

Minutes of the ALEPH DAQ Software meeting
CERN, Geneva, March 5th, 1986

List of Participants

Bari	F.Ruggieri		
Beijing	Z.Qian	T.Wang	
CERN	C.Arnault A.Farilla J.Thomas	T.Charity A.Lacourt W.von Ruden	R.Fantechi R.McClatchey
Ecole polytechnique	J.Bourotte	C.Roy	I.Videau
Glasgow	D.Martin		
Imperial college	M.MacDermott	J.Sedgbeer	
Orsay	A.Ducorps		
Pisa	G.Sanguinetti		
RAL	D.Botterill		
Royal Holloway College	M.Landon		
Saclay	J.F.Renardy		

The meeting was opened by T.Charity with the following agenda:

- 1) News
- 2) Status Reports
 - i) DAQ Frame
 - ii) Slow Control
 - iii) Fastbus Database Project
- 3) Evaluation of CMS
- 4) AOB

1) NEWS

17 DMT2200 terminals have been bought and people should get to like them (or at least try!)

CERN software is to be copyrighted . See transparency for the text to be put at the top of each program. Question is really what software is deemed to be ALEPH's and what is deemed to be CERN's and does ALEPH software have to be copyrighted?

There is a new simplified login procedure which is now much quicker (thanks to C.Arnault).

New upgrades to Macintoshes are available (see Saldana). principally upgrades from 512k to 1M and keypad included into keyboard.

Vaxset agreement: package of LSE, CMS, MMS, PCA and TMS available for 20k5F. Sitewide licence being negotiated by DD. Europewide licence being considered but DEC European structure interferes.

EMU error message facility status report was circulated, several problems were noted with EMU:

- i) not distributed around cluster
- ii) routing problems

UPI version 1.0 :- menu package released

2) STATUS REPORTS

DAQ FRAME :- J.Bourotte (see transparencies as well)

UPI, Switcher and Scheduler are now running. Templates for producer tasks (e.g. camac) and consumer tasks (e.g. monitoring) are available.

Tape handling was discussed at some considerable length. Tape

headers written on the VAX in ASCII should, in principle, be readable on VM but there was some doubt about what happened in practice. EPIO routines take 10µs to convert VAX 32 bit word to IBM 32 bit word. This is a rather large overhead which should be avoided if possible. Possible solution is to do bit swapping via hardware on VAX and the 1/2 word swapping on the IBM, however this leaves one with a tape which (without conversion) is unreadable on both machines. Alternative solution is the translate command on IBM although it is not known how long this will take. If timing of translate command is reasonable then tapes will be left in VAX format. Final problem is that of BOS bank names which could be in ASCII or EBCDIC. It was decided to use ASCII globally.

SLOW CONTROL :- T.Charity (see transparencies)

Microprocessor and VAX software is under development following the software design and communications between VXALTP and G64/6809 have been established.

FASTBUS DATABASE PROJECT :- R.McClatchey (see transparencies)

The transparencies are self explanatory!

3) CMS EVALUATION :- M.MacDermott (see report enclosed)

CMS was described and various advantages shown. The main advantage was its simplicity of use and the main disadvantage (which may have been overcome; see above in news) was its price. Discussion revolved around whether its advantages outweighed the cost. Finally it was decided that if it was free (i.e. ALEPH wasn't asked to pay for it) then it would be of considerable benefit, but that if it was going to cost ALEPH money then the decision would depend on whether there was money available which could be best spent on this product.

Agenda


- **News**

- **Status Reports:**
 - DAQ Frame JB
 - Slow Control TC
 - Fastbus projects RMcC

- **Evaluation of CMS** MM

- **A.O.B.**

News from Cern

- Terminals
- Copyrighting of CERNTM software
- Simplified login procedure
-  MacIntosh upgrades
- VAXSET agreement
- EMU Message Utility
- UPI release 1.0

Copyright CERN, Geneva 1986 - Copyrights and any other appropriate legal protection in these computer programs and associated documentation reserved in all countries of the world.

These programs or documentation may not be reproduced by any method without prior consent of the Director-General of CERN.

Permission for the usage of any programs described herein is granted apriori to those scientific institutes associated with the CERN experimental program or with whom CERN has concluded a scientific collaboration agreement.

Requests for information should be addressed to the Program Library, DD Division, CERN, 1211 Geneva, Switzerland.

DAQ SOFTWARE progress Report

Frame Work

BUFFER MANAGER (Responsible for space gestion and event distribution between producers and consumers tasks)

Last version has been installed on

VXALFB } CERN
VXALBH } RUTHERFORD
VX-----



NEED USER INTERFACE



DEVELOPMENT OF

**UPI
SWITCHER
SCHEDULER**

AND

PRODUCERS

CONSUMERS Templates

UPI

new menu package using SMG (VMS-RTL)
windowing facility - Demo has been done
yesterday in COMPRO meeting -

Status : running
doc done
distributed

SWITCHER

kind of terminal server, using UPI,
(can be viewed like an extension of UPI)
allowing the building of a menu with
sub-menus belonging to several processes

Status : running
doc in progress
to be used in BeamVax for
more weeks before distribution

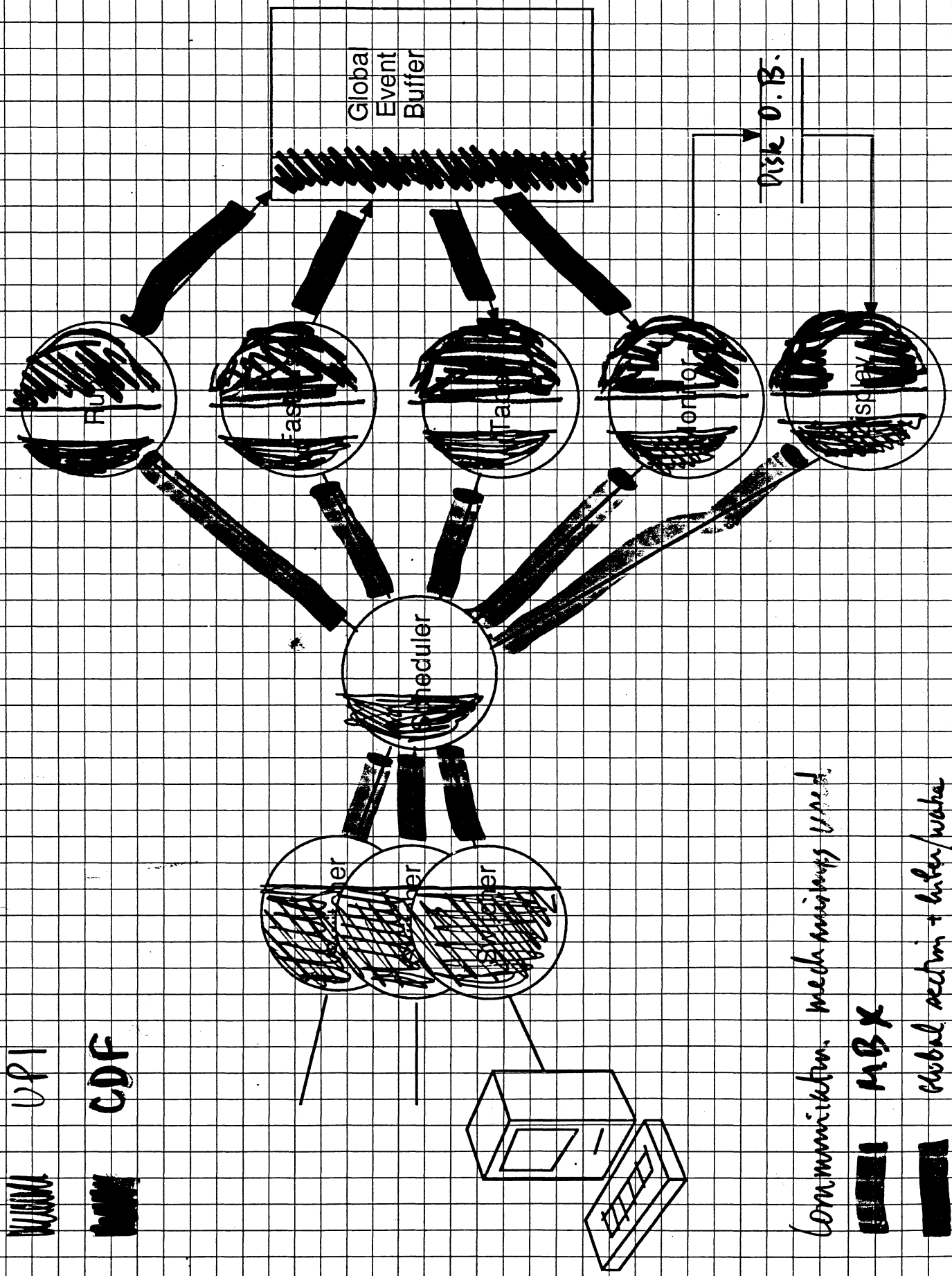
SCHEDULER

central process responsible for
process creation/deletion
control of operator/process communication

Where these packages are used!

UP

GDF



Communication mechanisms used.

