

# **INVESTIGATION INTO THE REQUIREMENTS FOR AN INTEGRATED COMPUTER-AIDED ENGINEERING ENVIRONMENT.**

---

**Anthony V. Blake**

**March 1991**

**A dissertation Submitted to the Faculty of Engineering, University of the  
Witwatersrand, Johannesburg for the Degree of Master of Science  
(Engineering)**

**Johannesburg 1991**

## **ABSTRACT**

Integration of islands of Computer Assisted Functions is becoming increasingly important in many organisations and is being driven by the need for Data Sharing and the reduction in Duplication of Effort in order to achieve an increasingly competitive edge. Technology is increasingly providing the means of taking the "Integrated Organisation" out of the realms of an "Idealistic Dream World" into the harsh environment of Reality.

The concept of Integrated Computer Aided Engineering (CAE) is discussed, and then some Critical success factors to achieve some degree of success are discussed:

- Non-technical issues, such as Management Involvement and the need for change in Organisational Culture, are placed in perspective.
- Possible "Models" and functions acting on these Models are explored in some detail and
- Real-world issues influencing the development of large integrated software projects are examined.

The contents of this report are based firstly on experience gained in developing and implementing the first phase of a large CAE environment for the Power Station Electrical Engineering Department of ESKOM (namely CEEDS<sup>1</sup>); and secondly on further research undertaken for the continued evolution of the CEEDS system into a truly "Integrated Computer Aided Engineering Environment".

A pragmatic approach has been adopted and some comments and observations may seem unduly cynical. This is because the "Real World" shows no mercy towards the intrusion of "Innocent bright ideas, hoards of momentumless talk and floundering amateurs". This is demonstrated by a failure rate of at least 75% in attempts to introduce CAE into organisations and an even higher failure rate for technologies such as expert systems [MGT 01].

It has been attempted to make the study of general applicability. Illustrative examples and arguments are based on a large organisation's environment and the CEEDS project.

---

(1) CEEDS is an acronym for: Centralised Electrical Engineering Database Scheme.

**Declaration:**

I declare that this dissertation is my own, unaided work. It is being submitted for the degree of Master of Science (Engineering) at the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other University.

A Blake

Dated this 8th day of March, 1991.

*To my wife, Irma*



## Acknowledgments

Thanks to all the team members of the Design Process Automation and Information Technology teams of ESKOM who helped in the development of CEEDS. They include Johan Pienaar, Andre Kotze, Lynn Clarke and Johan Erasmus from the DPA team and Ronel Piek and Helena Pienaar from IT. Thanks also to the managers of the Power Station Electrical Engineering Department with special thanks to Dawid van Rensburg for allowing the use of work time to complete this document. Thanks to all those who did proof reading, with special thanks going to Rez Bieloch, without whose spelling and grammar corrections, the document would have been of somewhat lesser quality. Thanks also to Terence Hertz for his patient assistance in laying out the document in Ventura.

Final special thanks going to my wife for her patience during these trying times.

## Conventions

- Indications of a reference to a paper or book, as listed in the chapter on "References and further reading" (R), are indicated in bold within square brackets "[ ]". For example **[MGT01]** is a reference to the paper listed as MGT01.
- References to other chapters or section within the main report are indicated by giving the chapter or section heading within quotes, in bold print and followed by the chapter/section number in parenthesis. For example, "Job design" (3.5.1), refers to section 3.5.1 in chapter 3.
- Appendixes are numbered as follows:
  - The first letter is an "A" to indicate an appendix;
  - The second number refers to the chapter in the main report that has the most bearing on the contents of the appendix;
  - Thereafter appendixes are numbered sequentially.

For example, Appendix A4.3 is the third appendix bearing relevance to chapter 4.

- Figures are numbered similar to appendixes, only that the number of the main section within a chapter is used and not only the chapter number. For example, Figure F4.3.02 refers to a drawing used in section 4.3. The figure number is prefixed with an "F" for figures used in the main report and with "FA" for figures used in an appendix.

## Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Integrated Computer Aided Engineering Systems</b>	<b>2</b>
2.1	General Discussion on CAE	2
2.1.1	CAE at a corporate level	2
2.1.2	CAE at an engineering discipline level	6
2.2	Brief History of CAE and CEEDS in ESKOM	9
2.2.1	Setting the scene for CEEDS	9
2.2.2	CEEDS I	10
2.2.3	CEEDS II	10
2.3	Critical Success Factors	13
2.4	High Level Development Strategy	13
<b>3</b>	<b>Non-Technical Aspects</b>	<b>17</b>
3.1	Introduction	17
3.2	Organisational Aspects	18
3.2.1	Organisational Culture	18
3.2.2	Right and wrong reasons for the introduction of CAE	18
3.2.3	Productivity and quality	19
3.2.4	Differences in the nature of work which affect implementation success.	20
3.2.5	Change is continuous	21
3.3	Management role	21
3.3.1	Commitment to change from management.	21
3.3.2	Resource planning	22
3.3.3	Non-traditional Design	22
3.3.4	The importance of system ownership	22
3.4	Design Philosophies, Procedures, Operations and Administration	23
3.4.1	Design Philosophies and practices	23
3.4.2	Procedures and Administration	23
3.5	User Aspects	24
3.5.1	Job design	24
3.5.2	Skill factors	25
3.5.3	Differences between users and non-users	25
3.5.4	Age factors	26
3.5.5	Job security	26
3.5.6	Ergonomic factors	26
3.6	Training	27

3.6.1	Skills shortages	27
3.6.2	Specific skills needed	28
3.6.3	Training methods	28
3.6.4	Continuation of Training	29
3.7	Considerations for System Development	29
<b>4</b>	<b>System Models</b>	<b>31</b>
4.1	Introduction	31
4.2	Physical Model	32
4.2.1	Graph based Node-Link Model	32
4.2.2	Systems, Identifiers, Equipment Classes and Types	35
4.2.2.1	Type Reference Tables	38
4.2.2.2	Systems	39
4.2.3	Representational Schemes	40
4.2.3.1	Conceptual Representation	40
4.2.3.2	Physical Database Structures	44
4.2.4	Detail of attributes of various nodes and links	48
4.2.4.1	Nodes	48
4.2.4.2	Links	49
4.2.4.3	Other Attribute Considerations	50
4.2.5	Special Nodes and Links	50
4.2.6	Subtypes of power consuming nodes	50
4.2.7	Extensions to the model	51
4.2.7.1	Cable Racking	52
4.2.7.2	Cable drums	53
4.2.8	The Deletion of Objects	54
4.2.9	Signal Model	54
4.3	Design Process Model	55
4.3.1	General discussion	55
4.3.2	Project Planning	56
4.3.3	Design Sequence	56
4.3.3.1	Representation of the Design Process Model	59
4.3.3.2	Design Automation	62
4.3.3.3	Some points to note	63
4.4	Financial Models	64
4.4.1	Costs directly coupled to the Physical model	64
4.4.1.1	Capital Expenditure costs	65
4.4.1.2	Re-Engineering Costs	66
4.4.1.3	Limitations on Physical Model based Financial Facilities	66
4.4.2	Other Facilities	66

<b>5</b>	<b>System Functionality</b>	<b>67</b>
5.1	Introduction	67
5.2	The Man Machine Interface (MMI)	67
5.2.1	Menu facilities	67
5.2.2	Data manipulation facilities	68
5.2.3	Reporting Facilities	69
5.2.4	Query facilities	70
5.2.5	Help facilities	70
5.2.6	Proposed Future MMI	71
5.2.7	Other factors	71
5.3	Design Functions	71
5.3.1	Creation, modification and deletion of Model Objects	71
5.3.1.1	Nodes	72
5.3.1.2	Links	72
5.3.1.3	Importing and Exporting of Model Data	74
5.3.1.4	Graphical Query capabilities	75
5.3.2	Design Verification Checks and Model analysis	76
5.3.2.1	Verification Checks: (Data integrity checks)	76
5.3.2.2	Analysis	77
5.3.3	Codification	78
5.3.3.1	The codification of Object ID's	79
5.3.3.2	Other codification	80
5.3.4	Artificial Intelligence (AI) Technology	80
5.4	Construction and Financial Functions	81
5.5	Documentation Production facilities	82
5.5.1	General reporting Facilities	82
5.5.2	Standard reports	83
5.5.3	Contractual documents	83
5.5.3.1	Modelling a document as an object	85
5.5.3.2	Configuration management philosophy	85
5.5.4	Report Design	86
5.5.5	Textual and graphical representations	88
5.6	Internal Functions and Utilities	88

