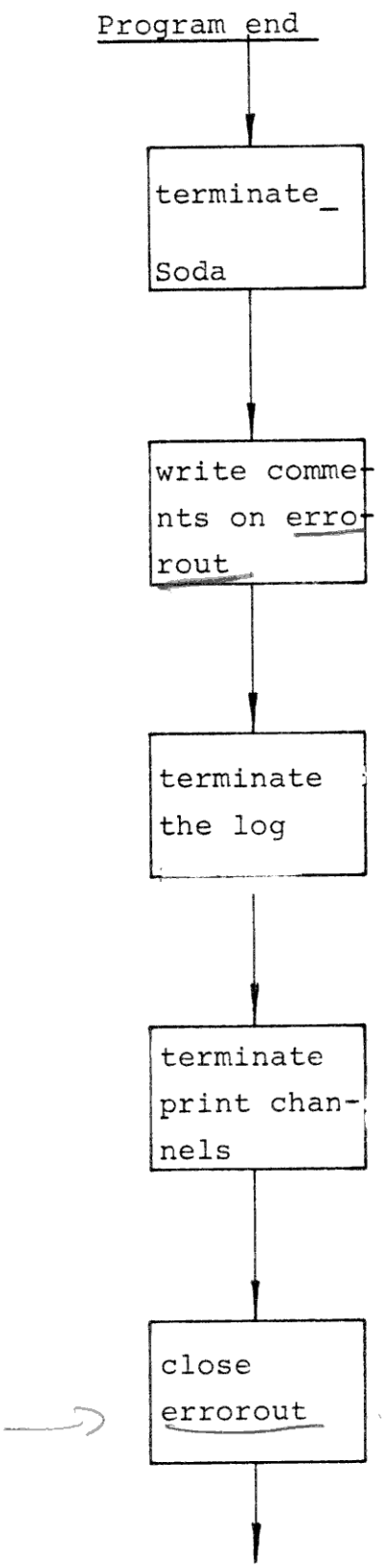

Program initialization
Read fp parameters
Error out opened
Init param for duet-soda-system
Number of printer channels = 2
Init and read status variables
Status read
init_duet1
init of reading from duet prog and descripfile (init_soda1)
init_duet2
init_soda2, init_soda3 and init_duet _interpreter
init comm. var from "vardecl"
opening of input channel and reprocessing log, if any, log, initialization of printer channels
input channel opened to DUETCOM when on-line, to a disc area when off-line.



Beskrivelse!?
(reference?)


Initialize and activate DUET interpreter
printer trans ?
no
yes
printerno = 0
printer var ass.
get S14 (termadm)
log-head update
log-head update
initialisation of reply areas and terminal file
assign number of entry points and current entry point
no_of_parts and current_part


Terminate transaction
log?
yes
input transaction logged
no
reprocessing log?
yes
input transaction logged
no
on-line
yes
send reply to duetcom and zero-fill reply area
(startcom)
no
terminate transfer to reply area
terminate transfer to terminal file, put S14
termadm TDADM file
terminate transfers on files used in database
(termadm)
update_status


Program end
terminate_
Soda
write comme-
nts on erro-
rout
terminate
the log
terminate
print chan-
nels
close
errorout

2.2 Read a transaction

Indud

TELEOP reads transactions from DUETCOM via the communication zone indud.
This excludes the use of the duet system reading facility - getline. When reading the transaction line TELEOP uses the external procedure, readgeneral, which makes the transaction read available for the read operation in the duet system.

Reading in is carried out according to the following algorithm:

brugås af process-diagrammer

```
NEXT_TRANS
   wait for trans.
   read in trans.
   init duet variable
   activate duet interpreter
   trans. termination
   if not close go to NEXT_TRANS
   programend
```

i stedet:
hvad sker der ved init duet interpreter
Hvad indstilles

+ KRAV tel transaktionen

Afsnit mangler:
System terminal (-transaktioner)

2.2.1 Define transaction type

TELEOP distinguishes between two types of transactions, namely:

lines input from a terminal and transactions generated by DUETCOM

Printer operation

Transactions generated by DUETCOM will either contain data on the result of a printer operation or be login/logout transactions.

TELEOP can only differentiate between printer transactions from selective printers and the other transactions. Concerning printers and their handling see the DUETCOM manual - RCSL 21-T003 and the Algol operation no. 7 - call convertarea

Terminal admin. record

When TELEOP has read a transaction and discovered that it comes from a terminal, the terminal administration record for the relevant terminal is fetched. Using information in the terminal admin. record the reply channel is connected to the reply area by the procedure open_pos_text - section 13.3 -, the ~~voucherline~~ terminal file for the terminal is opened, then the Duet interpreter is activated.

2.3. Activate the DUET Interpreter

Communication variable

Before the DUET interpreter - RCSL 21-V020 - is activated, a set of conditions must be fulfilled. All communication variables must be sensibly initialised by TELEOP, and readgeneral (see procedure description) must have placed the transaction read so as to satisfy the requirements of the DUET interpreter's reading operation.

Field index

Field index

The DUET interpreter's reading operation uses a pointer - the field index - to check how far it has come in the interpretation of the transaction. As TELEOP has already used the first 2 fields of the transaction it is unnecessary for the DUET interpreter to read them also, thus the field index of TELEOP is made to point at the third field of the transaction - the transaction code. is normally placed here. But the rest of the reading is controlled by the DUET program

Before the DUET interpreter is activated the procedure init_duet_interpreter - RCSL 21-V020 - is called, this puts the DUET interpreter in a neutral state by setting the error counters and error channels etc. to zero, as well as finding the start block in the DUET program.

til read a transact

After process completion in the DUET interpreter, return is made to TELEOP, which takes control and provides a suitable termination for the transaction processing.

Fejl; forklar (s 13) assign entry points ....

2.4 Transaction termination

On transaction termination TELEOP tidies up i.e. output on all files is terminated such that the database is in a well-defined state after the processing of a transaction.

Reprocessing log

A copy of the terminated transaction is made and output to the reprocessing log.

A reply, if any, is sent to the terminal where the transaction originated.

Replyarea Terminalfile

TELEOP ensures terminations and checking of transfers, if any, to the reply area and the terminal file, as well as storing the position in the variable, reserved for the purpose, in the terminal admin. record. The terminal admin. record is output and output on the log is terminated for this

log

transaction with the compulsory transfer of the last block by printing a dummy block.