MBX

Global section + buffer/packet

SMTP id

PRODUCERS

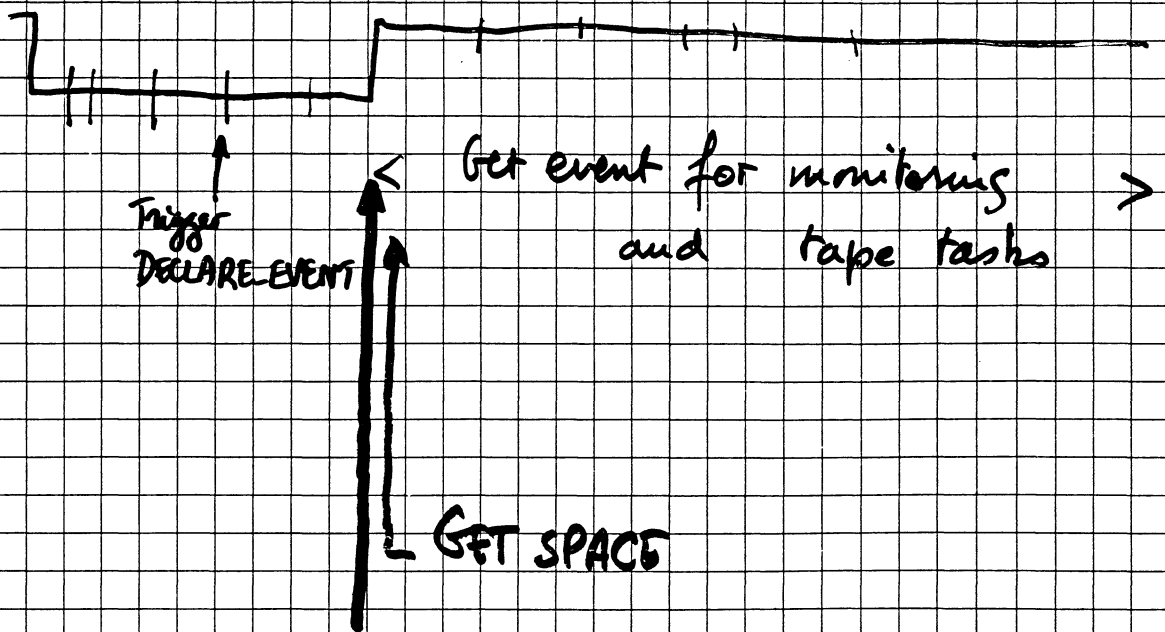
Dummy producer for debugging purposes has been implemented -

CAMAC producer has been written, including:

- * routines allowing to receive triggers from LAM's
- * routines using standard ESONE cells for single action (CSSA)
multiple camac functions in one call (CSGA)
- * DMA Transfer

CDP implementation in burst mode

ORIGINAL
GET-SPACE



E.O.B
SEND-SPACE

CONSUMERS

- Existing templates for monitoring free running and VIP consumers
- Standard USER routines to give event / part of event must be implemented, including
SOR
EOR
SOB
EOB

TAPE

D.R. Botterill is writing the software using
→ CDF standard VIP consumer task
→ VIP menu package

Decided: to use EP10 format with 32 bits physical / logical headers.

Not d. : IBM / VAX compatibility

Tape labels

Data Format

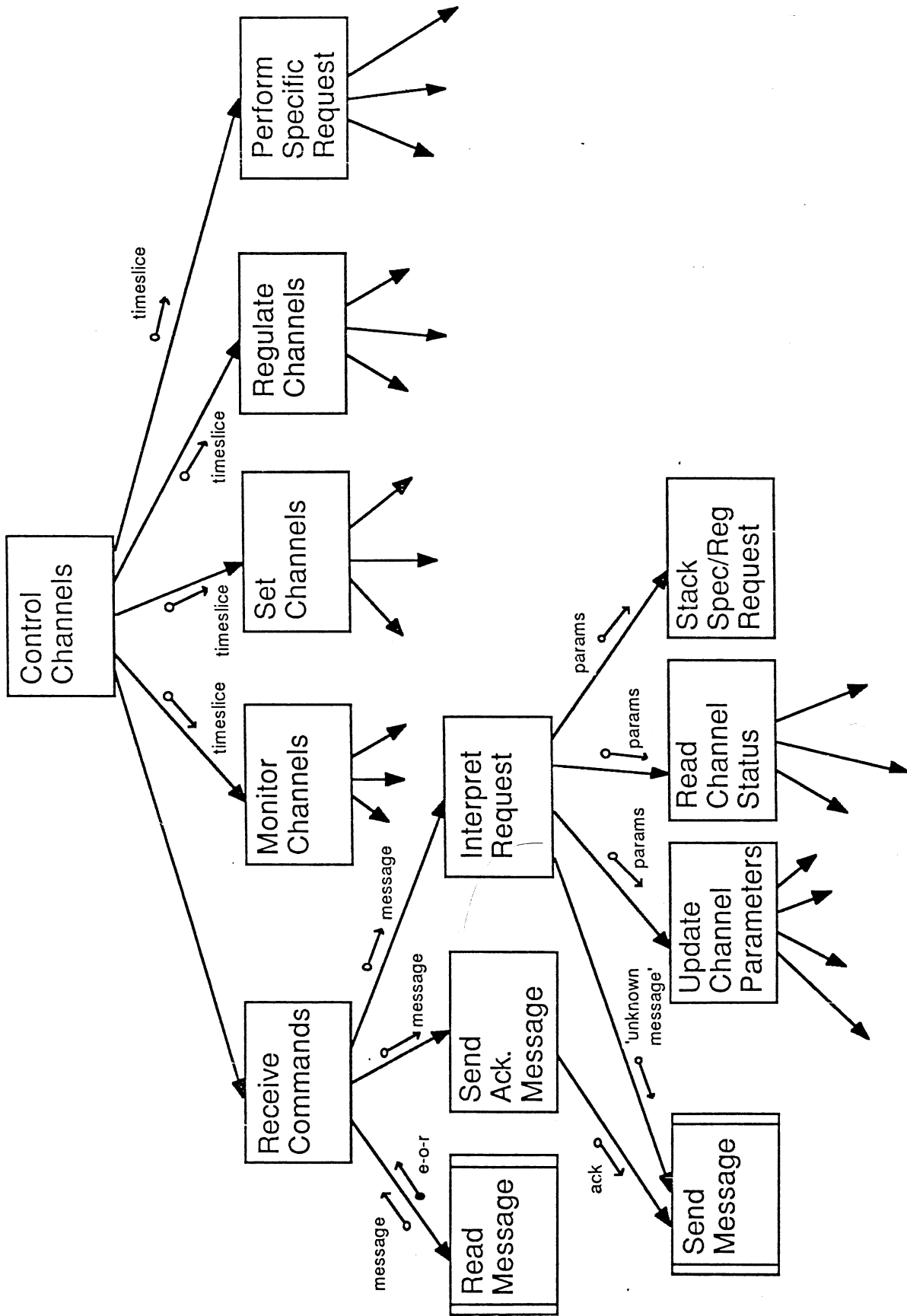
- byte ordering
- ASCII / EBCDIC
- BOS structure

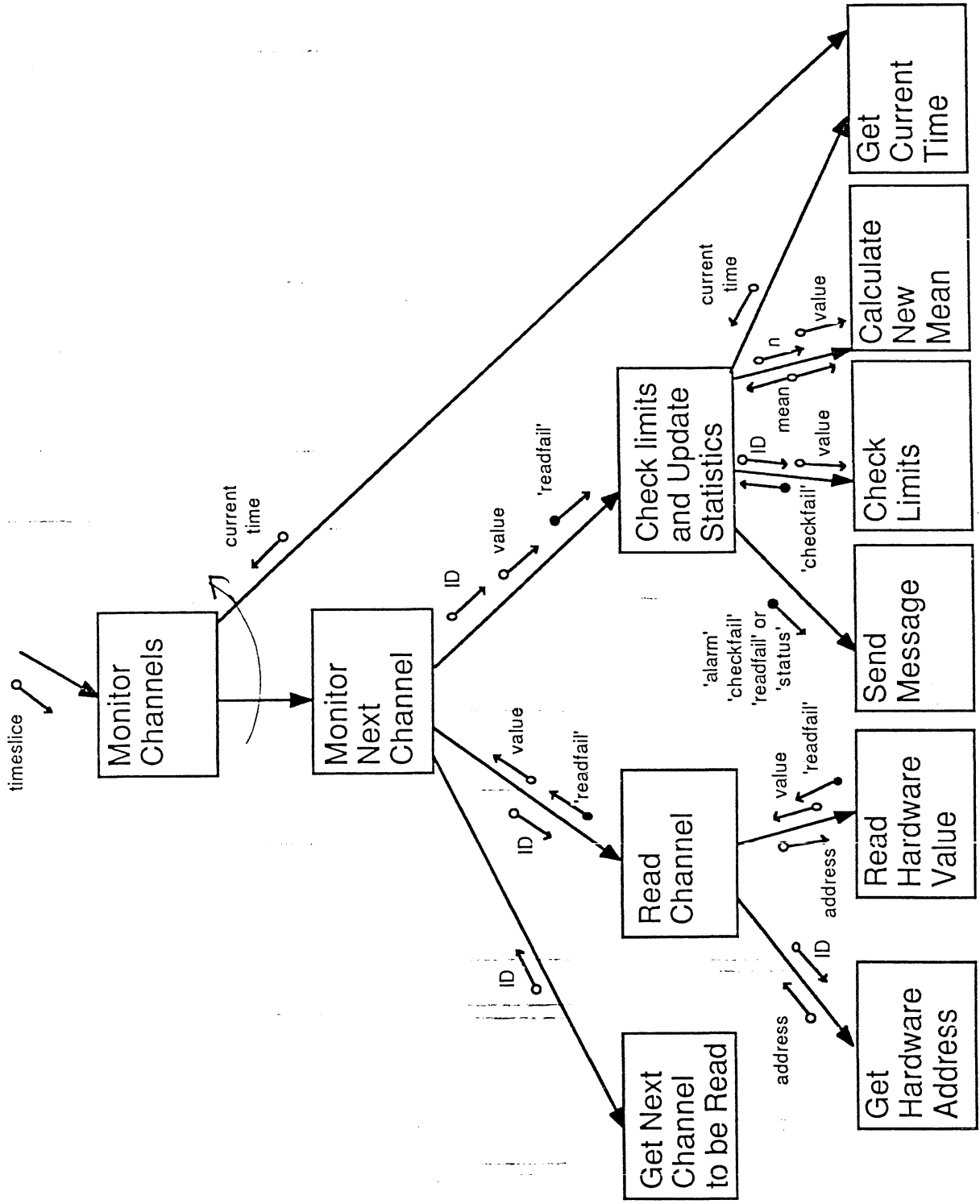
→ a minimal BOS structure including bank header with 4 char BANK name will be provided, allowing monitoring tasks to get part of event -