5.6.1	System Security . . . . .	88
5.6.1.1	Access to computing facilities and applications . . . . .	89
5.6.1.2	Security within an application . . . . .	90
5.6.1.3	Database security . . . . .	90
5.6.2	Logging Facilities . . . . .	91
5.6.2.1	External Data Import/Export logging . . . . .	91
5.6.2.2	User activity logging . . . . .	91
5.6.2.3	Report and Contractual Document Print logging . . . . .	92
5.6.2.4	Administration of log data . . . . .	92
5.6.3	History Facilities . . . . .	93
5.6.3.1	Facilities Required . . . . .	93
5.6.3.2	What data to capture . . . . .	94
5.6.4	Utilities . . . . .	94
5.6.4.1	Flexible Query Facilities . . . . .	94
5.6.4.2	Queries about the system itself . . . . .	95
5.6.4.3	Communication facilities . . . . .	95
5.6.4.4	Others facilities . . . . .	95
5.6.5	Mass update facilities . . . . .	96
6	Software and Development / Implementation Aspects . . . . .	97
6.1	Introduction . . . . .	97
6.2	Computer Environment . . . . .	97
6.2.1	Current platform . . . . .	97
6.2.2	Proposed future platform . . . . .	100
6.3	Development resources . . . . .	101
6.3.1	People . . . . .	101
6.3.2	Time . . . . .	102
6.4	Building and Maintaining the System . . . . .	103
6.4.1	Design methodologies . . . . .	103
6.4.2	Coding, Implementation, Testing, Production and Hand-over . . . . .	104
6.4.2.1	Generic and project specific standards . . . . .	105
6.4.2.2	Testing . . . . .	108
6.4.2.3	Conversion of old systems to the new system . . . . .	108
6.4.2.4	Possible Development Scenarios . . . . .	109
6.4.3	Software Configuration Control . . . . .	111
6.5	System Documentation . . . . .	112
6.5.1	Methods of producing Documentation . . . . .	112
6.5.2	User documentation . . . . .	113
6.5.3	Technical documentation . . . . .	114
6.6	System Administration . . . . .	114
7	Conclusions . . . . .	116

## APPENDIXES

1	A2.1	Ceeds i flaws . . . . .	117
2	A4.1	KKS and BSF Coding Systems . . . . .	118
3	A4.2	Model Structure Reference Data . . . . .	137
4	A4.3	Working within the Model to handle real world problems . .	140
5	A4.4	Current Reference Facilities . . . . .	149
6	A4.5	Current Database Structure . . . . .	154
7	A4.6	Equipment Classes being considered for detailed modelling . . . . .	157
8	A4.7	Cable Racking Facilities . . . . .	158
9	A4.8	Current DP model for Cable Design . . . . .	165
10	A4.9	Current Project Plan Link in CEEDS . . . . .	170
11	A4.10	Current and proposed Financial Models . . . . .	172
12	A4.11	Proposed Attribute Confidence Factors . . . . .	173
13	A5.1	Change Capture Facilities . . . . .	175
14	A5.2	CEEDS Contractual Documentation Production Facilities .	178
15	A5.3	Cable Sizing Algorithm . . . . .	187
16	A5.4	Cable Routing Facilities . . . . .	190
17	A5.5	CEEDS Batch Integrity Checks . . . . .	194
18	A6.1	CEEDS Naming Conventions . . . . .	195
19	A6.2	Future Plans for CEEDS Developments . . . . .	201

## **R REFERENCES AND FURTHER READING**

<b>1</b>	<b>R1 Non-Technical Issues . . . . .</b>	<b>214</b>
<b>2</b>	<b>R2 CAE and Modeling . . . . .</b>	<b>218</b>
<b>3</b>	<b>R3 Computer Environment . . . . .</b>	<b>224</b>
<b>4</b>	<b>R4 Design Methodologies and Software Engineering . . . . .</b>	<b>227</b>

<b>G</b>	<b>GLOSSARY OF ABBREVIATIONS . . . . .</b>	<b>232</b>
----------	--	------------

## LIST OF FIGURES

### Main Report Figures

1	F2.1.01 CAE at a Corporate Level . . . . .	4
2	F2.1.02 Possible Functional Corporate model for ESKOM . . . . .	5
3	F2.1.03 CAE at a Discipline Level . . . . .	6
4	F2.2.01 CEEDS history in ESKOM . . . . .	8
5	F2.2.02 CEEDS 1 . . . . .	9
6	F2.2.03 CEEDS 2 . . . . .	11
7	F2.4.01 Waves of CAE Integration . . . . .	14
8	F3.1.01 Factors influencing CAE in an Organisation . . . . .	17
9	F3.2.01 Performance Change Curve . . . . .	19
10	F3.2.02 Up-front design v's Traditional design . . . . .	20
11	F4.1.01 The Physical, Financial and Design Process Models . . . . .	30
12	F4.2.01 Hierarchical, Directional Multi-Graph Node-Link Network . . . . .	33
13	F4.2.02 From Physical to Conceptual to Representation . . . . .	34
14	F4.2.03 Tree Representation . . . . .	35
15	F4.2.04 Macro, Mid and Micro Levels of Detail . . . . .	36
16	F4.2.05 Coding Analogy . . . . .	37
17	F4.2.06 Coding Example . . . . .	37
18	F4.2.07 Relations between Volumes to find suitable routes . . . . .	53
19	F4.2.08 Relations between various Screen Windows (3D,2D and Text) and the underlying models. . . . .	42
20	F4.2.09 Relations between Model data, views of the model and documents . . . . .	43



21	F4.2.10A CEEDS Relational model (Main Model) . . . . .	46
22	F4.2.10B CEEDS Relational model (Software Aspects) . . . . .	47
23	<del>F4.2.01</del> Relationship between the Project Plan and Design Procedure Model . . . . .	57
24	F4.3.02 Relationship between the Design Process Model and the Physical model . . . . .	58
25	F4.3.03 State Dependency Graph for the Design Process Model .	61
26	F4.3.04 The Communication Protocol between the DPM and PMMF's . . . . .	60
27	F5.5.01 Standard report facilities and reports as objects . . . . .	84
28	F5.5.02 Comparison between Version Oriented and Change Oriented Configuration Control . . . . .	87
29	F6.2.01 Proposed new CEEDS computer platform . . . . .	99

## Appendix Figures

30	FA4.2.01 CEEDS Model Structure Reference Database . . . . .	137
31	FA4.3.01 Modelling of a Distribution Board . . . . .	142
32	FA4.3.02 Modelling of Sub-busbars . . . . .	143
33	FA4.3.03 Labeling Problems due to KKS Rules . . . . .	144
34	FA4.3.04 Switchgear with Daisy Chained Loads . . . . .	145
35	FA4.3.05 Single Node with Multiple Loads . . . . .	146
36	FA4.4.01 Rack Filling Criteria . . . . .	150
37	FA4.4.02 Switchgear Standards . . . . .	152
38	FA4.5.01 CEEDS Relational Model . . . . .	155
39	FA4.7.01 Cable Racking Model . . . . .	159
40	FA4.8.01 CEEDS Design Process Model . . . . .	166
41	FA4.9.01 CUE Representation of a WBS and its record structure within the CEEDS database . . . . .	171
42	FA5.1.01 CEEDS Change Capture Facilities . . . . .	177
43	FA5.4.01 Routing Program Matrix Layouts . . . . .	191
44	FA6.1.01 CEEDS Menu structure naming convention . . . . .	195
45	FA6.1.02 CEEDS Screen Form naming conventions . . . . .	196
46	FA6.1.03 CEEDS Report naming Conventions . . . . .	197
47	FA6.1.04 CEEDS Table naming Conventions . . . . .	198
48	FA6.1.05 CEEDS Menu Layout Examples . . . . .	200

## 1 Introduction

An Integrated Computer Aided Engineering ENVIRONMENT consists of teams of people and the computer tools they use to carry out their specific (probably inter-related) Engineering tasks, in a harmonious, free-flowing manner.

An Integrated Computer Aided Engineering SYSTEM consists of the suite of programs and databases that are capable of supporting the engineering function in a cooperative and transparent manner.

This investigation will briefly cover:

- CAE in general, (which sets the scope of this investigation);
- The history of the CEEDS system;
- A proposed Development Approach;
- High level system requirements.

The following will be reviewed in more detail:

- Non-technical Critical Success Factors (the Human aspect);
- Possible Models to support the detailed Engineering process;
- Functionality surrounding the Models<sup>1</sup>; and
- Software and System Development/Implementation aspects.<sup>2</sup>

- 
- (1) The chapters on "Models" and "System Functionality" are strongly based on work already completed for the CEEDS project and on work currently underway for the continued enhancement of CEEDS.
  - (2) Some sections of the "Software Development and Implementation Aspects" are specific to the computer environment on which CEEDS is being implemented and to the development approach (namely in-house development). These sections may not be of general applicability but may be adapted to other environments and disciplines. There is an incredible amount of common ground between very diverse applications at a conceptual level.

## 2 Integrated Computer Aided Engineering Systems

In this chapter,

- Computer Aided Engineering (CAE) is discussed from an organisational point of view;
- a brief history of the CEEDS system is presented;
- critical success factors are highlighted; and
- a development strategy is proposed.

### 2.1 General Discussion on CAE

It is felt that fully integrated information systems encompassing all information requirements of a large organisation (for example, Engineering, Marketing, Sales, Personnel, etc.) are still futuristic. The technology exists to build such systems, but organisations are generally unable to cope with such a high degree of integration. Introduction to the concepts of integration need to take place gradually and organisations need to adjust accordingly. One organisational discipline, namely Engineering, will be examined in some detail. Engineering systems can be viewed from a Corporate, Discipline and a Functional level. Corporate and Discipline levels are discussed in this chapter while the Functional level is dealt with in detail in the chapter on "System Functionality" (5). For the purpose of this study, examples are extracted from the point of view of a Power Utility Company (namely ESKOM) and a particular CAE system (CEEDS).

The Engineering division is subdivided into Power Station Engineering, Transmission Engineering and Distribution Engineering. The "operations" functions are similarly grouped. The Engineering division is responsible for the design, construction and modification of facilities. The relevant Operations divisions are responsible for the running and maintenance of these facilities. The Power Station Engineering department and their operations counterpart, "Generation", are chosen for a more detailed analysis. The principles relevant to this "partnership" are similarly applicable to other such partnerships as well as to the interface between functional groups, for example, Power Stations feeding Transmission.