Telestatus

Finally telestatus is updated. The transfinished is increased by 1, positions in the reprocessing log and the secondary input file, if it exists, are updated, log_record_no. is updated and also the terminal_no. Lastly the updated block in telestatus is forced out into the physical file and TELEOP is ready to read the next transaction.

3. The security system.

3.1 Telestatus

Telestatus is the basic security element for TELEOP in the form of a disc file containing information on the current state of the system.

System catalogue and security catalogue

Telestatus is divided into two sections, the system catalogue and the security catalogue. These two sections differ in their function ; the system catalogue contains information used on start up and the security catalogue is updated concurrently during the run.

3.1.1 The system catalogue

The system catalogue contains information for use by TELEOP in the start up phase, where it is necessary to initialize certain global variables e.g. logmode, supplementary data for logmode, names of log-tapes etc. ; some of the contents of the system catalogue are given below:

- logmode ; - informs which medium the the security log is to be written on - logmode can have one of 3 values:

  1: only mt. log
  2: only bs. log
  3: both mt. and bs. log

- startdate ; - contains todays date which is written in the label record of the log, as well as in each log record. The date must be inserted manually in telestatus using the status maintenance program ~~changestate.~~

Progfunk's: 1 = teledata
2 = telescriptare
3 = telestat (2)

Runmode: 1 = online
>1 = offline

Runstate: 1 = online indq (prio 1)
2 = prioritet 2 - indq

3.1.1 The system cataloque

- no. of log tapes ; - states the number of magnetic tapes available to TELEOP for output of the log if the log fills more more than one tape.

- current log tape ; - informs which number in the series of available log tapes is currently being used for output.

of
mt.-
ame.

Insert in front of line 11:

- safemode ;- a value which determines which degree of security TELEOP must operate with during the actual run. Safemode may have the values 1 through 3, i.e. there is at this time 3 different degrees of safety, explained in details below.

safemode = 1:
statusfile and rpr.log transports are enforced after each transaction meaning that in case of break-down a reprocessing of the once processed transaction 8 is required to start-up again.

safemode = 2:
transports to statusfile and log (tape or/and bs) are completed by enforcement, meaning that a reestablishing is possible.

safemode = 3:
All started datatransports - statusfile, log, database and textfiles - are completed by enforcemnet, this giving the possibility for immediate start-up after a break-down

3.1.2 Security catalogue

The security catalogue contains information on states and area positions which together with the information in the system catalogue form a basis for the evaluation by teleadm as to which program should be activated before TELEOP is begun, whether morning start is in question or restart after a breakdown. It is also left to teleadm after normal shutdown to decide whether the system is in such a state that it is correct to continue the daily maintenance.

The data in the security catalogue which is updated concurrently and read from telestatus on starting up is:

- position in log	- given by segment no, halfword no.
- position in bs. log	- as for log
- transaction no.	- gives the number of transactions processed completely. Zeroed daily
- record no.	- gives the number of records written in the log
- function no.	- not used as yet
- activity no.	- not used as yet

3.1.2 Security cataloque

The above data is read on the start up of TELEOP and updated on the completion of each transaction during the complete run. In the following, data which is only updated during the run and is not read on start up is given. It is divided into two groups, namely variables which are updated at the beginning of a transaction, and those which are updated on completion of a transaction.

By the beginning of a transaction is understood the first time the transaction causes the contents of the database to be altered. Such an alteration leads to the updating of the following two fields in the telestatus security catalogue:

- started_transno ; set equal to the current transaction no.
- started_input ; the transaction read in is copied in this field

On the termination of a transaction the remaining fields are updated in the security catalogue:

- termin_transno ; set equal to the number of the completed transaction
- termin_termno ; number of the terminal where the completed transaction was keyed in
- log_segm_pos ; segment position in the log

3.1.2 Security cataloque

- log_half_pos ; halfword position in the log

- log_tape_filno ; file used for mt. log

- log_tape_blkno ; last block number on mt. log

- segno ; segment position in ~~bs~~. log
  Nprocess

- last_used ; halfword position in ~~bs~~. log
  reprocess

Teleadm

As previously mentioned the data in the system ~~and security~~ catalogue is used by teleadm to evaluate the state of the system and thus decide which program or manual functions should be executed.

A more detailed description of the structure of ~~TELESTATUS plus a completed list of field names and addresses is found in 'telestatus record description'~~.
the Statusfile is found in the description of the TELESTATUS program.

3.2 Log

Log

The log, described by name and position in telestatus, is used by TELEOP to store a copy of each altered database record. By alteration is meant: the amendment of the contents of a field(s) in a record. Further more a copy of each transaction read in is stored.

Log record head

A log record head of ~~38~~ 34 halfwords is placed in front of the log record, this contains data on the type and origin of the log record, plus information about user no., terminal no., transaction no. and record no. in the log. Further details about the log record head are found in the log record description.

Reestablishing

The purpose of creating a log containing a copy of all altered database records is to ensure a speedy reestablishing of the database after a breakdown, with the loss of as few transactions as possible.

The log

Reprocessing

The advantage of using records from the log instead of executing a reprocessing is that records from the log can be copied directly onto the original database, whilst a reprocessing would cause reprocessing of the transactions dealt with previously. The production of unnecessary output data e.g. invoices, order vouchers etc. is also avoided.

3.3 Reprocessing log

The reprocessing log is a primitive measure which ensures that a reprocessing, and thus the reprocessing of all transactions, is possible.

The reprocessing log simply holds a copy of all transactions read in and processed, so that they can be reprocessed in a simple off-line run with TELEOP.

3.4 Data transfers

Data transfers

By data transfers is understood the transfer of data between TELEOP and a file whose contents is altered, thus.

To ensure status after the conclusion of a transaction TELEOP, dependent on safemode - 3.1.1 TELESTATUS -, provides for the termination of all data transfers begun, so that the database is always in a well-defined state after the termination of a transaction.

Reestablishing/reprocessing

This is necessary so that TELESTATUS can decide after a breakdown whether the run can continue immediately or whether a reestablishing/reprocessing should be executed.

Transfers are terminated on:

- cf files
- bs files
- log
- reprocessing log
- reply areas
- and always TELESTATUS

4. TELEOP input

In TELEOP only one input channel exists, namely
Indud the zone indud.

This zone can be opened to a process, if TELEOP is running on-line, or to a disc area if it is an off-line run or reprocessing.
Where TELEOP will receive input from is defined in the initialization phase of TELEOP by help of the fp parameter input.<input name> is either the name of a communication process or a disc file.
If TELEOP cannot identify <input name> as describing one of these two possibilities, the run is terminated with a hard error message.