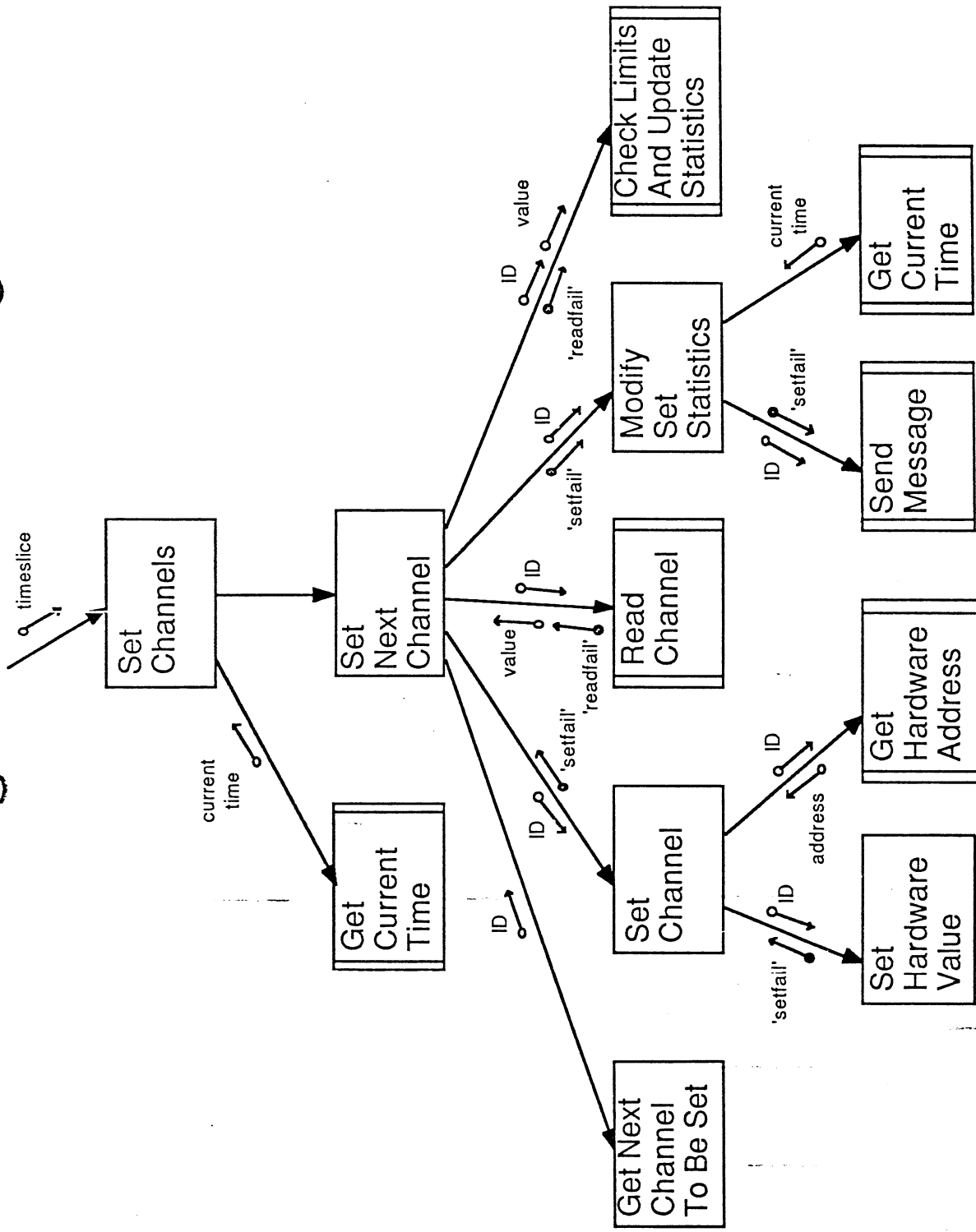
- Event Dump ready

Microprocessor software

- Module implementation and testing on TPC temperature/hv crate, and barrack safety crate.
- Software design is in accordance with structure charts developed from original dfds.
- Performs standalone monitoring and error reporting in accordance with parameters from downloaded datatables.
- Modules to control and readout CAEN-hv card are under development.
- Automatic crate and card identification is defined and implemented. This allows definition of VPA-space on power-up, for safety.







Communication and networking

- Communication between VXALTP and 6809/G64 system has been established via slave CAMAC/UTINET interface.
- Protocol is packet-level, using standard UTINET software without dynamic address allocation. The crate address is used to define the UTINET station address, and to declare it to the VAX.
- Information is exchanged through so-called 'atomic transactions' i.e:
 - **Connect-PutPacket-GetPacket-Disconnect**,
where each message requires a synchronous acknowledgement.
- Internal packet structure designed to be compatible with Marseilles e-gamma gas slow control system of P.Payre.
- Message-level exchange to be implemented using FL protocol on 6809 processor.

VAX software

- Low-level UTINET library written to control CAMAC interface, both in FORTRAN and PASCAL, via a translation of existing NODAL interface routines.
- PACKUTY-style network utility written on VAX to allow dialogue with stations on UTINET segments.
- Initialisation and control program for TPC station written, to allow continuous temperature monitoring and alarm generation. To be included as a start-of-run task during tests.
- Slow Control system is being modelled using the entity-relationship approach. Model will then be used to generate schemata and DDL for ADAMO, together with Oracle tables, to allow VAX initialisation and control software to be designed. (cf. Fastbus).

FASTBUS DATABASE PROJECT

- **Outline** of project and list of participants.
- Scope of project and **overall software design**.
- The advantages of using a **data modelling** technique to design data storage.
- The **ADaMo toolset** and how it interfaces to ORACLE.
- The **Fastbus system database** - its implementation.
- Present **status** and future **plans**.

Fastbus Databases at CERN.

PARTICIPANTS :

- T.Kozlowski (OC/RA)**
- T.Kokkinias (DELPHI)**
- J-P.Dufey (UA2)**
- C.Story (DELPHI)**
- R.McClatchey (ALEPH) †**

• **DD/OC propose** four projects :

1. **FASTBUS module description database + user interface** for entering and retrieving data.

2. **FASTBUS module and configuration diagnostics software.**

3. **FASTBUS configuration description database + user interface.**

4. **FASTBUS initialisation software.**

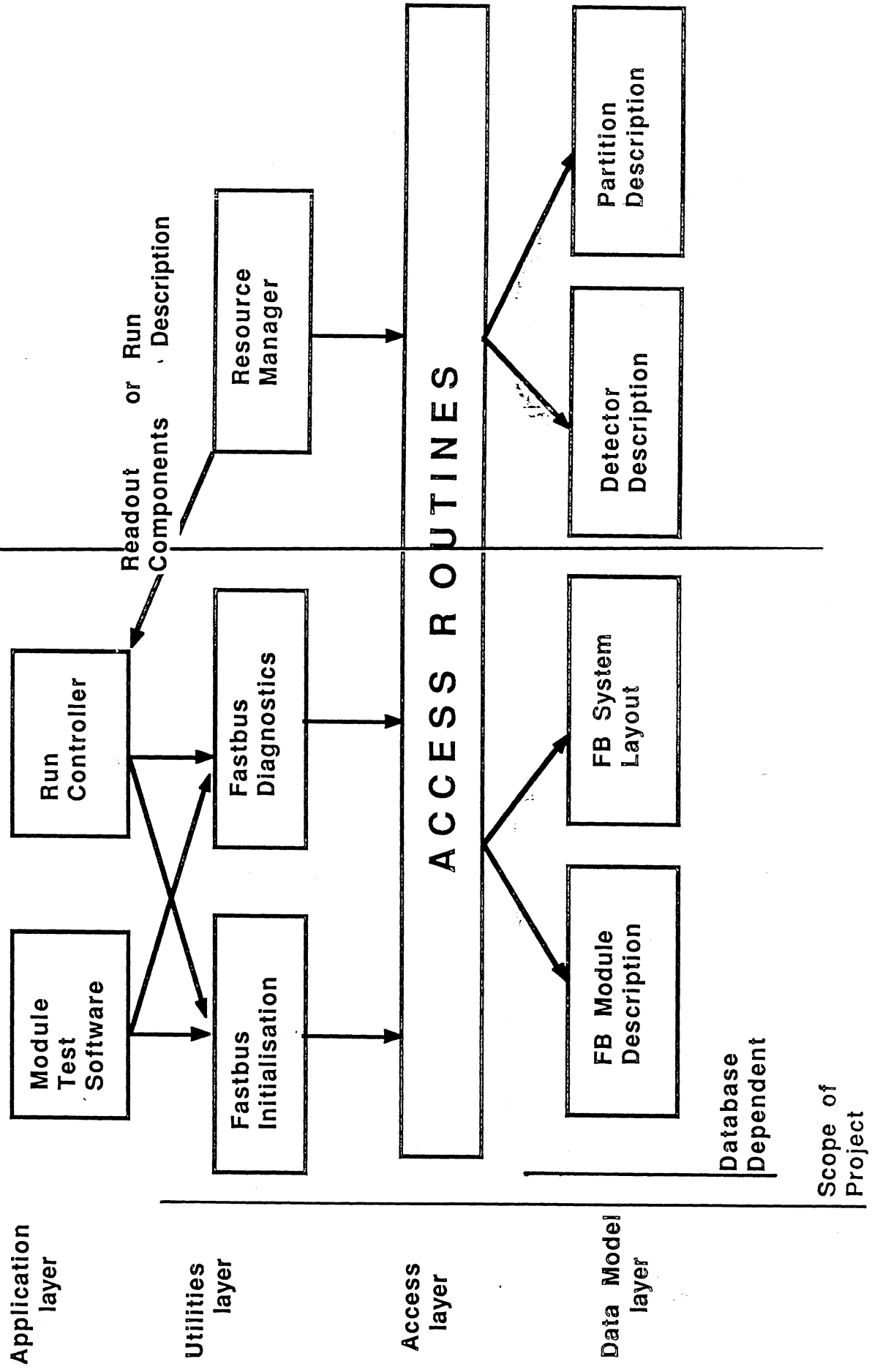
• Projects and data stored in database are **INTERRELATED** therefore must be developed in parallel.