#### 2.1.1 CAE at a corporate level

The following steps describe a typical definition, development and production cycle at corporate level:

- 1) The Generation Group establishes the requirements for a new power station.
- 2) Engineering is approached to establish detailed specifications against which various contractors can tender
- 3) Contracts are awarded and detailed design may begin.
- 4) Engineering acts as interface coordinator between the various contractors. Engineering also conducts some of the detailed design in-house and can therefore be considered as a contractor.
- 5) Construction begins and is coordinated by Engineering. The majority of construction is carried out by contractors.

- 6) Engineering commissions the plant and hands it over to Generation, along with the necessary documentation.
- 7) Generation operates and maintains the plant for its life time.
- 8) Major modifications or overhauls (for example, Life Extension programs) of the plant would be managed by Engineering on Generation's behalf.
- 9) The plant will finally be decommissioned and demolished (or kept as a museum).

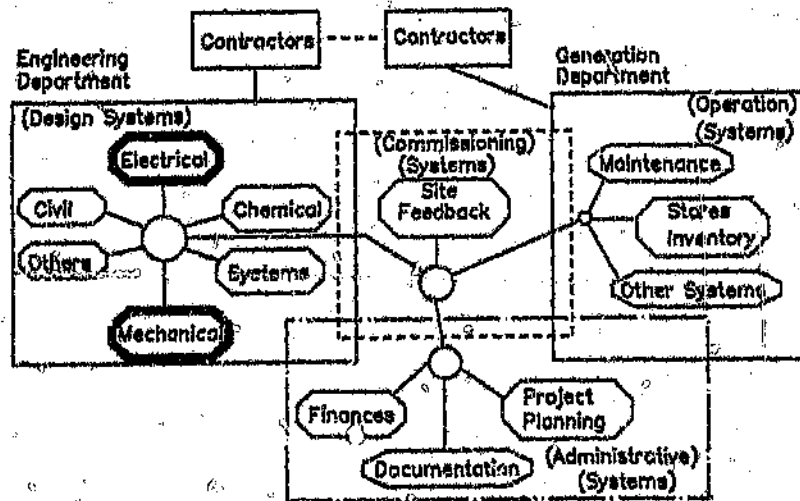
The steps presented above are a simplification of the real situation but are useful in defining the scope of activities being considered for inclusion in an integrated CAE system. Steps 4, 5, 6 and interfaces to 7 are currently considered viable for a detailed investigation.

Areas of activity where it is felt that computerised tools could be beneficial include:

- a) Engineering builds up a large amount of data relating to the detailed design of a plant (in the form of drawings, listings, manuals, etc.). Configuration management of this data is critical, especially data relating to various interacting contracts. (For example: one contractor supplies the boiler, another builds the support structure, another supplies the control systems for that boiler and yet another supplies the power to all equipment on the boiler).
- b) Engineering is also involved in some internal detailed design (Refer to paragraph "CAE at an Engineering Discipline Level" (2.1.2) for an example). Efficient capturing and control of design data is critical. Design optimisation is also important.
- c) Commissioning of such large quantities of complex plant (as found on a power station) requires a great deal of information (points (a) and (b) above). Stringent control must be carried out for safety, quality, project planning and progress monitoring purposes.
- d) On hand-over of the plant, Generation takes possession of the plant and the necessary documentation. They then have to begin to maintain the plant. This requires information about the plant, the components of the plant, maintenance procedures, diagnostic procedures and availability of spare parts.

Refer to Figures F2.1.01 and F2.1.02.

All these activities require large amounts of data. Much of this data is captured in electronic form either via a CAD system (drawings), simple databases and word processors or via various applications (e.g. design aids, maintenance packages, stores inventory systems etc.) It would seem very desirable to have the data from one system available to other systems that require it, in an electronically transferable state. This is where the ideas of system integration begin to form. There are obviously many levels of "Integration" from exchange of flat ASCII files (at a primitive level) to totally transparent, single source data usage (the ultimately desirable situation). There are just as many levels of "Control" over data interchange and integrity.



CAE at a Corporate Level.

2.1.01

Figure F2.1.01

Very briefly, the ultimate system is one in which:

- all data is stored in only the appropriate places (eliminating unnecessary data duplication and facilitating configuration control);
- this data is accessible by all systems requiring it, in a transparent (yet fully controlled) manner; whether that system be CAD, database, word processors, expert system, etc.

Technology (and standardisation thereof) is heading in the direction where this scenario may be possible. Managers, users and software development staff must just be ready to exploit the technology as it becomes available and must be very wary about being dragged into a single vendor's proposed "solutions".

What would seem to be a sensible approach would be to establish the desirable end goal and start working towards it in achievable incremental steps. This investigation is one such step. The area concentrated on is that of in-house Electrical Design activities, electrical construction and the necessary interfaces to other systems that have an influence on these activities.

POSSIBLE FUNCTIONAL CORPORATE MODEL FOR ESKOM

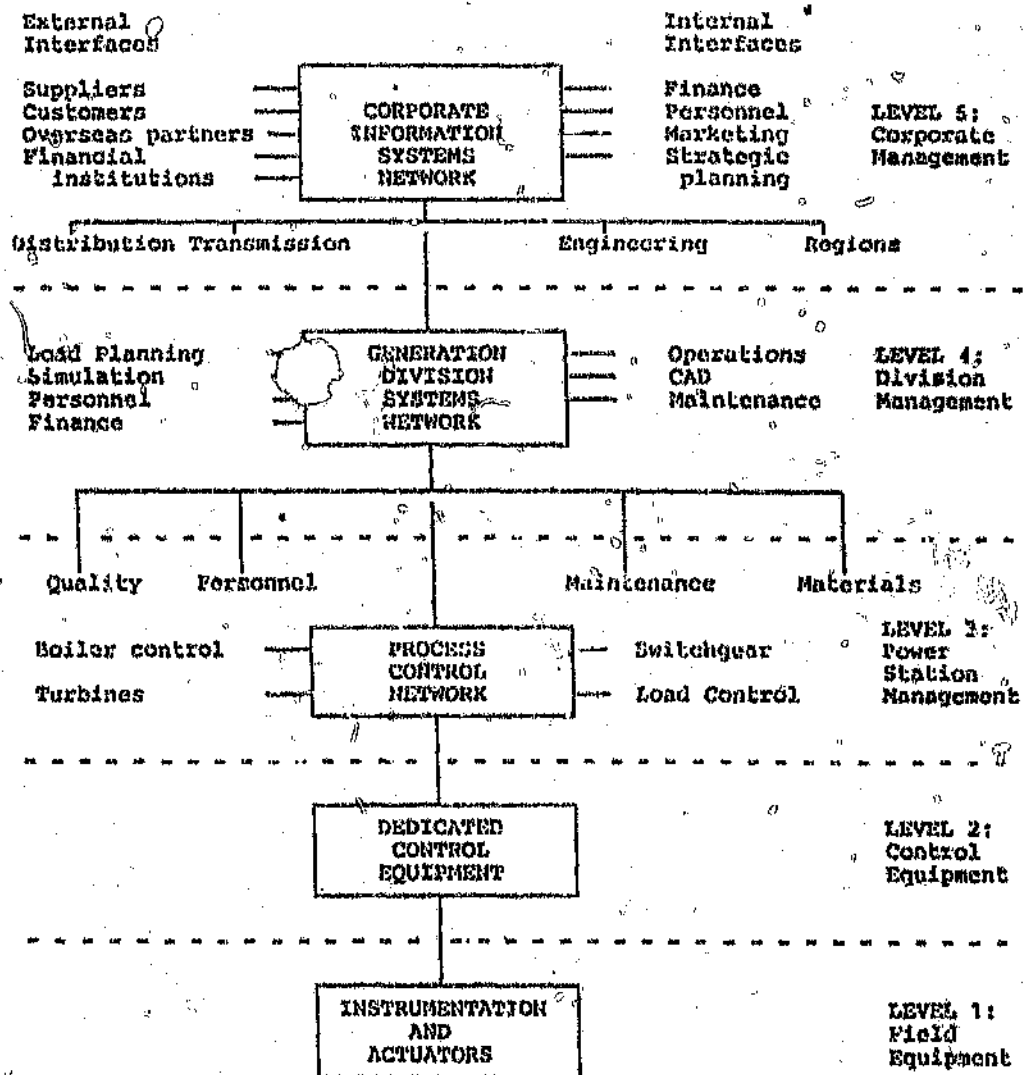
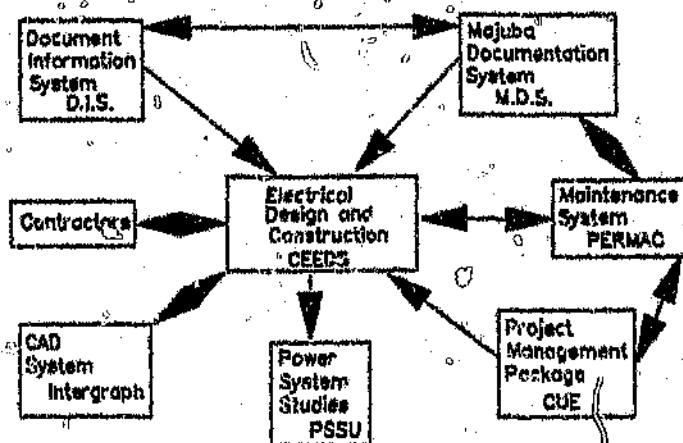


Figure F2.1.02 (Source: ESKOM IT Strategy Workshop)

### 2.1.2 CAE at an engineering discipline level

The Power Station Electrical Engineering Department (PSEED) is responsible for, amongst many other tasks, the detailed design and management of:

1. The Auxiliary Power Supply system; (which includes switchgear, transformers, emergency supply equipment etc.);
2. Lighting;
3. Certain cabling (e.g. power cables);
4. Cable Support Structures;
5. Certain Control systems.