5. TELEOP output

5.1 Current_out:

Is controlled using the fp utility program 'o' - all test output and 'exit end' for TELEOP are written on current_out.

The standard current_out is the terminal from which TELEOP is started. (OPERATOR TERMINAL).

5.2 Primary_out:

Primary_out, also called ~~errorout~~ in the TELEOP system will always appear on the terminal from which TELEOP is started.

On primary_out all the error messages come concerning the system, current information on trans no., processing time and the response time for every 100 transactions. (OPERATOR TERMINAL).

5.3 User_reply:

The reply channel is used, via DUETCOM, to communicate all messages to the user back to his terminal, via a disc area - the reply area - which is connected to the terminal of the relevant user. (USER TERMINAL).

5.4 User_output

Output channel

When required, the user can have the output generated during a run written out using the output channel and DUETCOM. Printer or terminal may be chosen depending on the user hardware configuration (SCREEN).

5. Teleop output

Remark to 5.2

Operator terminal

It is inappropriate to use the operator terminal as system or user terminal, as DUETCOM reserves the system and user terminals on login. When the operator terminal is reserved by DUETCOM, TELEOP cannot deliver its output, i.e. log, error messages etc. to this and thus breaks down.

Therefore it is strictly forbidden to log in DUETCOM from the operator terminal !

CURRENT VERSION : DUETCOM does not reserve the terminals!

6. Communication with the system and operator terminal

6.1 Operator terminals

The operator terminal is the terminal (console) from which TELEOP is activated. TELEOP also prints a concurrent log on the terminal about the progress of the transactions.

6.2 System terminal

Broadcast

The system terminal is used for simple operations on the running system cf. systemtrans., e.g assign test-bits, setting up of a terminal record and to 'broad-cast' a message to all logged in terminals. Finally normal shutdown of TELEOP is carried out using the system terminal.

6.3 User terminal

Each user keys his transactions to TELEDATA via a user terminal, after having identified himself, first to DUETCOM, login, with user name and number-, then to the teledata system, -cpw, with a user-defined password.

6.4 Terminal communication with TELEOP

Communica-tion

All communication to and from TELEOP goes through DUETCOM, only the operator terminal is an exception as it is unnecessary to load DUETCOM with the system messages such as log and system error.

7. System transactions

By logging in on the system terminal with 'system oo' it is possible for some so-called system transactions to be executed.

There is a limited choice, namely:

Set up a terminal record

zto Δ <term no.> Δ <user no.> Δ <terminal type> 1 0

assign test variable

testg. <bitx> . <bity>. —— <bitn> 23 0

system close down

close

broadcasting

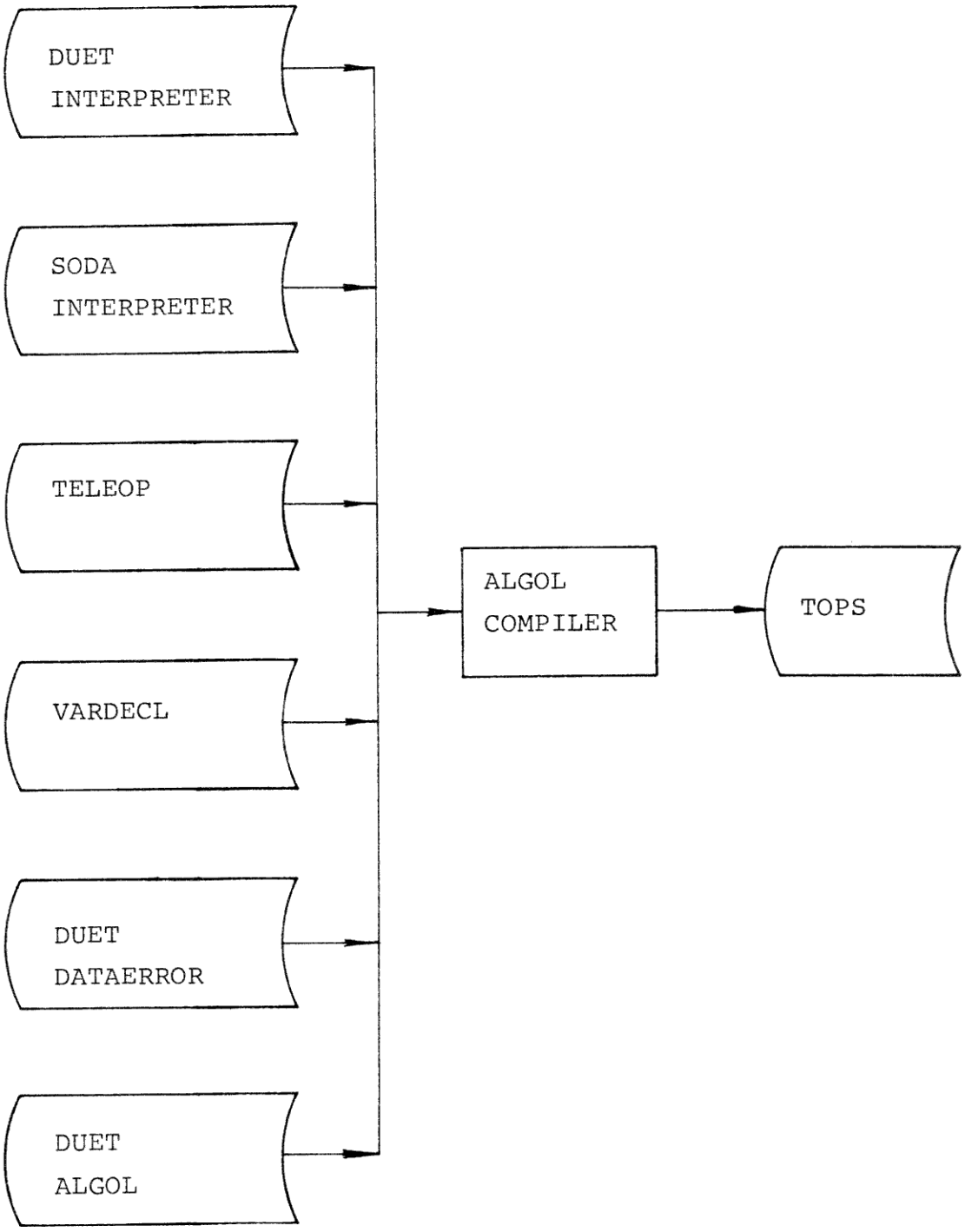
message Δ <any message>

logout

goodbye

8. Compilation

Survey



8.1 Compilation of TELEOP

Example

```
Job teledata 4  <progno> time 8 0 size 800000, area
12 buf 12 perm  <kitname> 500 1, temp disc 2400 24.

i prim; compile print system primula

teleop = set 480 <kitname>
teleop = algol toptexts message:yes
scope user teleop
finis
```

Comment:

In this job it is assumed that many Algol text areas are present; all are delivered with the system, but some may be altered or replaced as a matter of course.