• Proposal is to use the **ORACLE** system to allow :

- Centralised, non-duplicated **data** with protection.
- **Host language** and **interactive access** through a common query language.
- Multi-user access **concurrency control**.
- **Flexibility** and conceptual simplicity.
- Consistent with choice of DBMS implemented on IBM and DEC at CERN.

EXPERIMENT SPECIFIC SOFTWARE

GENERAL FASTBUS SOFTWARE



FASTBUS Database Software

- Software will be layered into :

- 1 User - supplied Application software (eg TPP test Software)
- 2 FASTBUS Software utilities (eg system initialisation & diagnostics)
- 3 Database independent access routines
(eg find description of module 'X'
insert new module position
change routing table N)
- 4 Database specific implementation of System & Module Descriptions.

Project directed at solving 2-4.

- Design of software (2 & 3) is using SA / SD and is in preliminary stages.

Design of Database structure (4) has been completed using **Data Modelling**.

**For a full SA / SD design the data must be fully understood -
therefore data modelling required at the outset.**

- Choose the **Entity-Relationship** Data Modelling technique :

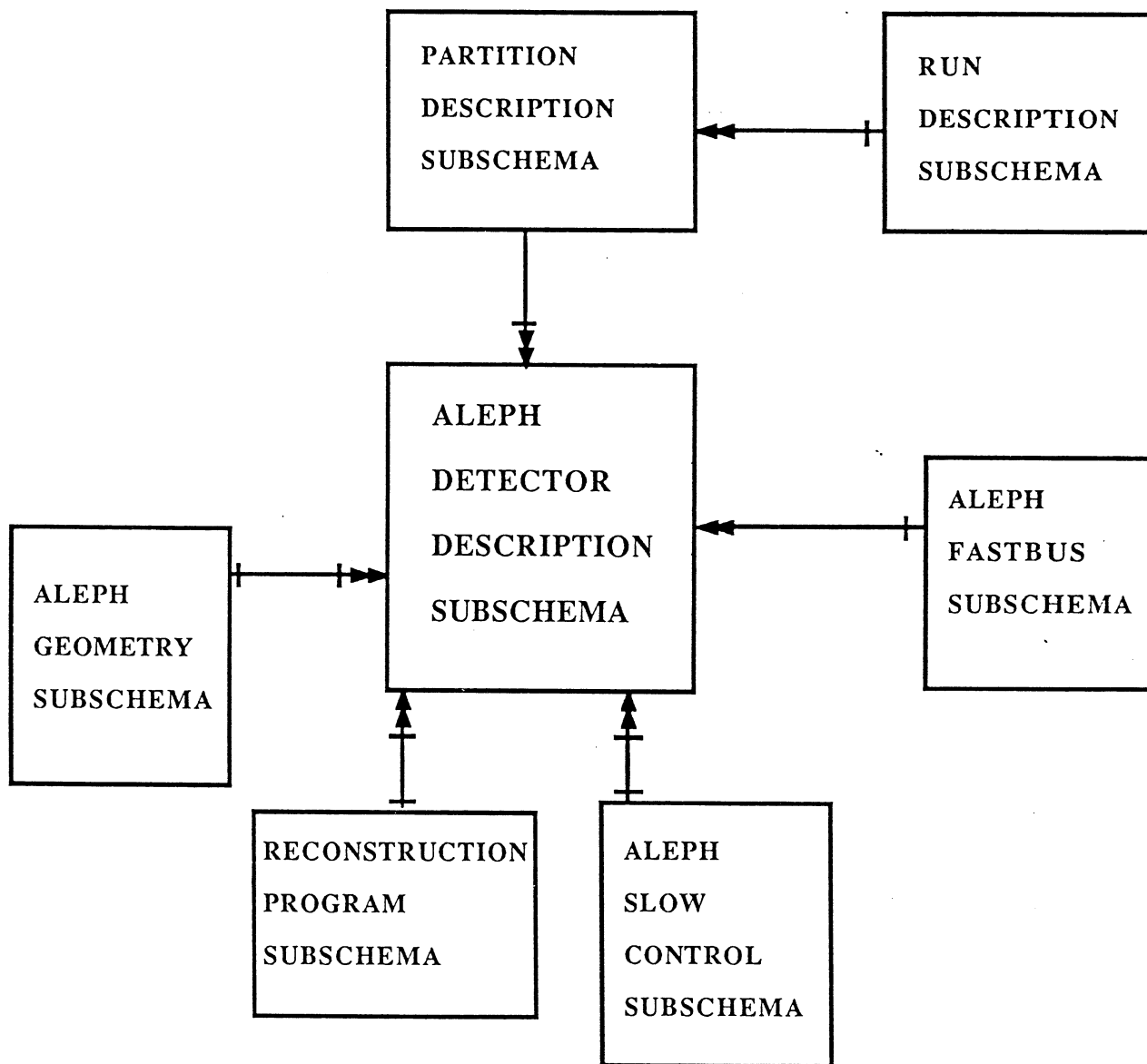
Based on both tables and graphs therefore **good communication tool** and easy to understand.

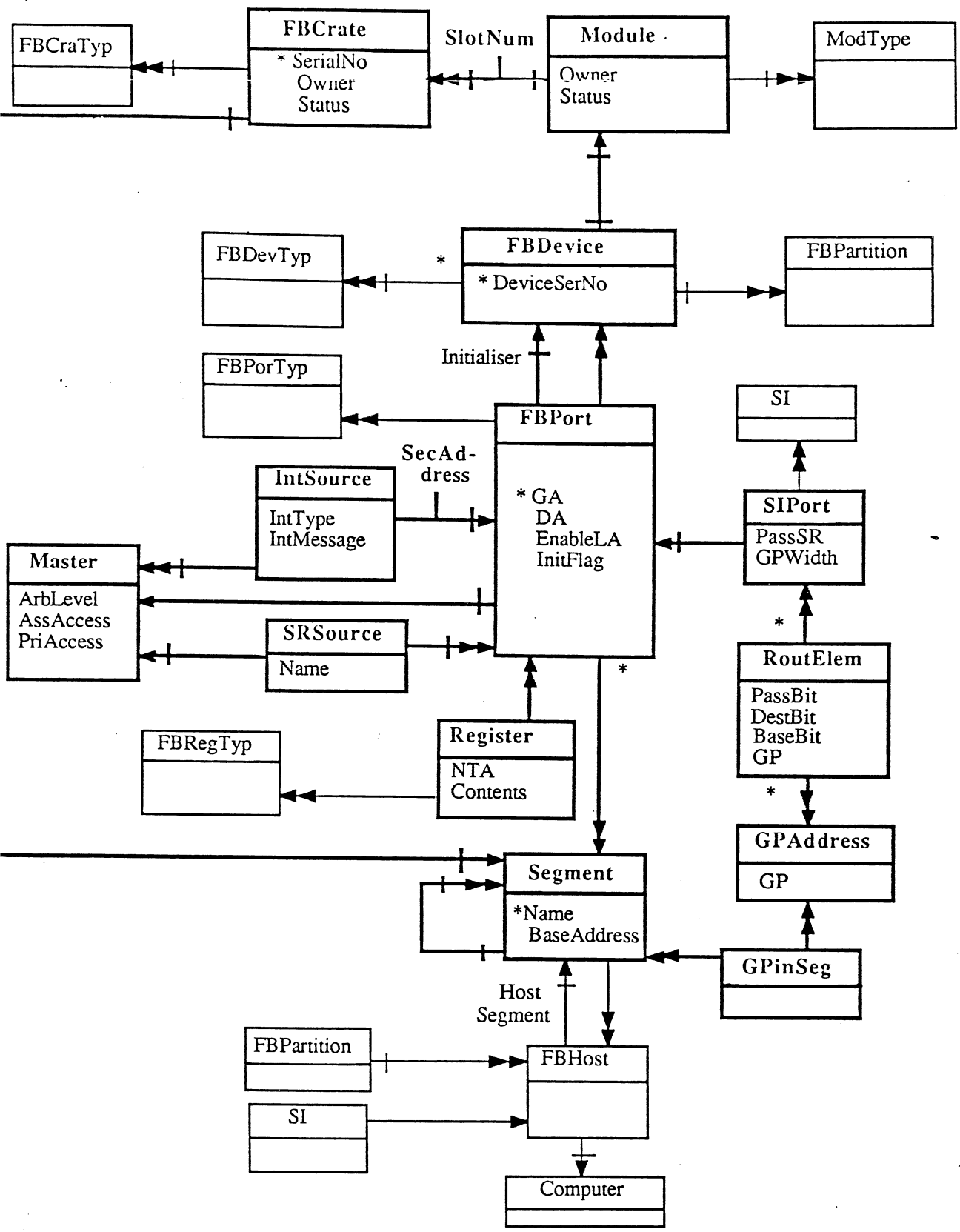
Provides an **interpretation of the data** and its interrelations in a strict formalism.