CAE at a Discipline Level.

Figure F2.1.03

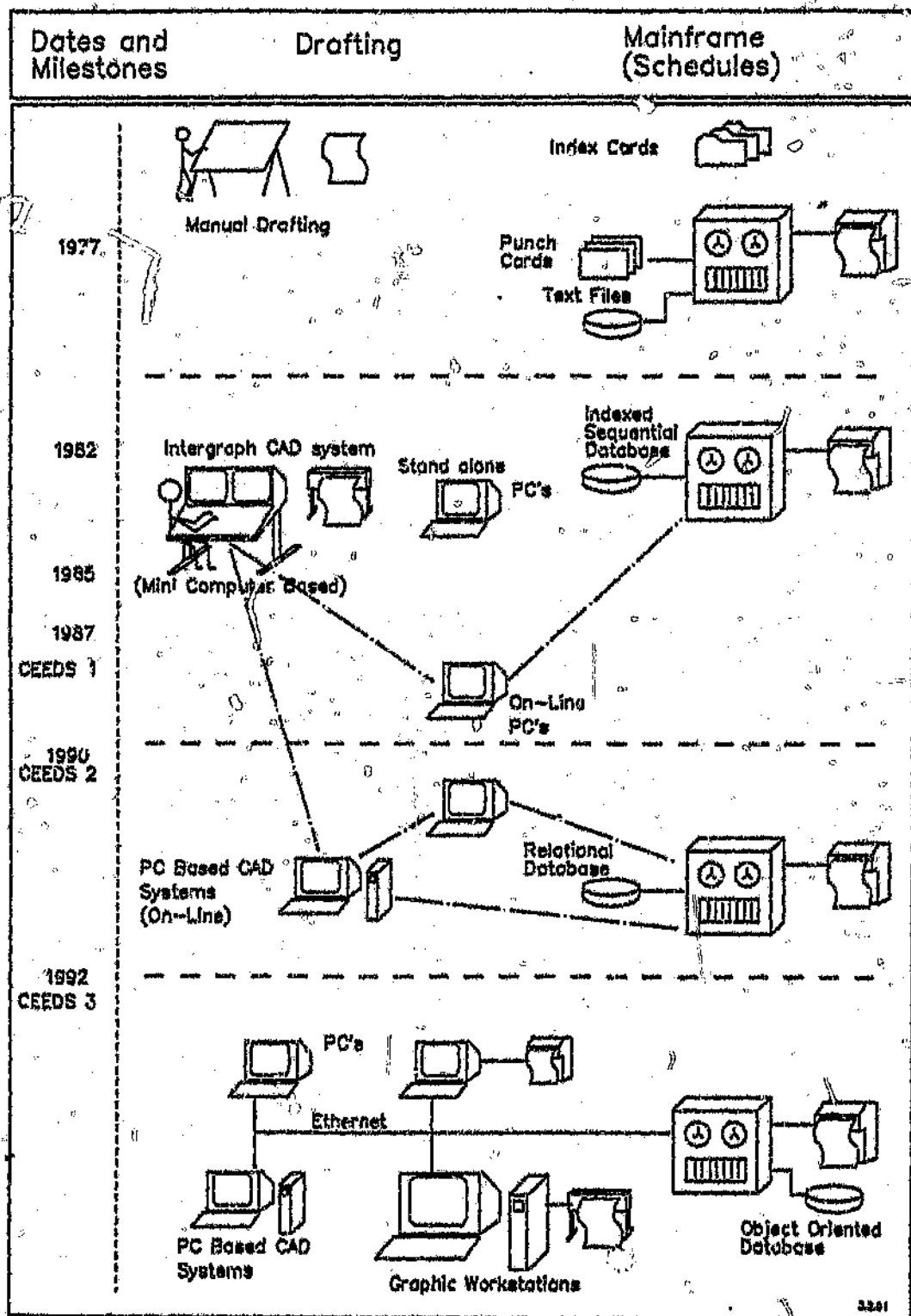


Equipment requiring power (and their details) must be provided by the various contractors. A supply network is then designed to meet these requirements. Equipment, control and cabling requirements of the supply network are defined and cable support structures are designed to meet the cabling needs. Design documentation is then produced for construction (and maintenance) purposes. During construction, feedback from site is used to measure progress and to control payments to contractors for work done. On plant hand-over, certain design data is loaded into the maintenance package, the Engineering Department then withdraws and Generation begin their task of operating and maintaining the plant.

It is not intended to go into detailed analysis of the system requirements to carry out these tasks. The intention here is to show the interfaces and interactions between the various systems involved, with the purpose of setting the scene for the "System Models" and "System Functionality" sections below. The CEEDS information system serves well as an example. Currently, most electrical Engineering data is resident in CEEDS and/or on the CAD system to fulfil the needs discussed above. Figure F2.1.03 provides an overview of the CEEDS system and its interactions with other systems.

The following provides more detail to be referred to along with Figure F2.1.03:

- PERMAC is the Generation Maintenance Package (Purchased Package, MVS Mainframe);
- CUE is the Project Management Package used by Engineering (Purchased Package, MVS Mainframe);
- DIS is the Engineering Documentation System (Developed In-house, VM Mainframe, Oracle Database);
- MDS is the Documentation Configuration Management system being used on Majuba (duplication with DIS due to political reasons) (Developed In-house, VM Mainframe, Oracle Database);
- CAD is the Electrical Engineering Computer Aided Drafting System (Purchased Intergraph software, PC's and Vax mini-computer);
- PSSU is a Power Network Analysis Package (Purchased Package, PC and Apollo workstation based).



CEEDS History in ESKOM.

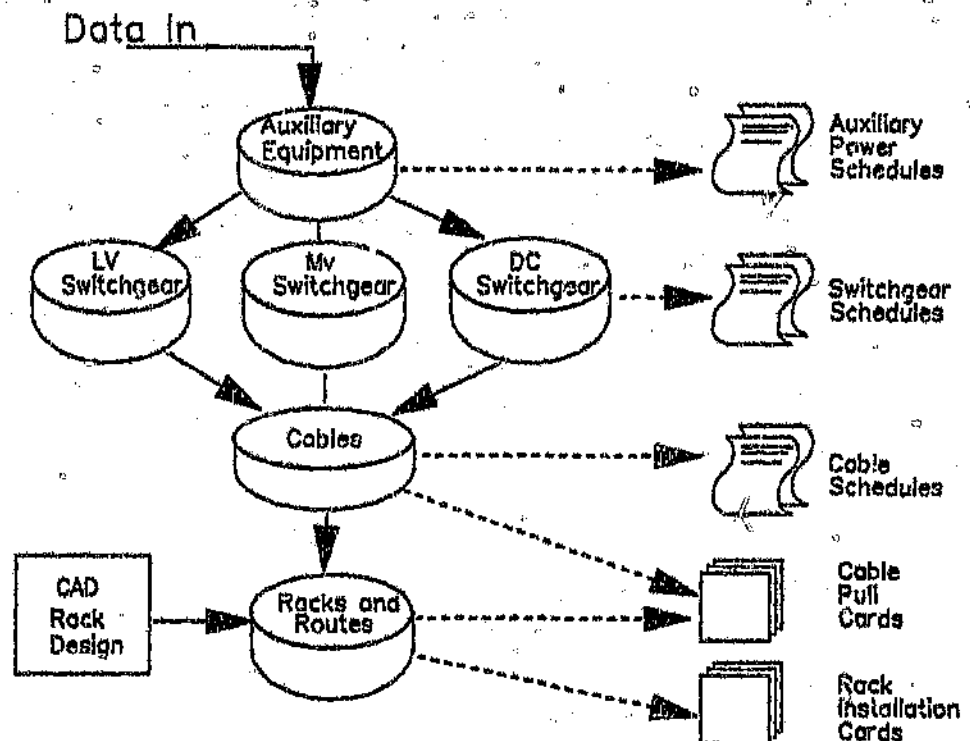
Figure F2.2.01

## 2.2 Brief History of CAE and CEEDS in ESKOM

### 2.2.1 Setting the scene for CEEDS

Refer to Figure F2.2.01. Initially the whole engineering design process was done manually. Drafting was done manually on drawing boards and "schedule" (list) type information was either kept on pre-printed sheets or index cards. A CAD system was introduced in 1982 and most of the manual drafting gradually moved over to the CAD. Some sections, the architects for example, are still doing manual drafting. The CAD system was simply a computerised replacement of drawing boards. Schedule type information was placed on a master overlay and details manually entered. At about the same time, Mainframe computers were also increasingly used for storing certain schedule information. These facilities started off with punch card technology and flat files, moved to on-line editing of flat files, then onto Indexed-Sequential and Hierarchical Indexed-Sequential Databases. There was no automatic correlation between CAD activity and Mainframe data, and in many cases there was no manual correlation either. PC's started to make inroads in about 1982 and some pro-active people started to keep individual schedules, lists etc. This simply compounded the correlation problem and strengthened functional islanding.