The following texts must never be altered:

prim	:	contains the slang compilation of all the primula system procedures
duettext_1	:	DUET global duetvariable
duettext_2	:	DUET duetprocedures
duettext_3	:	DUET interpreter
sodatext_1	:	SODA interpreter
sodatext_2	:	SODA interpreter
sodatext_3	:	SODA interpreter
ldfields	:	initialization of descripfile
statustext	:	status initialization
testxp 1	:	reading in of fp parameters
teletext	:	Algol 'skeleton'

8.1 Compilation of TELEOP

The following texts are altered as necessary:

spectext : Algol operations, DUET-Algol, here it is possible to add new operations

vardecl : created by the SODA - ld compiler, and altered each time a variable declaration is altered.

duetdataf : printout of DUET data errors. The printed text may be translated from english.

9. Operating instructions

Before starting up TELEOP it is necessary to check that the condition for normal start up are present. The following files mus be present:

- TELEOP !
- DUETFILE (DF)
- TELESTATUS (TS)
- DESCRIPFILE (LD)
- LOG (L)
- REPROCESSING LOG (RL)
- DATABASE (DB) which contains all files (CF, SQ and text files).

consistency between DP and LD checked by checksum on variable part of LD)

Further some conditions for the database must be valid e.g. there must not be an updatemark in the TELEFILES (CF and SQ files), version number agreement between DP and LD (not checked as of now) and TD must have terminated the previous run properly, this being checked in TS ~~by the administration program TELEADM~~.

TELEADM

= by Teleop.

9.1 Terminal dependent files

The reply area

For every terminal which is to have access to normal on-line running on TELEDATA, a file must be set up with the name:

RPLXX where XX is the logical terminal number.

e.g. RPL01
or PRL25

The terminal file

For every terminal there is also a terminal file where voucher lines are stored, sorting is carried out and calculations made. The name of this file is:

WRKFILXX where XX is the logical terminal number.

e.g. WRKFIL01
or WRKFIL25

For use by SODA there must exist a file with the name: TERMAREA at start up.

This file is described in the descripfile as a sequential (sq) file, and SODA uses this description on initialization of the zone(s) to be used later in connection with the terminal file.

Output files

on start up two areas must exist for each described output type, the name format of these areas is not defined as yet but a suggestion as to how it could be formed is given below:

9.1 Terminal dependent files

<voucher type>XX$^{A}_{B}$ 1

Where XX is the logical terminal number, and A or B is included as the user must have the possibility of creating output at the same time as previously created output, if any, are written out. Examples of names for output voucher areas.

```
INVC01A   ;  invoice A - terminal 1
SORD25B   ;  sales order area B - terminal 25
LABL18A   ;  label area A - terminal 18
etc.
```

In the description of the functions and resource requirements for the user, it is possible to decide directly those areas which require creation for this user.

9.2 System dependent files

The database

basic files

Naturally the database containing the ~~logical files~~ belonging to the user must be present at start up.

and the names of the physical files must match the data base description

These files are described in the DB80 description, ~~file and their contents and type must agree with the description~~. This also holds for the terminal file which is used by SODA for opening and dimensioning the zone. (through TELEOP "udskifter" the document to the proper terminal work file - WRKFIL07 for example)

~~DZ~~ STATUSFILE

Telestatus

Telestatus is the file which contains vital information for TELEOP on each start up. The contents of TELESTATUS can be amended at one's discretion by the program, changestate, with various teleop reactions as a result. The name of the telestatus file is optional, as it MUST be given as a fp parameter in the TELEOP call.

TELESTATUS
Changestate

Can be written and updated by the utility program TELESTATUS.

LD description file

The descripfile contains a compiled version of the local data description. This file is required so that SODA can initialize zones and field variables, as well as identify the sets chosen. The name for this file is optional, but if the name:

~~DESCRIPFILE~~ DESCRIPFILE

is used, TELEOP does not need the name as a fp parameter.

- by the SODA "makeup", which initially loads the description. The filename is optionally, and is stated as fp-parameter. Default is: descripfile

9.2 System dependent files

The DUET program

Naturally a file containing the compiled Duet ~~code~~ program must exist, this file name is also optional, ~~but if use is made of the name:~~ and stated as fp parameter. Default: duetfile

DUETFILE

DUETFILE

~~the fp parameter - duetfile~~ <duetprg. name> ~~- can be omitted.~~

The log

Depending on the logmode (see ~~telestatus~~ description) a file must exist which can hold the log.
~~If the name is unknown it can be written out using the program, changestate, which writes the file name from telestatus.~~ If a file with this name does not exist it must be set up and scoped project/user (permanent).
The name

TELELOG

TELELOG

is generally recognised and used in current systems.

9.3 Program call

To activate TELEOP at least two fp parameters are required:

teleop<s>input.<inputname><s>{status}.<statusname>

Fp parameters

An fp parameter consists of a keyword which identifies to TELEOP the data which is to be modified using the following information. The fp parameters are divided into two categories, test output controlling and run dependent parameters. The test output controlling parameters have as a keyword the name of the test-variable which is to be given a value, the other parameters have designations of their function or concept as keywords. In the following all the keywords and their effects are described in detail:

Test output controlling

KEYWORD:	MODIFICATION:	EXPLANATION:
testa. testb. testc. testd. teste. testf. testg. testh.	<bit number> " "	<bit number> is the bit(s) in the test variable to have the value 1 <bit number> can have values from 0 to 23. An arbitrary number of combinations of <bit number> may appear e.g. .1.8.11.6 etc. The above holds for all test vaiables.
Status	<status name>	<status name> is the name of the disc file containing telestatus.
duetfile	<duet prg name>	<duet prg name> is the name of the disc file containing the compiled DUET program

9.3. Program call

KEYWORD:	MODIFICATION:	EXPLANATION:
descripfile	<descrip name>	<ldfile name> is the name of the disc file containing the compiled ld description.
input	<input name>	<input name> is the name of the process - typically DUETCOM - or the file from which TELEOP primarily will read transactions
log	yes / no	decides whether TELEOP must copy records in the log - overtyped status-inf ?
secondary	yes / no	decides whether TELEOP must use idle time to process secondary transactions. For the present this facility is not being implemented.