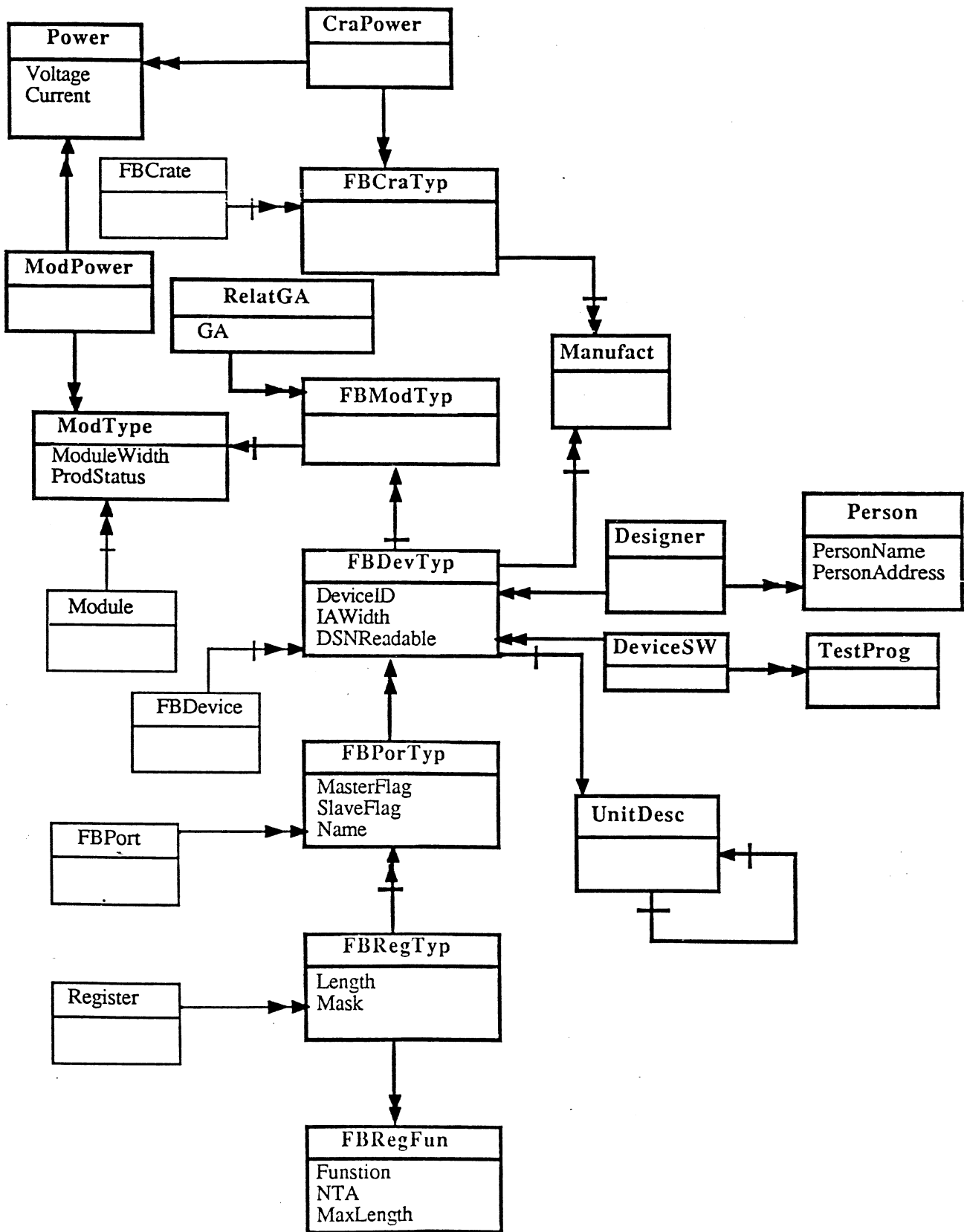
Maps closely to the **relational model** of DBMS and therefore design of the physical database.

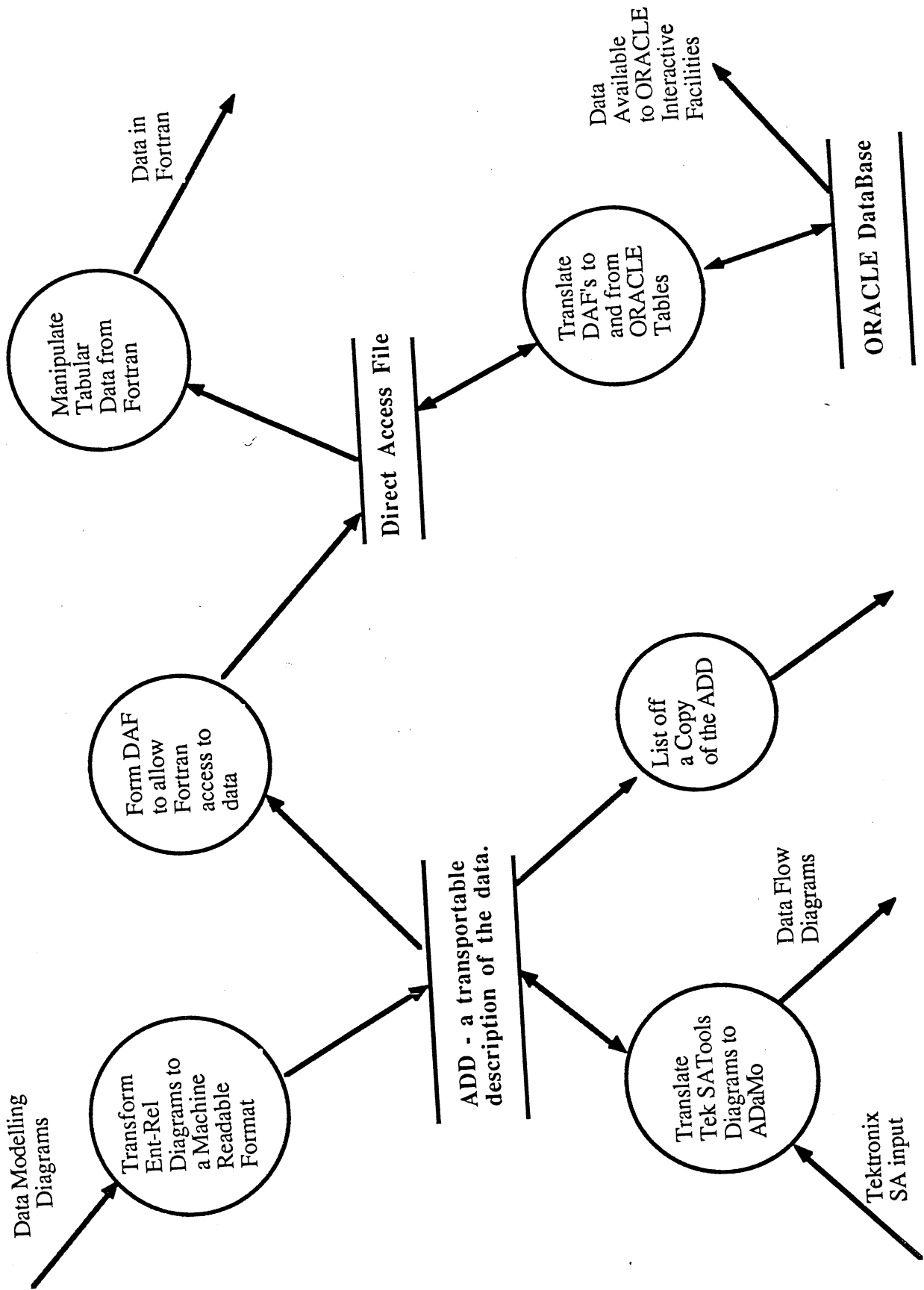
Not restricted to one-to-many relationships as in the hierarchical model.