In the early days of the Majuba Power Station Project (1986), it was discovered that the "Auxiliary Power Schedule" and the "Cable Schedule" (two important base documents) were on Mainframe and practically everything else was on CAD. It was then that the idea of a set of "Cooperating databases" was established whereby data could be "poured" into the top (The Auxiliary Power Schedule Database), filter into a set of Switchgear and Transformer databases and down into the Cable Schedule Database. Refer to Figure F2.2.02.



CEEDS I.  
Figure F2.2.02

### 2.2.2 CEEDS I

Since CEEDS I was developed on a Cyber 175 using the IPF2 Hierarchical Indexed-Sequential Database system and later Fortran Advanced Access Methods (for increased speed).

Based on the new CEEDS system, ESKOM made a decision to manage the cable contract for Majuba in-house. Previous power station projects contracted out for this function. This provided the necessary incentive to get the system working properly.

Cable Racking design was being done on the CAD system at that time and software for data extraction from CAD had become available (1987). An ambitious addition was made to CEEDS to extract three dimensional (3D) racking information from the CAD system to enable the optimised routing of Cables identified on CEEDS.

The system performed adequately in that it produced an Auxiliary Power Schedule, Cable Schedules, Switchgear Schedules, Rack Installation cards and Cable pull cards. One of its major contributions though, was that it had introduced the division to "Integrated Computer Facilities". There was extreme resistance to this new technology as traditional roles were being challenged and the "Black Box" ("Blik Brein") syndrome set in. People felt that they no longer had full autonomy over their activities. Despite such feelings, some degree of acceptance was evident.

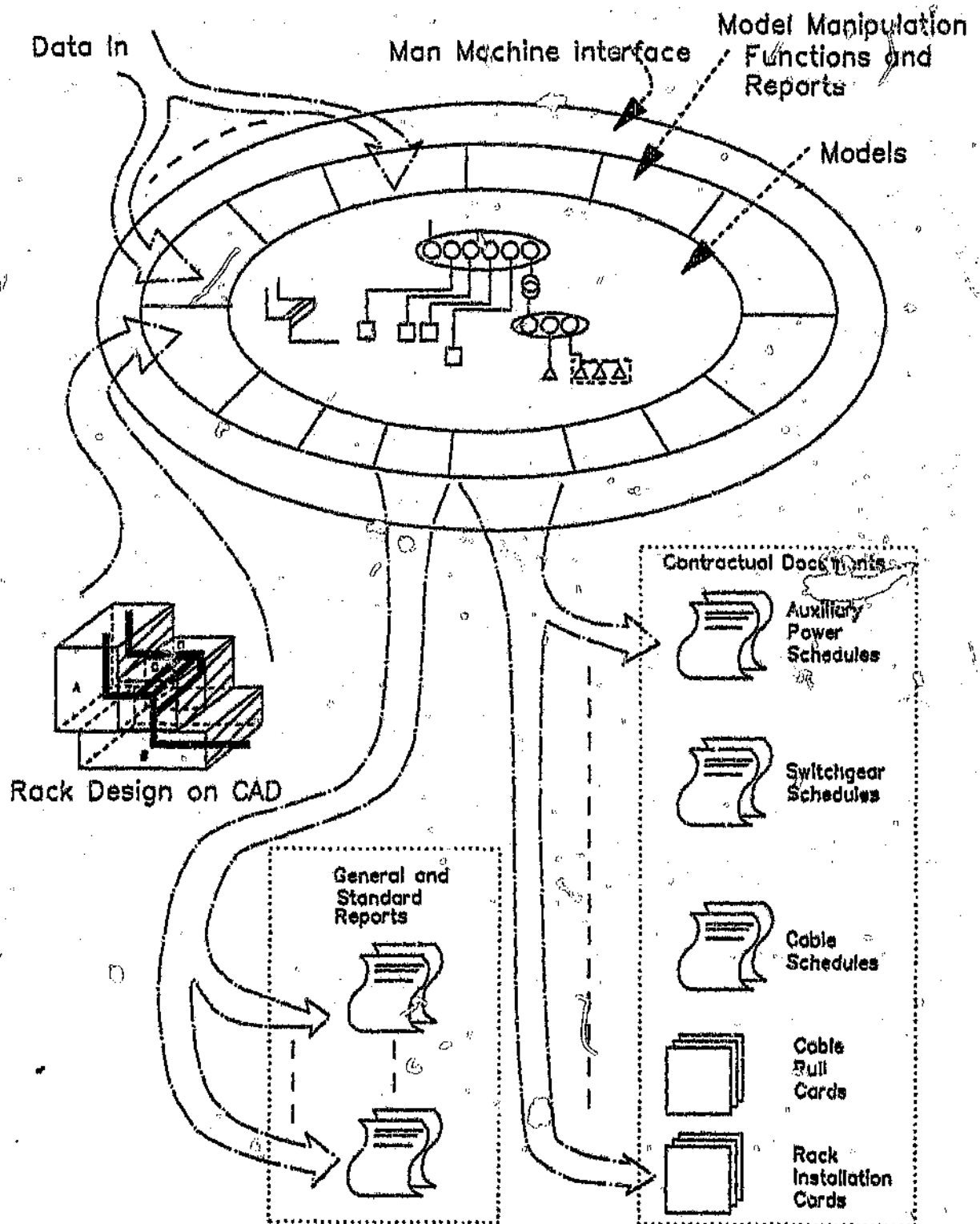
CEEDS I had a number of inherent flaws, many of which are contributory factors to system failure. (Refer to Appendix A2.1 for details). At the beginning of 1989, it was decided that the Cyber was to be decommissioned and replaced with an IBM. The Oracle relational database was purchased for the new machine. CEEDS would have to move off the old Cyber and onto the new IBM. This was a prime opportunity to carry out a re-design of the CEEDS system. This led to the CEEDS II development, much of which is detailed in this report.

### 2.2.3 CEEDS II

The Software environment and development methodology standards were established first (in the absence of any existing standards). System Requirements analysis began in September 1989, Coding was phased in during April 1990 and the current system was implemented by December 1990.

Current Application characteristics: (Refer to Figure F2.2.03.)

- This system consists of a "physical model" and parts of the cabling "financial model". It has rudimentary cable design process facilities built into the functions that manipulate cables. (Refer to the chapter on "System Models" (4) for details about these models). There is a "link" into the CAD system in order to import Rack designs and the physical locations (x,y,z coordinates) of equipment. External data (equipment and cables captured by contractors on CEEDS PC software) can be imported.



CEEDS 2.

3.2.03

Figure F2.2.03

- There is no automatic access to data in interfacing systems. This is mainly because they are purchased packages and are resident on a different mainframe. Necessary Data transfer is via flat files on a once off or periodic basis. The interface is purely textual, although data in specified format can be exported to programs written for the CAD system so that a graphical representation can be generated.
- Contractual data modification auditing and system activity logging are available.
- Documentation production and configuration management facilities are available (but can do with some enhancement).

#### The current Software Situation:

- The Oracle Relational database manager forms the basis of the system. Most data capture and manipulation takes place using Oracle products such as Structured Query Language (SQL), SQL Forms for screen forms, and a Fortran/SQL Precompiler for more complex data retrieval and manipulation.
- The user interface is totally menu driven and has user specific access to menu options. Flexible query and reporting facilities are available that make it possible to produce almost any desired report and to extract data in any desired format (for exporting to other systems).
- The system has its own set of software configuration management tools integrated into the system. This facilitates limited automatic software documentation production and allows maintenance staff to query the system to determine the scope of change for any required modifications.

The system is still essentially stand alone but has reasonably flexible importing and exporting facilities. Data transfer between CAD and CEEDS is still via flat files. The main restriction has been non-transparent computer environments.

#### Future activities (briefly):

- The system will migrate to a more suitable computer platform consisting of networked workstations, high performance PC's and the mainframe acting as a database server. The first stage has already begun with the delivery of a suitable workstation that is being coupled to an Ethernet LAN that will eventually have a link into the mainframe system.
- Creating a consistent graphical front end for the models.
- Increasingly transparent interaction with cooperating systems and the establishment of centralised sources of common data.
- Improved Models and Functionality.
- Introduction of Artificial Intelligence (AI) type software for various functions.

One of the main reason for this investigation has been to gain insight into the requirements for these future activities.

## 2.3 Critical Success Factors

There are a number of factors contributing to the disturbingly high failure rate in trying to implement systems successfully (or even partially so).

Critical factors can be divided into:

- Non-Technical Factors such as:
  - Skills training;
  - Changes to procedures;
  - Organisational Structure;
  - Company Strategy; and
  - Organisational culture.

Refer to "Non-Technical aspects" (3) for more detail.

- Technological Factors such as:
  - A sound theoretical and consistent foundation (Model) for tools;
  - A sufficiently rich, flexible and reliable Computer Environment; and
  - System maintainability and expendability.

Refer to "System Models" (4), "System Functionality" (5) and "Software and Development /Implementation aspects" (6) for more detail.