9.4 Example of a run

```
;   job to initialize TELEDATA's database.
job teleinit <projno> size 100000 time 8 0,
perm <kitname> 2500 30 temp disc 2500 outp 100000,
mode list.yes

;   set up files for use by reorg_80
reorgtable    = set 1 disc
newreorgfile  = set 1 disc 0 0 0 20.1

;   possible existing old files removed

clear user tdadmin debcred items userinf orderref,
      accentry partslist dcstructure,
      orderlines termarea

;   create and scope new files

tdadmin       = set 1 <kitname>
debcred       = set 1     -
items         = set 1     -
userinf       = set 1     -
orderref      = set 1     -
orders        = set 1     -
accentry      = set 1     -
partslist     = set 1     -
dcstructure   = set 1     -
orderlines    = set 1     -
termarea      = set 1     -
```

Kun det for TELEOP vedvendige (relevante) skal med!

Status prim201 repl01 workf01 invo1a
02 02 02 016
(tellestatus)

9.4 Example of a run

```
Scope user tdadmin debcred items userinf orderref,
      orders, accentry partslist dcstructure,
      orderlines termarea,

;  initialization of file heads
descripfile  = reinset

;  create the areas necessary for logical file main-
   tenance of the operator terminal, terminal no. 1
;  and 2

rpl_00   =  copy 0
rpl_01   =  copy 0
rpl_02   =  copy 0

prinz1   =  copy 0
prinz2   =  copy 0

;  create and initialize telestatus

telestatus  =  set 1 <kitname>

Changestate d.telestatus zero.all logmode.2,
        progfunc.1 safemode.3 bslog.telelog

;  the logical file maintenance is run through and
;  creates customer, item and termadm records
;  for test use so that the files are in order
;  for the actual test run

teleop input.reorginput status. telestatus duetfile.,
teleprg descripfile. descripfile
```

9.4 Example of a run

```
;  then the new initialized files are placed
;  at user level ready for use in the
;  following test runs.

clear user t1 t2 t3 t4 t5 t6 t8 t10 t11 t12 t20 t21

t1      = set 1 <kitname>
t2      = set 1     -
t3      = set 1     -
t4      = set 1     -
t5      = set 1     -
t6      = set 1     -
t8      = set 1     -
t10     = set 1     -
t11     = set 1     -
t12     = set 1     -
t20     = set 1     -
t21     = set 1     -

;  move the contents of the initialized files
;  to the safety files.

t1      = move tdadmin
t2      = move debcred
t3      = move items
t4      = move userinf
t5      = move orderref
t6      = move orders
t8      = move accentry
t10     = move partslist
t11     = move dcstructure
t12     = move orderlines
t20     = move termarea
```

9.4 Example of a run

```
scope user t1 t2 t3 t4 t5 t6 t8 t10 t11 t12 t20 t21

;  after this new initialized files are
;  generated which in future can just be
;  moved to the test files if one wishes
;  to begin again

finis

;  example of an off-line test job with at
;  least two terminals
job teletest 1 <projno> size 100000 time 10 0,
area 36 buf 24 perm <kitname> 400 4,
temp disc 3600 24 outp 200000
mode list.yes
; reply and printer areas are zeroed
rpl_00  = copy 0
rpl_01  = copy 0
rpl_02  = copy 0

prinz 1 = copy 0
prinz 2 = copy 0

; create and scope new terminal files and output
; areas for invoicing

clear user wrkfil01 wrkfil02 iv01a iv01b,
        iv02a iv026b

wrkfil_01 = set 1 <kitname> 0 0 0 20.1
wrkfil_02 = set 1 <kitname> 0 0 0 20.1

iv01a     = set 1 <kitname>
iv01b     = set 1   -"-

iv02a     = set 1   -"-
iv02b     = set 1   -"-
```

9.4 Example of a run

```
scope  user wrkfil_01 wrkfil02 iv01a iv01b iv02a iv02b

;  safety copy of the database is moved to testfiles

tdadmin    = move t1
debcred    = move t2
items      = move t3
userinf    = move t4
orderref   = move t5
orders     = move t6
accentry   = move t8
partslist  = move t10
dcstructure = move t11
orderlines = move t12
termarea   = move t20

; create and initialize telestatus

telestatus = set 1 disc
changestate d.telestatus zero.all logmode.2,
bslog.telelog  progfunc.1 safemode.3 rrlog.logexp1


;  introductory actions concluded the test
;  proper can begin

tops input.testinput status.telestatus,
duetfile.teleprg descripfile.descripfile,
testg.0.1.4.7.8.11.12.13.14.18.22
```

testg

```
;  testg controls  various test print-outs - see
;  description

head 1

;  test the results obtained
```

9.4 Example of a run

```
changestate d.telestatus printout.all

;  printout telestatus

list rpl_01 ; printout reply area for terminal no.1
list rpl_02 ;    -       -    -    -     -     no.2

list testinput ;  print the processed transactions

;  print possible contents of terminal files
;  using print80
print80 func.print.all file.wrkfil01.seq listout.out
print80 func.print.all file.wrkfil02.seq listout.out

;  here the test run is terminated, please
;  examine the output BEFORE the next run.

finis
```

10 Hard errors, errors and error messages

10.1 Hard errors

Hard errors are those errors jugded by TELEOP to be so serious that the run is stopped with a hard error message.

10.1.1 Parameter hard errors

All error messages concerning fp parameters start with 3 asterisks.

```
***status name    ; status name missing
***duetfile       ; duet program name missing
***descripfile    ; descripfile name missing
***inputchannel   ; input name missing
```

10.1.2 Other hard errors

If the two vital fp parameters status name and input name are omitted in the call the run is stopped with one - or both - of the following error messages:

```
statusname undefined
inputchannel undefined
```

If TELEOP finds that a disc file with the given status name does not exist, the run is stopped with the following message:

```
statusname <name> does not exist
```

TELEOP also stops the run if either a process or a disc file with the name given by <input channel> is not found, the following message is output:

```
illegal inputspec
```

input < > does not exist

When initializing the log, TELEOP ensures that a file with the name given in TELESTATUS exists. If not the run is stopped with the message:

10.1.2 Other hard errors

log <area name> is not permanent/
does not exist

If TELEOP discovers an error checking the name of the log against the logmode, the following error message is output:
***logname error
logmode set to zero

Setting logmode to zero causes TELEOP to stop the run.