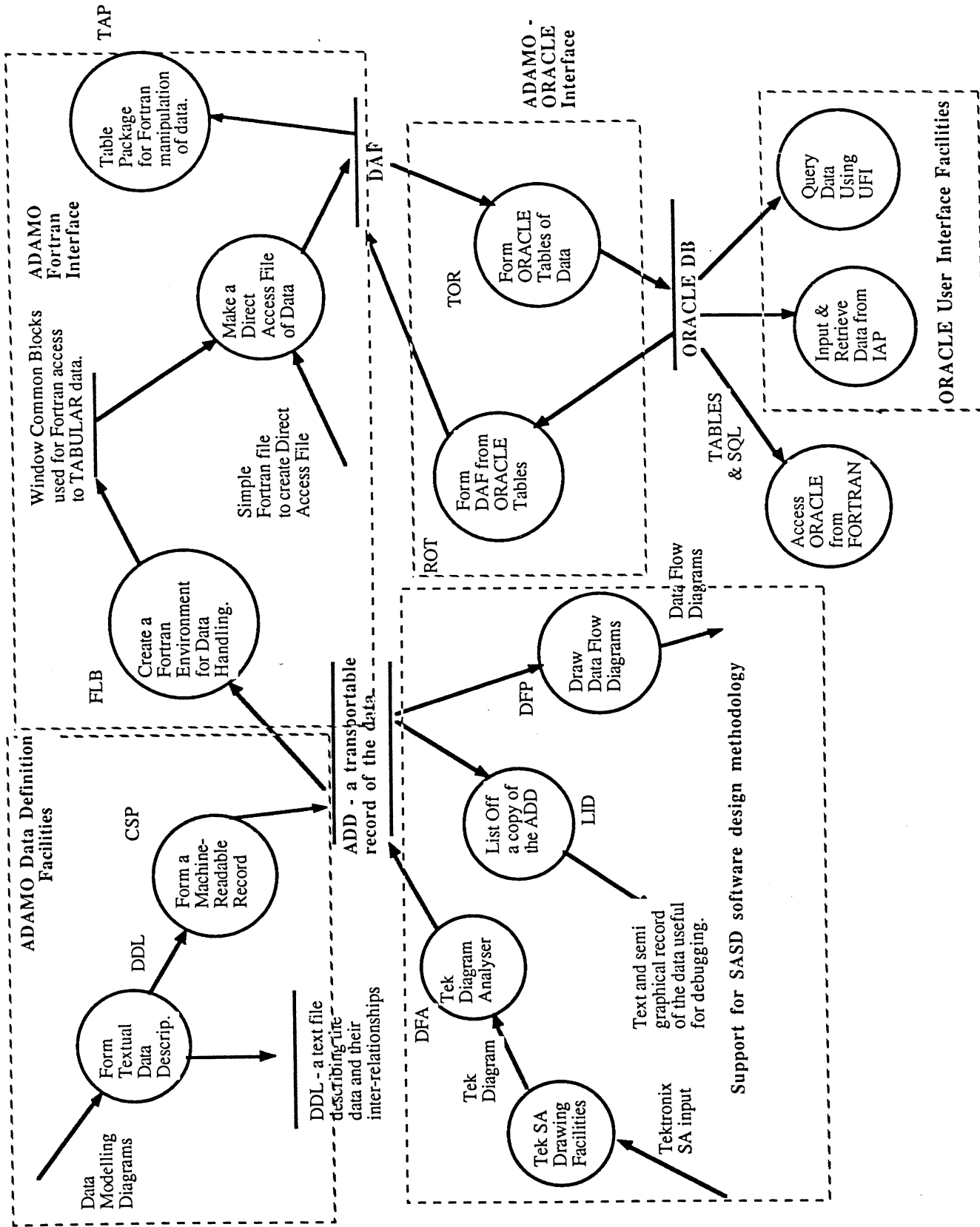
ADaMo tools exist to implement an Entity-Relationship model of the data.











SUBSCHEMA Configuration

: 'FASTBUS configuration description'

AUTHOR 'S. M. Fisher, R. H. McClatchey'
REVIEWER ' , '
VERSION '0.1'

DEFINE ESET

FBPort = (GA,DA,EnableLA,InitFlag)
KEY/Dummy (->Segment,GA)
: 'Geographically addressable part of a FB Device';

Segment = (Name,BaseAddress)
KEY/Dummy (Name)
: 'any Fastbus crate or cable segment';

GPAddr = (GP)
: 'Fastbus group address on a segment';

GPInSeg : 'A particular group address assigned to a particular segment';

RouteElem = (PassBit,DestBit,BaseBit,GP)
KEY/Dummy (->GPAddr,->SIPort)
PARTIAL (GP)
: 'Routing table element for specific SI port and GP address';

SIPort = (PassSR,GPWidth)
: 'Any single Segment Interconnect Port';

Register = (NTA,Contents)
: 'A particular register in a FB port of a FB device';

FBDevice = (DeviceSerNo)
KEY/Dummy (->FBDevTyp,DeviceSerNo)
: 'A specific addressable FB device having an owner and status flag';

Module = (Owner,Status)
: 'FB devices may be modules which can be addressable in a FB crate';

FBcrate = (SerialNo,Owner,Status)
KEY/Dummy (SerialNo)
: 'All FB crates contain FB modules';

IntSource = (IntType,IntMessage)
: 'FB interrupt sources are of types A B or C'

V

PAGE (1, 1) OF (1, 2)

ESet	AtNo	AtNam	ForNam	Type	Cmt(1)	Cmt(2)	Cmt(3)	Cmt(4)
CraPower					Set relating a particular crate	type with a power rating		
Designer					Set relating a particular device	with an individual		
DeviceSW					Set relating a particular device	with an associated program		
FB CraTyp					A generic form of a FB crate			
FB Crate	1	SerialNo		INTE	All FB crates contain FB modules			
	2	Owner		CH16				
	3	Status		CH16				
FB DevTyp					Each device is one of a specific	device type having a device ID		
	1	DeviceID		INTE				
	2	IAWidth		INTE				
	3	DSNReadable		LOGI				
FB Device					A specific addressable FB device	having an owner and status flag		
FB Host					There is one (and only one) FB Host per FB system			
FB ModTyp					A generic form of a FB module (Device in FB crate)			
FB PortTyp					Each FB port can be classed as one of a generic type			
	1	MasterFlag		LOGI				
	2	SlaveFlag		LOGI				
	3	Name		CH16				
FB Port					Geographically addressable part of a FB Device			
	1	GA		INTE				
	2	DA		INTE				
	3	EnableLA		LOGI				
	4	InitFlag		LOGI				
FB RegFun					A collection of specific functions for FB registers			
	1	Function		CH16				
	2	NTA		INTE				
	3	MaxLength		INTE				
FB RegTyp					A generic type for FB registers			
	1	Length		INTE				
	2	Mask		BITP				
GP Addr					Fastbus group address on a segment			
	1	GP		INTE				
GP InSeg					A particular group address assigned to a particular segment			
Int Sourc					FB interrupt sources are of types A B or C			
	1	IntType		CHA4				
	2	IntMessage		BITP				

UFI> select * from tab;

Thu Feb 27

F a s t b u s D e s c r i p t i o n D a t a b a s e

TNAME	TABTYPE	CLUSTERID
MODPOWER	TABLE	
MASTER	TABLE	
MANUFACT	TABLE	
FBREGTYP	TABLE	
FBPORT	TABLE	
FBPORTYP	TABLE	
INTSOURC	TABLE	
MODULE	TABLE	
FBMODTYP	TABLE	
FBHOST	TABLE	
FBDEVICE	TABLE	
FBDEVTYP	TABLE	
MOOTYPE	TABLE	
FBCRATE	TABLE	
PERSON	TABLE	
GPINSEG	TABLE	
FBCRATYP	TABLE	
DEVICESW	TABLE	
DESIGNER	TABLE	
GPADDR	TABLE	
SCHEMA	TABLE	
CRAPOWER	TABLE	
DIRECTORY	TABLE	
POWER	TABLE	

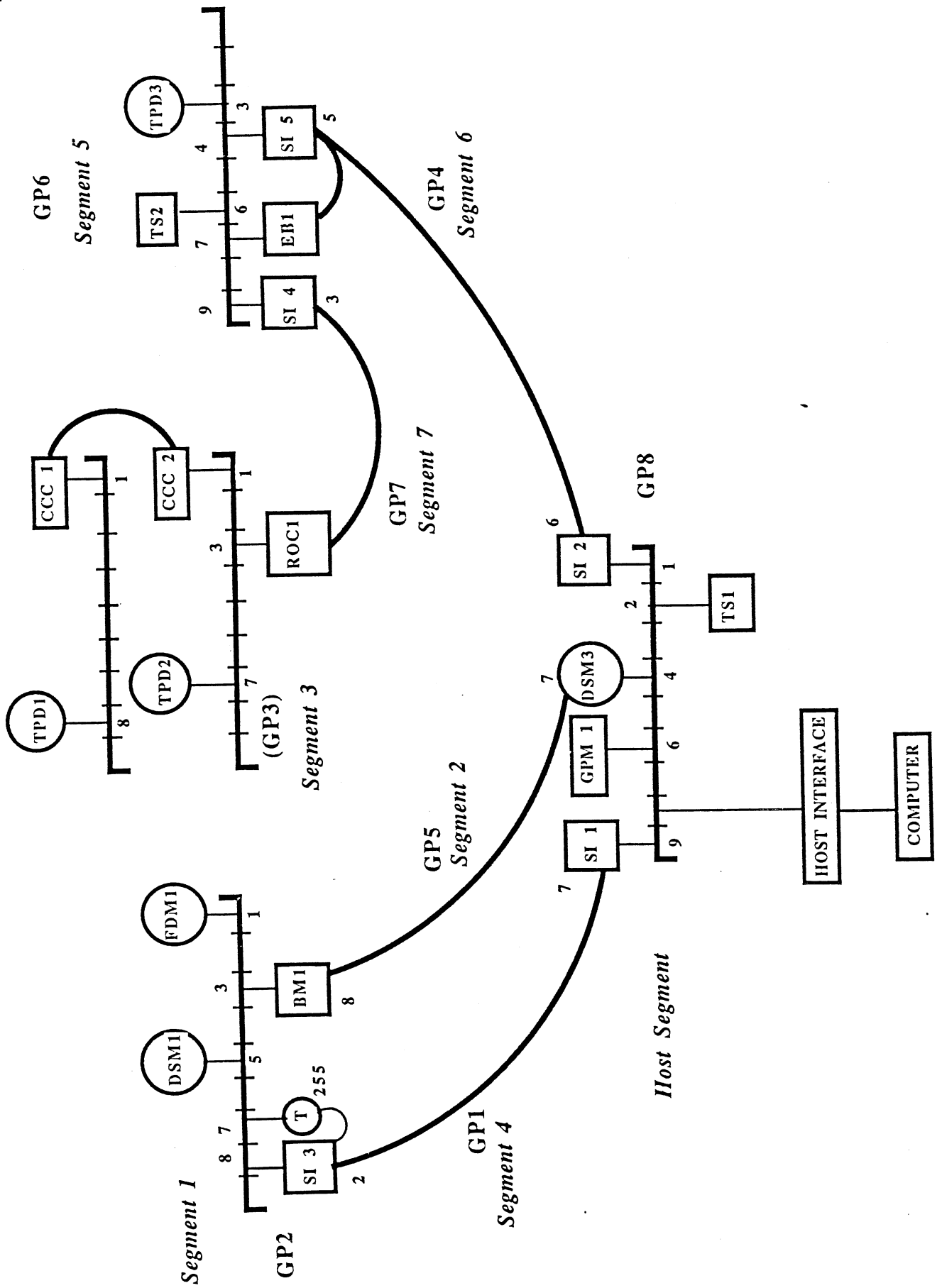
Thu Feb 27

F a s t b u s D e s c r i p t i o n D a t a b a s e

TNAME	TABTYPE	CLUSTERID
REGISTER	TABLE	
RELATGA	TABLE	
ROUTELEM	TABLE	
SIPORT	TABLE	
SRSOURCE	TABLE	
SEGMENT	TABLE	
TESTPROG	TABLE	
UNITDESC	TABLE	