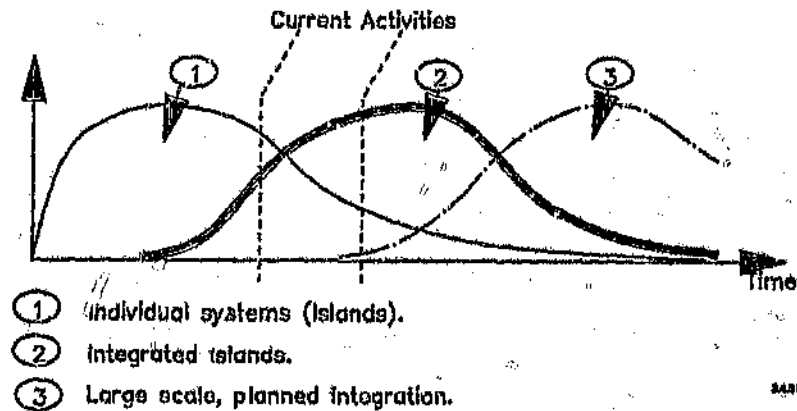
Experience indicates that early failures (the majority) are mainly caused by non-technical issues such as resistance to change and inability to adjust organisationally. Failures further down the line are influenced by both technical and non-technical aspects. The later into the life cycle that faults occur, the more likely it is that inadequate or incorrect maintenance has led to a "fragile" system. One that can no longer be modified to meet changing requirements. (The "fix one bug, create three more" syndrome).

## 2.4 High Level Development Strategy

A strategy that will work needs to be established. It is anticipated that there are three major waves of system development that take place within any organisation.

- The first is where individuals or small teams begin to build up their own sets of tools to aid them in their detailed day-to-day activities.
- The second is where these individuals and teams begin to realise that the teams they interact with also have tools and a lot of commonality exists (data and functionality). Some cooperation begins to take place and it is realised that a concerted effort must be made to rationalise the proliferation of small systems.
- The third wave is where such rationalisation has already taken place in relatively large "islands" and a concerted effort must now be made to integrate at a corporate level.

These waves have large overlaps and cannot always be clearly distinguished. Refer to Figure F2.4.01, ESKOM in general is currently at the beginning of the second wave, although some development teams are approaching the third wave.



### Waves of CAE Integration.

Figure F2.4.01

In the development of any large multi-faceted project that involves and affects many people, a sound strategy must be established if any semblance of success is to be hoped for. It is extremely important to have a very clear end goal in order to avoid the "moving goal post" scenario and the "I'll do it my way" empire building phenomenon.

There are essentially two extreme strategies with a continuum in-between:

1. Initially develop islands of small systems, then attempt to centralise and integrate them (Bottom-Up approach); or
2. Gradually create one large centralised system and then allow de-centralised control (Top-Down approach).

Both of these extremes have glaring flaws that make neither suitable on their own. Using the first approach, it is unrealistic to expect the systems to be sufficiently compatible to allow integration without major re-designs. There is also the danger that once a system suits a particular group's immediate needs, they will be unwilling to cooperate in any activity advocating change to their system. A "Let the other team change their system" attitude could set in.

Using the second approach, it could take so long before the system is usable that individual groups would revert to the first method to meet their own needs.

The recommended strategy is to attempt to use the best of both extremes [MGT 16]:

- 1) Start with a few large islands. For example, each Engineering discipline (Electrical, Mechanical, Civil, etc.) would be an island, Generation another;



- 2) Establish a global strategy on a centralised basis (to allow for the third wave) and allow the individual islands to sort out their own detail. (Make sure that the organisational body looking after the interests of the centralised facilities has sufficient authority to eventually enforce the use of these facilities by individual islands).
- 3) Establish which data needs to be centrally accessible and the sources and administration of such data.
- 4) Establish interface requirements between islands. Islands should be established that have clean interfaces, i.e. there should be as much independence as possible between chosen islands. For example, disciplines should not be broken into further islands, (such as Process control and Electrical systems, or Boiler plant and Turbine plant) because of the high degree of interdependence between such similar activities.
- 5) Development can then begin individually within each island in such a manner that best suits the requirements of each island. For example, mechanical engineers may have more need of CAD facilities and less need for supporting databases whereas with electrical engineering, it may be the other way around.

There are certain pre-requisites applicable to the recommended approach:

- 1) The requirements of the centralised facilities and the interface requirements must be taken into account during development. Initially these facilities can be duplicated or simulated within the individual islands (to facilitate speedier development), but they must eventually be integrated.
- 2) The integration must be a definite, planned-for activity, not just something that everyone knows must take place at some time or another, but has no immediate importance.
- 3) Development should be grouped into phases in such a manner that immediate needs can be met within as short a time as necessary. Progressive enhancements can then be made to the system. Some of these enhancements will include the interfaces to centralised facilities and to other islands.
- 4) A flexible scope should be established and tasks prioritised to suit.
- 5) In order to allow a system to evolve successfully, the foundations upon which further facilities are built must be very stable. A very clear and stable philosophy must be adhered to when expanding the system. If this is not done, any change to requirements of the core system could cause failure in all dependent systems. What may seem like a simple decision by management to "slightly" change a philosophy could invalidate many man years of system development work. The "System Models" chapter is an attempt at establishing such a foundation for the Power Station Electrical Engineering Discipline.

The question may be asked, "Why not just skip the second wave and progress directly to the third more desirable wave?". There are a number of reasons:

- Each wave requires a very sound basis upon which to be built. It is usually the preceding wave that provides such a basis. (Such a basis could be purchased at a price, but the general organisation would lack the necessary culture. This culture, on the other hand, cannot be purchased without a radical staff turnover);
- Organisational culture can very seldom handle revolutionary change, especially in large organisations; and
- The sheer lack of people in the organisation experienced in integrated CAE implementations. (This expertise will hopefully be built up during preceding waves).

It is important to note the following:

- The establishment of centralised functions and inter-island interfaces **MUST** be de-politicised. Logic and Information requirements must be the driving factors and not empire protection or personal ambition.
- All parties that could be participants must be identified and involved in Interface discussions. This is mainly to prevent unnecessary duplication of facilities and data. Duplication invariably leads to integrity problems such as having different groups working on the same job using different and often conflicting information.
- Company wide Standards must be established and enforced.
- Phased Implementation must not be overlooked because of support needed for old designs using old methods. This has the effect of increasing manpower costs for possibly a long period, during which multiple administrative systems are supported.
- Implementation speed should never exceed the rate of retraining of the users of the systems or the rate at which new administrative procedures can be established.
- It must be decided up-front whether a given system is to support multiple projects or whether a framework is to be developed that could be customised to meet the detailed needs of an individual project. For example, would a system be utilised in the design and construction of a new facility as well as in the management of old facilities. The fact that any system tends to be very project-specific is often underestimated. If the ability to customize the system is not provided for at the beginning, modifying an existing system to meet the requirements of another project could be a prohibitively expensive undertaking. This dependence on the current state of affairs is not only restricted to systems developed for islands but affects the whole integrated system. For example, if on one project it is decided to contract out the whole management function, and on another project, this function is managed within the company, then the specifications for an island's system as well as most interfaces could be radically different.

Additional factors to be considered by all parties involved in the development of such a system are:

- The **REALITIES** of such a system must not be overlooked. The complexity of such systems seldom surfaces during initial discussions. Real problems must not be over-simplified and then expectations placed on a system (developed to solve the simplified problem) to solve the Real problem(s);
- The development of such systems takes substantial time and effort;
- The size of such systems must not be under-estimated.

Software developers should pay attention to the following considerations:

- The system being developed is a **TOOL** for Engineers and Designers, not a theoretical exercise in computer modelling.
- Maintainable and Usable software must be written. Rather cut back on sophistication than creating a massively complex system that "falls over" the first time it is launched into the real world or shatters when it is attempted to tweak something within. Even worse is if the system degrades (or corrupts data) in a manner difficult to detect so that when it is noticed that something is wrong, it's too late.

All of the above factors should not be taken lightly as they are all major contributors to system failure.

### 3 Non-Technical Aspects

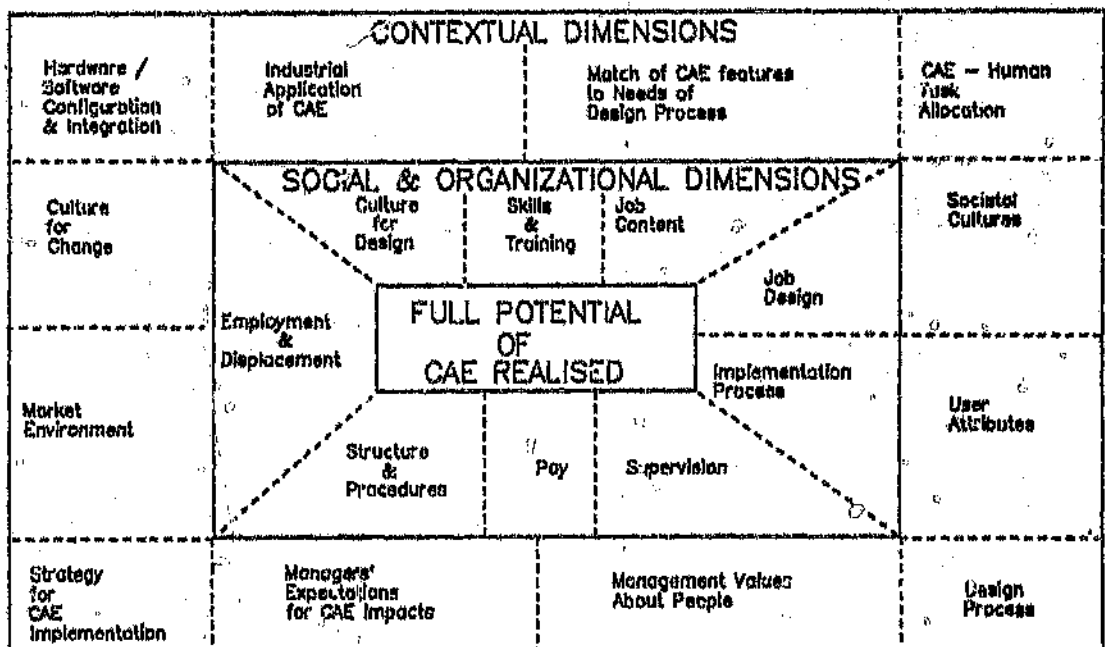
#### 3.1 Introduction

There are a large number of factors which contribute to the success or failure of CAE within an organisation. The majority of these factors are not technical in nature. When management is thinking of introducing CAE systems, they must be aware of these issues as the track record shows very few fully successful implementations [MGT 94]. The vast majority of these systems were doomed to failure, even before the question of specific technology was addressed, simply because of some general misconceptions and myths about the effect of introducing advanced technology into an organisation.