When magnetic tape is used to store the log TELEOP checks that the tape label follows certain conventions. If this is not the case logmode is set to zero.

When log is to be stored in a disc file TELEOP checks that the file exists and is large enough in proportion to TELEOP's start position, if an error is found, one or more of the following error messages are output:
dislog: <log name>
is not permanent/does not exist
***disc log size

then logmode is set to zero.

If TELEOP finds the logmode to be zero the run is stopped with the error message:

.....> ***NB*** ***NB***
logmode = zero

10.2 Errors

An error is defined as being a part of the unusable result of a run, produced on account of incorrect input.

Therefore it is established that the concept, error, is not found in TELEOP

10.3 Error messages

TELEOP is supplied with the posibility of advising the operator about various errors in the processing of a transaction. All these error messages are collected in the TELEOP error procedure - topserror - wich always halts the further processing in question.

10.3.1 Topserror

Error messages in connection with the execution of an Algol operation always start with

```
error in duetalgol - entry x
```

where x gives the number of the Algol operation where the inconsistency was found, information concerning the variables involved is output, together with information on the cause for this error message.
Seen from transaction level the error message is regarded as a hard error.

10.3.2 Other error messages

TELEOP has one error message in connection with fp parameter check, which is:

```
error in logparam.
```

Which means an attempt has been made to adapt the log function, but this is not understandable to TELEOP. TELEOP continues but with output of the log.

10.3.2 Other error messages

An error message may appear from the TELEOP procedure open_pos_text concerning the area to be opend. If the area cannot be found TELEOP tries to create and reserve it. The situation gives rise to two types of error messages:

```
success in creating and reserving
the area

<area name> created on <disc name>

unsuccessful

 creation of <area name> unsuccessful
 create/reserve <area name>
 result: <create>Δ<reserve>
```

In connection with the output of the log, which is monitored by the block procedures, messages may appear concerning errors in the log module.
When a hard error occurs on disc or tape, in cases with parallel disc and tape logs, the following is output on the operator terminal:

```
hard error on tape/disc <tape name>/<area name>
continuation on tape/disc <tape name>/<area name>
number of reces. on tape/disc <no. of recs.>
```

Sort error?
Hvad kommer der på terminalen?

11. Test output.

testg

All test output from TELEOP is controlled by the test variable testg; each bit in testg - which is an integer - controls its own type of test output. An explanation of the output related to the individual bits is given in the following:

Bit 0

The output from the procedure open _pos_text:

```
openpostext: <area name>
position   : <seg no.> ▽ <halfword no.>
```

The output from the procedure close_pos_text:

```
closepostext: <area name>
position    : <seg no.> ▽ <halfword no.>
```

Bit 1

The output from the procedure start_com:

```
start testout from start_com
start_com - primulazone
end testout from start_com
```

The output from start-areaoutput:

```
convert_area started
end convert_area
result <result>
```

The output from the procedure wait:

```
wait: time <sec> secs.
```

11. Test output.

The output from the procedure comend:

```
comend ent.
comend exit.
```

Bit 2

Bit number 2 of testg is used to activate the procedures outzone and outshare in those places where appropriate, e.g. in the procedures startcom, open_pos_text, indudproc and after the initialization of the primula zones.

Outzone and outshare are testprocedures which print, element for element, the contents of a zone descriptor and a share descriptor respectively. These facilities are useful in cases where one requires to know exactly what a zone is used for e.g. on running in a process communication.

Bit 3

Prints the length of the primula system's administration buffer, stated in halfwords:

```
primbufflength = <buff length in halfwords>
```

Bit 4

Prints, as a string of text, the transactions read in:

```
read through indud <transaction.....>
```

11. Test output.

Bit 5

If bit 4 is set, the transactions read in are printed as character values:

```
<ch1> <ch2> <ch3>....   ....<chn>
```

Bit 6

Prints DUETCOM generated printer transactions:

```
printertrans.: <transaction>
<trans.code>
```

Bit 7

Controls the test output from Algol operation no. 1 - sorting:

```
start testout from duetagol - entry 1
paramno.   : <paramno.><paramvalue>
inp.file   : <input filename>
outp.file  : <output filename>
disc.name  : <documentname>

coresort/discsort entered
sorted recs. are placed in: <filemane>
end coresort/discsort

position after sort:
segmno.    : <segment no,>
wordaddr.: <word no.>
end testout from duetagol - entry 1
```

11. Test output.

Bit 8

Controls the test output from Algol operation number 2 - create area:

```
start testout from duetagol - entry 2
create area: <area name>
on disc    : <kit name>
end testout from duetalgol entry 2
```

Bit 9

Controls the test output from Algol operation number 3 - alphanumeric key (text cruncher):

```
start testout from duetalgol - entry 3
key in : <alphakey text>
key out: <numeric key>
end testout from duetalgol - entry 3
```

Bit 10

Controls the test output from Algol operation number 4 - time measurement:

```
start testout from duetalgol - entry 4
cpu base  : <timebase in sec>          ; return
real base : <timebase in sec>          ; return
call base : <timebase for realtime>    ; call
cpu.time measured : <cpu base in m.sec>     ; return
real time measured: <real time used in sec> ; return
end testout from duetalgol - entry 4
```

11. Test output.

Bit 11

Controls the test output from Algol operation no. 5 - call open_pos_text:

```
start testout from duetalgol - entry 5
channel no.      : <channel no.>
block, halfword : <seg.no>Δ<halfword no.>
areaname         : <area name>
end testout from duetalgol - entry 5
```

Bit 12

Controls the test output from Algol operation no. 6 - call close_pos_text:

```
start testout from duetalgol - entry 6
channel no.      : <channel no.>
returnpos.       : <return position>
areaname         : <area name>
end testout from duetalgol - entry 6
```

Bit 13

Controls the test output from Algol operation no. 7 - call convert_area:

```
start testout duetalgol - entry 7
areaname         : <area name>
printerinf.      : <printer information>
forminf.         : <form information>
end testout from duetalgol - entry 7
```

11. test output.