32 records selected.

UFI> desc FBPORT	#	size	cs	type	name
	1	22	40	2 numeric	ID
	2	22	40	2 numeric	GA
	3	22	40	2 numeric	DA
	4	2	1	1 character	ENABLELA
	5	2	1	1 character	INITFLAG
	6	22	40	2 numeric	FBDEVICE
	7	22	40	2 numeric	FBPORTYP
	8	22	40	2 numeric	INITIALISER
	9	22	40	2 numeric	MASTER
	10	22	40	2 numeric	SEGMENT



Present Status and Future Plans

- Simple E.P Fastbus module database and user interface implemented in ORACLE under VM / CMS. Tests being carried out for functionality. Final implementation will be merged with the system description database.
- Fastbus system description database has been modelled (90%) using Entity-Relationship technique and tables exist in ORACLE on VAX 8600. Final model to be established soon.
- SA / SD design of Fastbus initialisation software has begun based on these databases. Should be simple after modelling has been completed.
- Diagnostic software will be provided at the system-wide level only. Users to supply module-specific diagnostics in the first instance.
- ADaMo techniques for producing DAF's and ORACLE tables from DDL to be automated in VMS command files.
- Automatic data-loading methods to be established for filling ORACLE tables.
- Test data to be introduced into ORACLE to rigorously test database. Setup to be of 1986-type eg TPC tests, UA2 and DELPHI test beams (most present devices will be used as input to the database).
- User-interface to be written to access database eg ORACLE UFI and IAP facilities driven through the User Interface package.

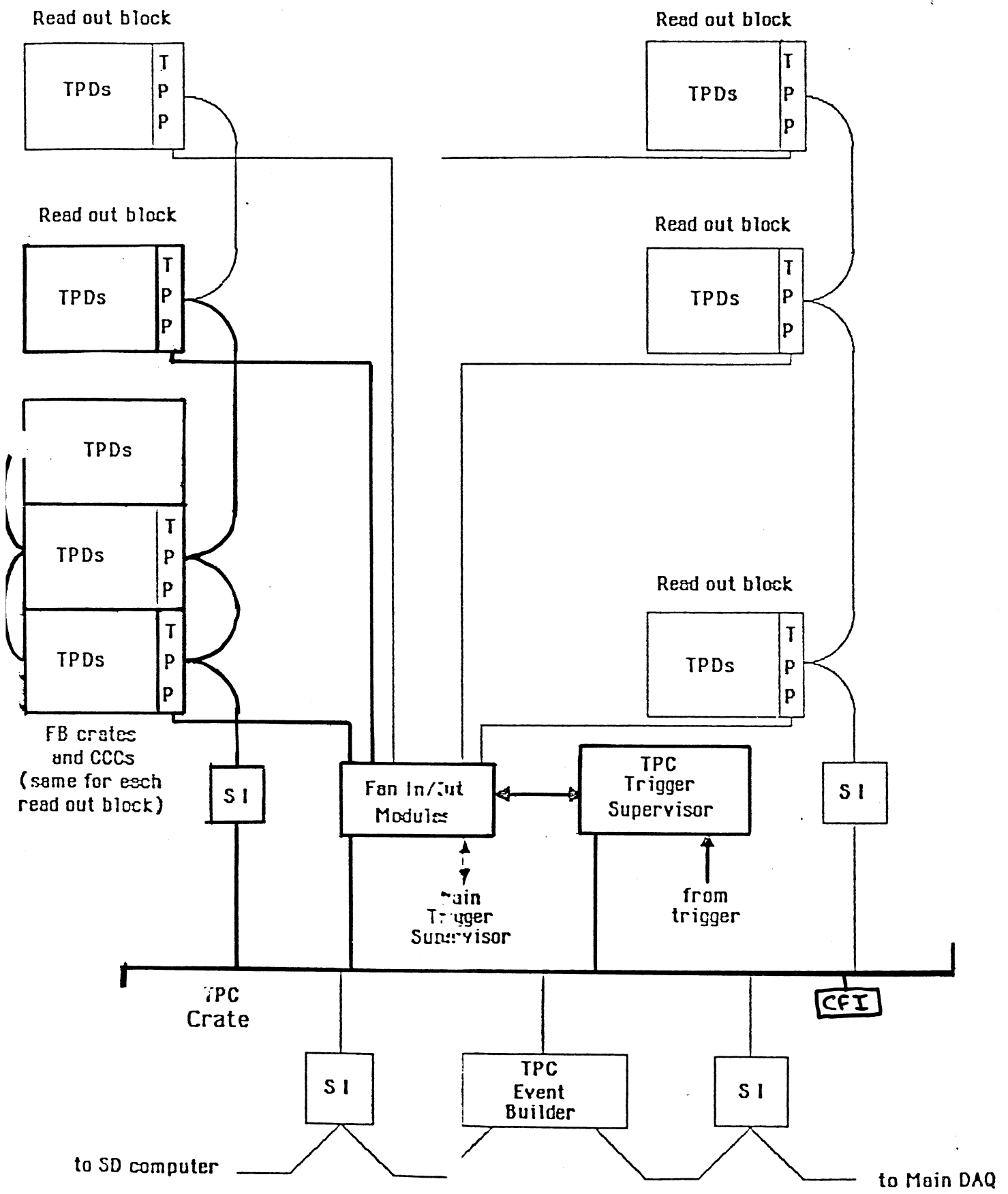
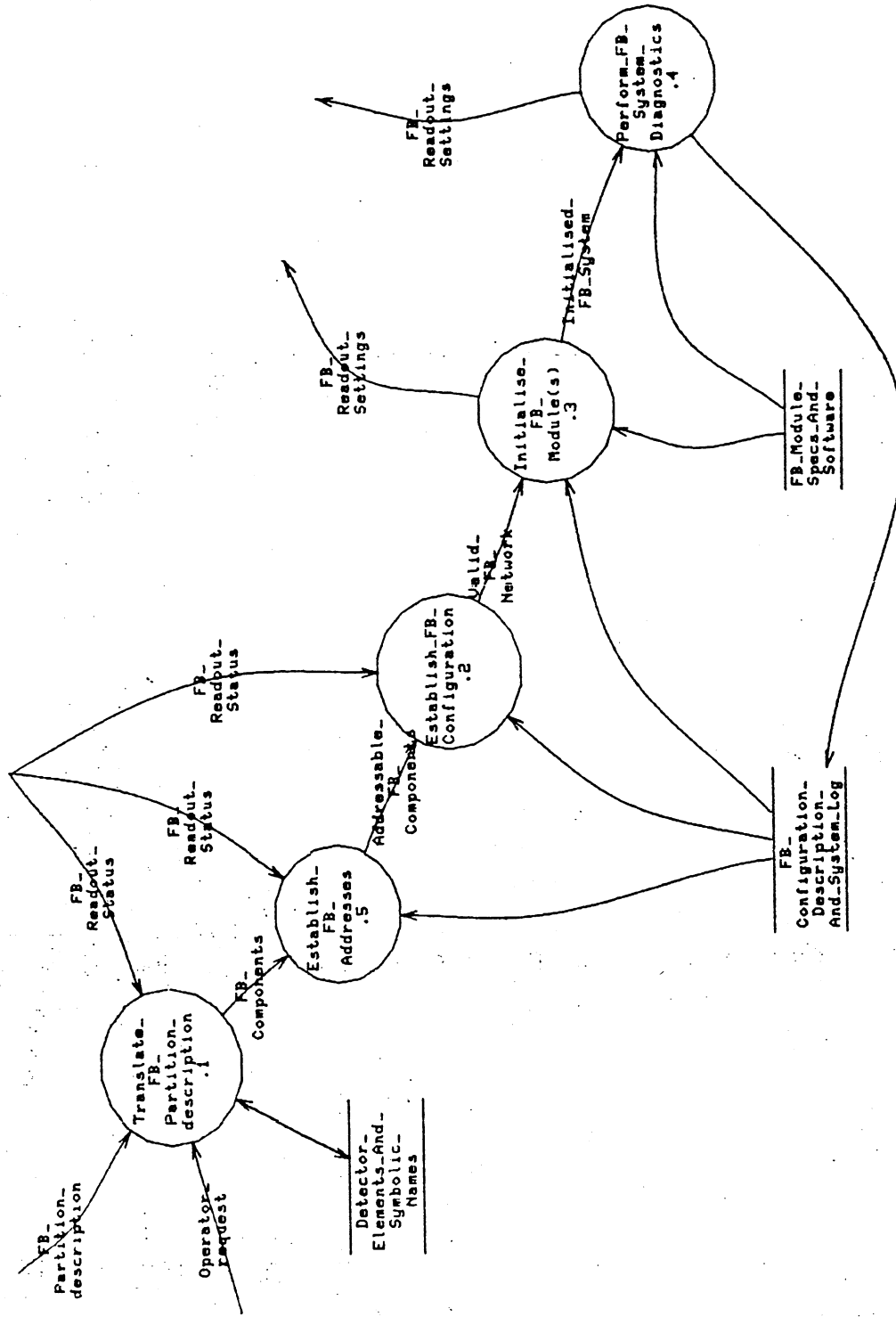


Fig. 2 : TPC read-out scheme



3/ 4/86-knobloch

DF D 1.4.3.1 - INITIALISE FASTBUS SYSTEM

ALEPH Tests of ORACLE.