The following factors will be covered in some detail:

- Organisational culture:
  - Structure (e.g. Hierarchical/matrix);
  - Strategy (How and why is CAE introduced);
- Managerial role and commitment;
- Changes to procedures;
- Correct employee attitudes (new culture);
- Skills training; and
- Developers' attitudes.

Refer to Figure F3.1.01



Source: IEEE Transactions on Engineering Management  
August 1989, Vol. 36, Number 3.

4.101

Factors influencing CAE in an Organisation

Figure F3.1.01

## 3.2 Organisational Aspects

### 3.2.1 Organisational Culture

Successful implementation of advanced technological systems can only be achieved where social factors such as job content, training, organizational change, etc. are taken into account and planned for in advance to implementation (or at least during implementation).

#### **The introduction of technology will bring about automatic change.**

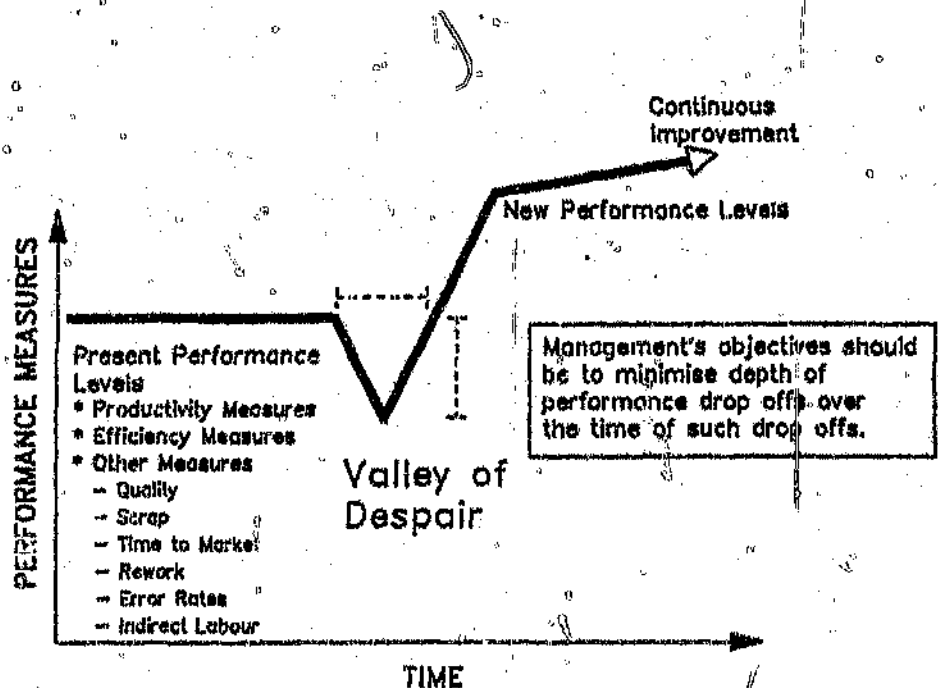
The above statement is one of the major misconceptions about the introduction of technology. "Let's put the system in place and see what happens" soon leads to "What's going wrong?, one would swear we've never done this job before!!". It is the integration of data and functionality, made possible by the introduction of a technologically advanced system, that encourage and promote change. The change itself must be planned for and orchestrated by management [MGT 03]. This is not a job that can be delegated down. It has also been shown that more success is achieved in high-trust environments (although there are cases of success in low trust environments) [MGT 02].

### 3.2.2 Right and wrong reasons for the introduction of CAE

- If the integrated system is designed to simply automate or replace existing practice without using the computers' inherent benefits, it will be an in-effective and expensive exercise for no few benefits. For example, using CAD purely as an electronic drawing board (for producing neater drawings) or using a database simply to produce a specific list (previously done manually) are only the tip of the iceberg regarding the inherent benefits of using computer technology. Computer technology can create tools, but to gain maximum benefits it must form an integral part of a company's culture. Tools should not be thought of as "machine systems" but as "Person-machine systems", an integrated partnership between person and computer.
- If the system is regarded purely as a capital-for-labour replacement i.e. in the hope of reducing staff, failure is inevitable. This is a favorite ploy of vendors who advocate the potential of incredible productivity increases, which in turn can be equated to a corresponding reduction in staff to carry out the current work load. In many cases trends indicate just the opposite [MGT 08]. Roles change but people aren't replaced. An actual increase in labour is inevitable if the new system is mismanaged and if parallel old system(s) are maintained for too long. Trends also indicate that operator and management training is seldom adequate to achieve anywhere near the productivity gains claimed by vendors. [Most references under MGT highlight the training problem].
- Successful implementations have been driven by using new technology as a strategic weapon, to reduce design cycle times, design more effectively, reduce information conflict and integrity problems, improve product quality, rationalise operations, etc.

### 3.2.3 Productivity and quality

- CAE has the potential to increase productivity and service / product quality. This can be achieved by skillful use of the inherent benefits of CAE.
- One of the main benefits is reduced cycle time. A change in design detail can be managed far more quickly using CAE than using manual methods. Design integrity also remains intact with a good CAE system. A design can therefore go through far more refinements than previously and thereby improving quality. Case studies have shown that reduced iteration time can often pay for the investment in technology [MGT 01].
- Time saving due to the concurrent nature of design, made possible by CAE, is another benefit. (One designer doesn't have to wait for the completion of one activity before starting another, Refer to "Design Philosophies, Procedures, Operations and Administration" (3.4) below).
- Productivity gains will inevitably go through an initial "valley of despair". Refer to Figure F3.2.01. This is mainly due to unfamiliarity with the system and with new procedures surrounding the system. Management must not overreact and revert back to old methods or try other changes too soon. Such "oversteering" simply leads to confusion and loss of direction. The length and depth of this "valley" are very dependant on training (too many people in training leads to sharp drops in productivity while insufficient training prolongs the period of reduced productivity).



Source: Anderson Consulting Change Management Services

1291

Performance Change Curve.

Figure F3.2.01

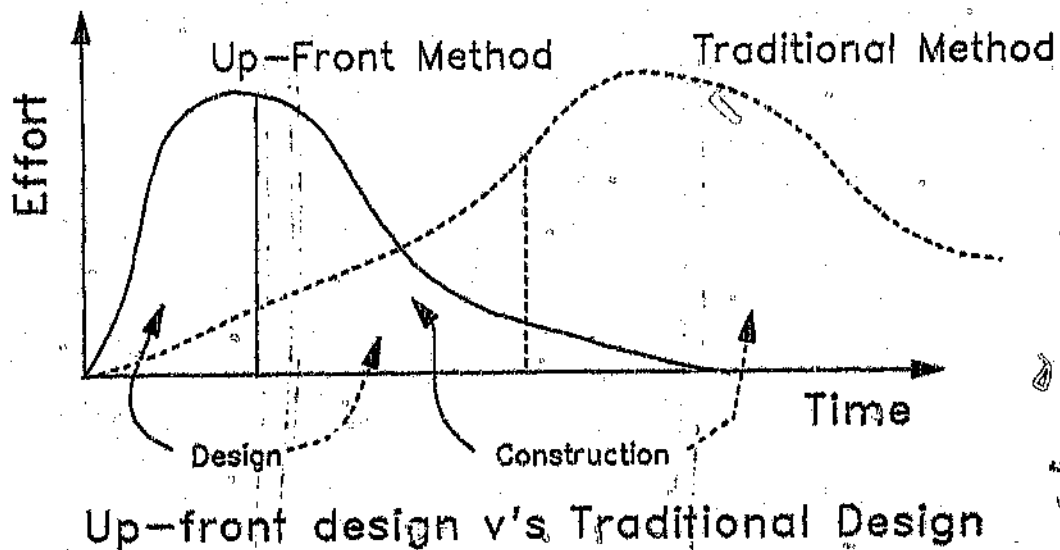


Figure F3.2.02

- Experience has shown that a higher quality of preliminary design is required to best suit computer tools. Refer to Figure F3.2.02. A major benefit is that design errors are trapped during the design phase and not left to be sorted out during construction. Errors are far cheaper to correct during design than after equipment has arrived on site and found to be incompatible with related equipment. Moving a line on a CAD to indicate a cable being moved from one termination to another is obviously far cheaper than having to uncouple and discard an incorrect cable and then pull and terminate the new cable on site. "Get it right first time" attitudes should be encouraged.