Bit 14

Controls the test output from the Algol operation no. 8 - prepare convert by rename area:

```
start testout from duetalgol - entry 8
old areaname     : <area name>
rename to        : <area name>
form.type        : <text>
result           : <action result>
end testout from duetalgol - entry 8
```

Bit 15

Ensures that the calculated times for each transaction are output:

```
cpu time         : <cpu time used>
proc. time       : <proc. time used>
real time        : <real time used>
resp. time       : <appr. resp time>
```

Bit 16

Controls the test output from Algol operation no. 9 - bit manipulation:

```
start testout from duetalgol - entry 9
all bits cleared
or
all bits set
or
clear bitno.     : <bit number>
or
set bitno.       : <bit number>
returned bitpattern <bitpattern for 24 bits>
end testout from duetalgol - entry 9
```

11. Test output.

Bit 17

Not used

Bit 18

Causes output of the variable value from the log head in text log records:

```
testout from textlog
textarea        : <area name>
datalength      : <length of block>
position        : <segment no.>
```

Bit 19

Not used.

Bit 20

This produces one test output per operation on the reply area of the system terminal:

```
opcom opened
opcom closed
```

Bit 21

Not used

Bit 22

On assigns of the testvariable - including testg - the bit pattern assumed by the test variable after the assign is printed:

11. Test output.

test a <bit pattern for 24 bits>
test b <- - - - - - - - - - - - ->

test h <- - - - - - - - - - - - ->

Bit 23

Takes care of all that is undertaken via the system terminal:

listing of the transaction
field contents with 'z' trans.
result of put with 'z' trans.
bit no. for each bit with test variable assign.

12. Algol operations

Sorting

Algol operation no.1 carries out sorting of sequential files, as sorting is not possible in Duet.

The operation requires 5 parameters, namely:

1. : no of records to be sorted - word
2. : max. record length - "
3. : no. of sorting keys - "
4. : a key discriptor (cf. sortprocbs-resl 21-V015)
5. : name array for use by sortprocbs - text array

If the core size permits it internal sorting, using the Algol sorting procedures newsort and outsort is preferred, otherwise sortprocbs is used.

On any error occurring during execution of operation 1, the terminal which has caused activation is blocked and an error message is printed on the operator terminal.

12. Algol operations

Algol operation no. 2

Area creation

Creation of an area on a given disc.
The action requires two parameters when called, namely:

1.: area name - text array
2.: document name (kitname) - text array

An area with <area name> is created on the disc <document name> if possible, otherwise an error message is sent to the operator terminal.

12. Algol operations

Algol operation no. 3

Alphanumeric key

The generation of an alphanumeric key. The operation requires two parameters, namely:

1.: a text string - text array
2.: a return key - long

Using a simple algorithm, the text string is 'crunched' to an integer, which is placed in <return key>.

No error messages or reactions, but crunching stops when a <NULL> character is met - the return key is always of type long.

12. Algol operations

Algol operation no. 4

The operation has 3 functions, namely:

Real time	give date and time
measurement	.. base time for real time measurement
Real time used	.. calculated real time used

3 parameters are required where the first parameter is an integer, which defines the function, the other parameters are longs.

Function 1	Call:	Return:
param.2	irrel	date <dd_mm_yy>
param.3	"	time <hh_mm_ss>

Function 2		
param.2	irrel	cpu time used from jobstart
param.3	"	real base time

Function 3		
param.2	real base time	cpu time used from jobstart
param.3	irrel	real time used

12. Algol operations

Algol operation no. 5

Open_pos_text

Call open_pos_text. The operation has 3 parameters, namely:

1.: channel number - word
2.: area name - text array
3.: area position - word
(seg<12 + halfword)

The parameters are used to open and position a zone - prinz (channel no.) - to the area <area name>,then output can continue on this. Rudimentary parameter type control is carried out in the Algol operation.

Area position: segment shift 12
+ halfword

12. Algol operations

Algol operation no. 6

Close_pos_text

Call close_pos_text. The operation has 3 parameters, namely:

1.: channel number - word
2.: area name - text array
3.: return position - word
(seg<12 + halfword)

The parameters are used as follows:

In the zone prinz (channel no.) a check is made that it is opened to the document <area name>, then the zone is closed - i.e. started transfers are terminated - and the end position is placed in return position.

12. Algol operations

Algol operation nc. 7

Convert_area

Call convert_area. The operation has 4 parameters, namely:

1.:	area name	-	text array 11 char
2.:	printer inf	-	word
3.:	form. inf	-	word
4.:	return value	-	word

<area name> is the name of the area to be converted by DUETCOM. <printer inf.>states how the output is to be executed. <printer inf.>has 3 legal values: 16, 30 or a printer identification corresponding to DUETCOM's idenfification of the same printer - a text string with at most 3 characters -

printer inf = 16 :	write on that printer which is stated in the DUETCOM terminal catalogue, as belonging to the terminal which sent the transaction.
-"- = 30 :	output to the terminal and connected printer, if any, on the same transmission line section.
-"- =<xxx>:	output to the printer which DUETCOM idenfifies as <xxx>

12. Algol operations

form. information gives the type of form. to be used for printing, packed in the following way:

<no. of lines from top of form until VT> shift 12 add
<no. of lines until FF> helt på formuläret

The return value contains the answer from DUET-COM to the print order:

returnvalue = 1: printmessage OK
-"- > 1: meaning various things depending on the type of printer in question. If the printmessage was sent to a selective printer, the return-value > 1 means - printer busy - otherwise the returnvalue indicates some error concerning printer, terminal, terminalcatalog or printercatalog.

12. Algol operations

Algol operation no. 8

Rename_area

Rename_area. Four parameters are required, the last two are return parameters:

1.:	old area name	(call) text array
2.:	paper type	(call) -"-
3.:	disc name	(return) -"-
4.:	result of action	(return) word

The area <old area name>is renamed to <wrkxxxxxx> and a message is sent to the operator to write out <wrkxxxxxx> on a specifically defined paper type, i.e. the text contained by the parameter paper type. The name of the disc holding <old area name> is returned in parameter 3; on return, parameter 4 contains the result of the action in the form

result of lookup <old area name> x 10 +
-"- -"- generate wrk.name x 10 +
-"- -"- rename entry.

If the total result is not equal to 0 (zero), an error message instead of a print out message is written out to the operator.

12. Algol operations

Algol operation no. 9

Is used to correct or remove bits in an integer variable. The action requires 2 parameters, namely:

1.: result_var : integer - call and return value
2.: bit no. : integer giving which bit is to be altered.

Result_var

Result_var is altered according to the value of bit no.:

+ bit no.	⇒	bit no.<bit no.>set to one
- bit no.	⇒	bit no.<bit no.>set to zero
bit no.<-24	⇒	all bits in result_var set to zero
bit no. >+24	⇒	all bits in result_var set to one.

N.B. the bits are numbered from 1 - 24, bit 0 is not allowed.