- Trial period of 6 months. Tests included:
 - Single-user **timing tests** of select, insert and update operations.
Both of **discrete** variables and of arrays of data (in a LONG byte string)
Also stored and retrieved **histograms** from ORACLE.
 - Tests also performed across a LAN (Ethernet).

RESULTS :

- * • Response times for interactive discrete access tolerable.
- * • Arrays must be accessed in **byte strings** to reduce response times.
 - **CPU, I/O demands considerable.**
 - Interactive access (SQL) very **user-friendly** and simple to learn.
 - FORTRAN interface good when used with pre-compilers (+ embedded SQL).
 - **Dynamic SQL** important for FORTRAN applications.
 - Careful database design essential to optimise performance.
 - **Support** from DBMS specialists essential (DD division).
 - General lack of education in HEP in the use of RDBMS's.
- * • Complicated data operations can be **tedious** (eg multiple data entry).
- * • Design of IAF forms can be laborious - requires learning new key-pad set-up.
- * • Access to ORACLE across a network is at present **VERY SLOW** .

----- > **Further Pilot Projects required to test
ORACLE in a Physics environment.**

ORACLE Version 5 Features :

Moving towards a truly distributed architecture .

- **ORACLE NET** Allows transparent, **remote access** to ORACLE on heterogeneous networks eg VAX -- > IBM.
- **VAX CLUSTER ORACLE** Split ORACLE workload across processors in a **distributed file system** environment.
Locking performed with distributed lock manager.
- **IMPROVED PERFORMANCE** : faster sorting, merging functions.
faster processing of data held in buffers.
- **Array handling** : can access multiple rows of data from tables in HLI. (PCC)
- **Improved SQL** operations (SQL-PLUS).
- IAF upgrade using **macro-keys, menuing** (eventually pop-up).
- **Procedural UFI** (PASCAL-like), multiple cursor support.
- **Advanced UFI** including simple **graphics** (data-driven).

The approach of this report has been to look critically at CMS to see if the advantages of CMS sufficiently outweigh the problems of choosing a second code management system (the first being Historian).

CMS DESCRIPTION

CMS provides a simple way of managing code which could be being created and changed by more than one person. All source code is kept in one subdirectory which has been set up as a CMS library. A record is kept of all commands issued on any element within the library and includes a comment inserted by the person issuing the command. The code is split up into elements (normally one element = one subroutine) which will have a varying number of generations associated with them, depending on how many times changes have been made to them. These elements can be gathered into groups to facilitate the command logic. It is also possible to specify classes whereby one can extract easily (for example) the correct generation of each element used to create a previous running version of a given program. A more dangerous facility, if not treated properly, is the idea of a variant line. It is possible to develop an element through a number of generations until a point is reached when two independent development paths are needed, then a variant line can be created. This is not dangerous in itself but there is a facility to merge the two development lines together at any point in the future which is dangerous. All modifications to a CMS library element are done by withdrawing it from the library into a default directory and then either moving it over networks to one's favourite computer or editing it in situ. The choice of editor is irrelevant to CMS as it compares the old and new copies and generates its own edits internally in the file. There is only one file kept with a series of include and delete lines associated with a given generation.

ADVANTAGES OF CMS OVER HISTORIAN

1) CMS can specify, using the class facility, a specific point in the development of the library and hence, very simply, return to an older version of the program or be able to continue development of various elements and still access the source code for the current running version.

2) CMS can specify a group which allows one to access a whole range of elements making it easier to issue commands. This is similar to the Historian command set but, to my mind, easier to use.

3) CMS keeps a record of who did what when and why to a given element. So provided people give a comprehensible comment, a complete history of each element is available.

4) A variant line can be created when the development of an element splits into two (or more) different paths. If two people insist on working on one element at the same time then a variant line is insisted on for the second person. CMS warns you if you want to work on an element which is already reserved or if you are replacing an element which is being worked on by someone else at the same time. It is possible to merge back these two lines together but this is obviously dangerous and should be done with care.

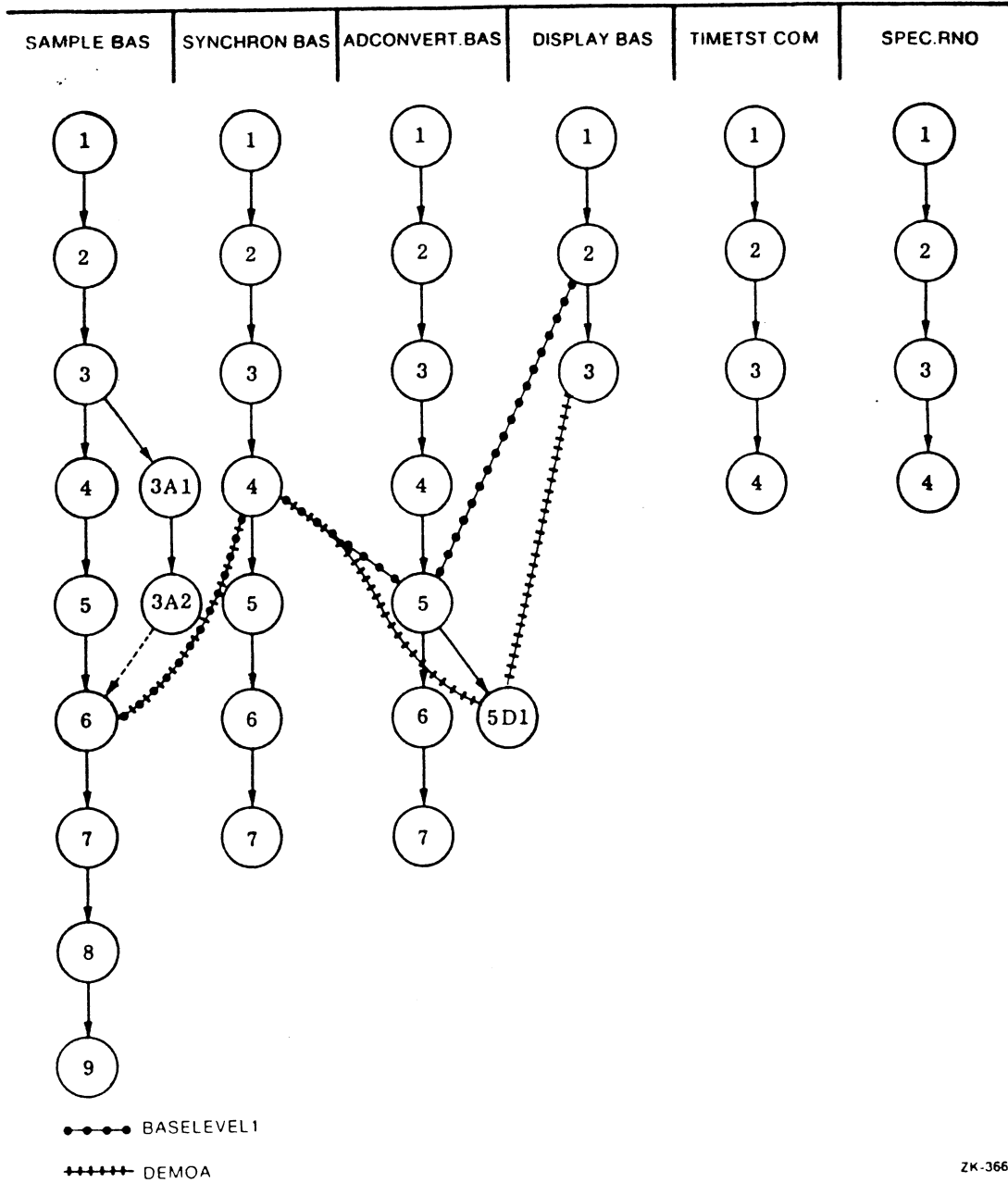
5) The expense of CMS implies that it will be only available on the DAQ computer and that people working on various elements will remove them from the library and then work on them independent of CMS. The transfer into and out of the CMS library is always done in source code and contains no history of work done on the element in another CMS library elsewhere.

The above things can mostly be done by Historian but all of the commands are logically easier with CMS, some very much easier. In fact, CMS is easy enough to learn in one day.

CMS OR HISTORIAN

Obviously we can make Historian do the job, however CMS is a lot simpler and easy to use. The choice of editor/computer for development work on individual elements is almost infinite. So, given the information provided by Alan Silverman that we could well get a site wide license paid for by DD, I recommend that we should use CMS on the DAQ VAX.

CMS LIBRARY



ZK-366

Figure 6-3: Multiple CMS Classes