13. Definition of the communication procedures

13.1 Start_com

Start_com has only one parameter, i.e. the zone containing data intended for a receiver. That is to say that the data required to be transmitted is written to the area to which the zone is attached.

Using information found in the zone description, start_com can decide whether the transfer of data can be executed in one transmission. If so, the relevant part of the zone buffer is transferred to the receiver to which the input channel is connected; by definition the receiver is always an internal process.

On the other hand, if the data fills more than one buffer, the area is transferred with the help of convert_area to the process, which is to deal with it.

After transfer of data, start_com zeroes the data zone position as an indication that the data has been transmitted.

13.2 Convert_area

Parameters: area name, printer, form

area name : the name of the area, the contents of which are to be transferred to the communication process.

printer : gives the output possibilities for the receiver
16 : print
30 : convert or
<printerident>, which is a printer identification of 24 bits identical to that belonging to DUETCOM, namely a two-digit number followed by a 'p'. Used in connection with a so-called 'selective print message'

form : contains packed information about the format of the form. (c.f. Algol operation no. 7)

The parameters are packed into a message buffer in such a way as to be intelligible to the communication process, and are sent off to this, which ensures the writing out of the area contents.

13.3 Open_pos_text

Opens and positions a zone on a text file.

Parameters : zone, zone description, file name, position.

- zone : the zone which is to be opened and positioned.
- zone descrip. : an array containing basic information about the zone.
- file name : the name of the file to which the zone is to be opened.
- position : packed information about the block (segment) and halfword to which the zone is to be positioned. (Segm. < 12 + halfword)

The zone is opened to the file defined by name and positioned according to position. Using the basic information in the zone description, it is decided whether primula or write is to be used for output via the zone. The zone receives the status:
after character output, corresponding to continued output.

13.4 Close_pos_text

Closes a zone and stores the end position.

Parameters: zone, position

zone : the zone to be closed

position : variable in which end position is stored

The procedure ensures that the last block is forced out onto the attached area, and also takes the position from the zone description and stores is as segment no. and halfword no. in the return variable, position. (Segm. < 12 + halfword)

13.5 Wait

'Wait' is not a proper communication procedure but is used in those cases where it is appropriate to wait for an event.

'Wait' has only one parameter, sec, which contains the value giving the number of seconds one whishes to wait, often in connection with reply area reservation.

14. Definition of the security procedures

14.1 Update_status

Security catalogue

Update_status takes care of the concurrent updating of the TELESTATUS security catalogue (cf.section 3.1.2) so that this at all times represents the present state.

Update_status has only one parameter, start, which defines those fields which are to be updated. Update_status is called twice per transaction, first, when it is realized that the next operation will cause alteration of the database, and second, when the transaction is completed.

The first call, which is recognized by update_status by start having the value TRUE, causes the transaction number of the current transaction and a copy of the transaction to be placed in the security catalogue.

The second call causes all the essential information concerning the completed transaction to be stored, namely:

completed transaction no.
-"- terminal no.
-"- record no. (record output to log)
position in log
-"- in reprocessing log

After every updating of TELESTATUS the updated block is output onto the physical file with the help of setposition.

14.2 Writelog

log
log head

Writelog transfers a record to the log. The record consists of a log head followed by the log record proper. Writelog itself calculates length and checksum.

Parameters	:	head, log data, log data length
head	:	array containing the head belonging to the current log data
log data	:	array containing the log record
log data length	:	integer giving the length of the log data

Writelog generates a record containing the head and given log data, calculates a total length and a checksum, then transfers this record to one or several log zones.

14.3 Mt_log_proc

Mt_log_proc is the block procedure for the log zone which writes on magnetic tape.

Parameters : zone, status, bytes; which holds for all types of block procedures, see 'ALGOL6-user's manual' section 6.3.4.

Mt_log_proc supervises the zone mt_log_zone and attempts to remedy two error situations, namely: end_of_tape and hard error.

At end_of_tape an end record containing the name of the text tape is transferred to the exceeded tape, when the procedure asks for the next tape to be mounted from the pool, and output can continue.

On hard error it is investigated wheter the run has the log on magnetic tape, if so, the run is stopped with stderror. If a disc log is also used, the mt_log is disconnected and a message is printed on the operator console stating the changed conditions.

14.4 BS_log_proc

BS_log_proc corresponds to mt_log_proc, here it is not necessary to change magnetic tape.

Bs_log_proc is the block procedure for bs log zone and supervises this in case of hard error. If any occur it is investigated if a mt_log is used in the run. If so the disc log is disconnected and the operator is informed, otherwise the run is terminated with stderror.

14.5 Indudproc

Indud zone

Indudproc is the block procedure for the input zone and its function is to ignore timer error and report reject errors to the operator console; any other type of error causes inudproc to stop the run with stderror.

14.6 Comend

Comend ensures that the completed secondary input area is left in a well defined state.

Comend has no parameters and its function consists of indicating to the system that it is ready to process a new secondary input area.

System cataloque

This is done by removing the name of the just terminated area from TELESTATUS system cataloque, as well as zeroing the position pointers for the area which are stored in the TELESTATUS security cataloque.

In the entry tail of the completed area, a status is inserted which indicates that it has been processed, and the transaction number which is also found in the entry tail is set to zero.

Comstate

An internal variable 'comstate' in TELEOP is also set to zero, this is an indicator as to whether TELEOP should look for a new secondary input area or not.

14.7 Sense

Sense is the procedure which TELEOP uses to send queries to DUETCOM about the transaction queue.

Parameters : process name, queue, mode

process name : array which contains the name of the process to be questioned (DUETCOM)

queue : integer return value, which contains the number of transactions to be processed.

mode : integer which informs the process (DUETCOM) who the transaction receiver is

Sense is used to discover whether there is time to process secondary input using as criterion:
If the on-line transaction queue is empty, then there is time!

15. Index

15. Index

READER'S COMMENTS TELEOP

RCSL No: 21-T002

A/S Regnecentralen maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback - your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability:

_____

_____

_____

Do you find errors in this manual ? If so, specify by page.

_____

_____

How can this manual be improved ?

_____

_____

Other comments ?

_____

_____

_____

_____

==========

Please state your position: _____

Name: _____ Organization: _____

Address: _____ Department: _____

_____ _____

_____ Date: _____

Thank you !

RETURN LETTER - CONTENTS AND LAYOUT

---------- Fold here ----------

==========

Affix
postage
here

A/S REGNECENTRALEN
Marketing Department
Falkoner Allé 1
2000 Copenhagen F
Denmark