### 3.2.4 Differences in the nature of work which affect implementation success.

- Electronics industries (especially Integrated Circuit (IC) and Printed Circuit Board (PCB) production) have some extremely successful CAE systems. Design is highly automated, manufacture is fed directly from design and testing is almost totally automated. There is obviously a very close link between the electronics industry and CAE technology, so it seems logical that this industry would make maximum use of CAE.
- More traditional disciplines, such as heavy current, must work harder to change. The industry operates within a highly regulated environment, and work is usually tender-based. The groups setting up specifications, doing the design and manufacturing the equipment are usually far removed and often belong to different companies. This is in total contradiction to the electronics industry. It is difficult to rationalise products into families for multiple use, for example, modularised switchgear components that can be combined into various "standard" combinations (again the electronics industry is way ahead). The organisational structure tends to be fragmented into specialist groups with little expertise in total system design [MGT 13].
- Heavy current is also a small market and has not influenced commercial CAD/CAE products as have the aerospace-, electronics- and automotive industries.

- Integrated environments work best in smaller teams with limited design scope. It is more difficult in companies with wide scope and low volume and where products range from components to complete systems.
- Product rationalisation and standardisation are necessary to get the best out of CAE systems. If a design can be done by simply selecting and configuring suitable pre-designed building blocks, CAE can be extremely successful. Analysis of the configured system (using CAE) could then be carried out to optimise the system. (The design of electrical supply systems could fall into this category).

### 3.2.5 Change is continuous

Today's "Hi-Tech" is tomorrow's "Old Hat". The rate of change is increasing with time and attitudes and organisational cultures will have to cope in order to survive. Managers must be wary though of attempting too much change too quickly. There is a limit to what an organisation can handle and survival is a delicate balance between rapid adjustment and self destruction.

## 3.3 Management role

The degree of success with which CAE systems are implemented has less to do with technology than with the management of technology within the human and organisational constraints of an organisation.

### 3.3.1 Commitment to change from management.

Direct involvement by Managers is vital. It does not help giving lots of verbose, verbal commitment with little visible action. I.e. get the ball rolling, then stand back and exercise "arms-length" management. This will not work. The managers of the users of the system must be involved in the details of the system (including the development). One cannot rely on a system and have no idea of what it consists of or how it is to be used. Managers need to control the development of the system (thereby gaining the necessary insight into what is involved), and they must control the procedures surrounding the use of the system. There is a tendency for a gap to form between team leaders and middle management. Unless management becomes involved, islanding is inevitable and integration becomes impossible. Managers are responsible for "Playing the Politics" in breaking down independent islands of activity. Unless this is done, any attempt at integration is seen as meddling in another's affairs and will be rejected.

Managers must at least have a basic understanding of the technology used by CAE systems and they must be trained to manage change due to the introduction of advanced technology. Without such understanding and skills, management expectations tend to be totally unrealistic. The misinformed either see computers as mystical boxes that can do wonderful things at the "press of a button", or as "Big Brother" that messes up telephone accounts. Development time and effort tend to be grossly under-estimated. A person who has developed a small spreadsheet and perhaps even written a small BASIC program is placed under the misconception that software development is easy. Development effort increases exponentially with project size [SM 08].

Expectations of what the system can do also tend to be overrated. This is mainly due to "sales pressure" talk from the marketing oriented developer. Emphasis is placed on the "fancy" features while the amount of up-front preparation work, to make use of these features, is conveniently glossed over. When the system is finally implemented, unrealistic demands are made on the system. System Managers must be aware of the details of the technology being used by their subordinates. If this is not so, realistic procedures cannot be established and the resulting lack of respect by subordinates will lead to cooperation problems.

### 3.3.2 Resource planning

The introduction of CAE does not occur overnight. It must be planned for and resources allocated accordingly. Some factors, due to the introduction of CAE, affecting resource allocation include:

- Re-training of people. (Refer to the paragraphs on "Training" (3.6) below);
- Time investment of key people in designing and implementing the system. Many people, usually fully allocated to their normal work, will have to spend quite a substantial amount of time with the system developers and implementation staff. This time is an overhead and must be accounted for;
- Plan for staff turnover (especially key personnel). The implementation of a system can be severely hampered by the untimely removal of key people. Contingency plans should be made (if resources allow it);
- The effects of unions and workforce demarcation need to be accounted for;
- The availability of skilled manpower must be taken into account. It is no good having a hi-tech solution with no people to run it;
- Development costs in terms of people, time, hardware and software. This cost is a major negotiation point when external contractors do the development, but is often overlooked when in-house development is done.

### 3.3.3 Non-traditional Design

Managers must be aware of "new (non-traditional) mechanisms" in the design process. (Refer to the paragraph on "Procedures and Administration" (3.4.2) below). People's roles need to be more carefully delimited because what one person does directly affects his neighbour. Previously, designers would have worked in isolation, compared notes at the end of the day, then gone back for the next iteration. The loss of visibility of an individual's contribution in a centralised, integrated environment must be compensated for by other intrinsic rewards.

### 3.3.4 The importance of system ownership

A project champion is important. Without pro-active people in an organisation, stagnation and loss of competitive spirit results. The introduction of CAE is one of those activities that places strain on an organisation and someone must create and maintain the momentum required for eventual success.



Once development is complete, the system is handed over to the "owners" and software maintainers. These "owners" must be clearly defined, as someone (an individual, or a number of individuals) must take responsibility for the system. Someone must pay for running costs. These costs include operations (CPU time, disc and tape storage), maintenance (hardware and software), enhancements, insurance, etc. It doesn't make sense spending a great deal on development and then hoping the system will look after itself. These costs must be taken into account otherwise system degradation is inevitable.

### 3.4 Design Philosophies, Procedures, Operations and Administration

#### 3.4.1 Design Philosophies and practices

Traditional organisations follow a serial design philosophy. One person's activities are directly dependent on the completion of another person's activities. This can be compared to an "In-Basket / Out-Basket" operation. A preliminary design is placed on a drawing. This drawing circulates through all relevant design teams for comments and to give them the opportunity to adjust their part in the total design and in turn circulate this information to all interested parties. This method works (and has been done so for quite some time), but it is slow (therefore preventing many iterations) and it is prone to error. The final touches of getting a design to work are carried out in the field.

If the necessary integrated tools are available and the staff are adequately trained in their use, then multi-discipline, concurrent design is possible. Any work done by one team is immediately accessible to all other teams, who are then able to add their contribution (largely in parallel). Each team does not have to collect and collate information from all the other teams for their own purpose. The proliferation of locally stored, out-of-date information is largely eliminated. When one team wants to modify a design that could compromise another team's work, the system can detect and prevent such clashes. This enforces improved communications, as the building up of the total design is now everyone's responsibility. If sufficient facilities are available, the emergence of very tightly coordinated, close-knit, MULTI-Discipline design teams will emerge.

#### 3.4.2 Procedures and Administration

New technology (especially integrated technology) spreads over traditional functions. New interfaces and procedures need to be established and managed. An organisation must be in tune with its information systems. Procedures and policies need re-evaluation and modernisation as organisational information systems evolve to utilise more modern technology. Old paper based procedures need to be replaced with "communication protocols". Communication is the new key word (as information is generally freely available). New administrative relationships need to be established between the new teams organised to take advantage of the new integrated systems.

It takes time to implement, organise, test and establish new procedures. Management and users must accept this. The following point is emphasized once again; these new procedures will not materialise by themselves as a result of a new system being installed. They must be explicitly established through negotiation between all parties involved.

### 3.5 User Aspects

The three supporting structures of a CAE system are Management, Users and Technology. Management has been dealt with above and Technology is dealt with in other chapters below. Some factors effecting users (and non-users) of CAE systems are discussed.

#### 3.5.1 Job design

Decisions need to be made about task allocation between human and computer. Some tasks previously done by people will be taken over by the system, but people are needed to run the system and their roles will therefore change. Different skills are needed. Jobs must be re-designed to make full use of the benefits of CAE and not simply a translation from manual to computer. Human aspect must be taken into account though, to prevent bitterness about having work taken away. Retraining of replaced workers, to carry out higher skill and alternative activities, should take place before a system is introduced.

Only a fraction of a designer's time is actually spent interacting with the system. Communication about a design, meetings, design reviews, etc. are still normal activities. Information gathering still consists of a lot of person-to-person contact but a lot of information would now be accessible via the system. System bugs and software and hardware issues often take up too much of a designer's time. They tend to get too involved. A trained system administrator is needed to isolate users from these problems.

Users would experience more formalism and therefore less autonomy over how they do their jobs. It could be felt by designers that being forced to use a common tool may stifle creativity (in preference to better adherence to standards). Creativity and innovation are admirable and necessary qualities in a design team. Although, when the individuality and ego that accompanies these qualities becomes counter productive, then restrictive measures have their place. The system should be flexible enough to cater for individual design styles, but must be restricted within reasonable bounds.

The role of the Drawing Office Supervisor could be greatly affected by an integrated CAE system [MGT 10]. Traditionally, the drawing office is the custodian of all design data, whereas now, much of this design data is freely available to anyone requiring it. If the system is centralised, then power (and staff) tend to be removed from the Drawing Office and distributed to where they are better utilised. The authority to allocate work between human and computer could also be removed from the D.O. supervisor.

### 3.5.2 Skill factors

Deskilling (i.e. the replacement of skilled people with less skilled people) does not generally take place. The opposite actually tends to happen, because designers begin to do their own drafting and data control. Low skill jobs could become threatened. Much of the functionality of Drafters, tracers, data controllers, etc. who perform little more than a clerical role, would be taken over by the system and used directly by the higher skilled designers. A side effect of this elimination of lower skilled people is that older designers (familiar with paper based design) refuse to directly interact with the CAE system. This is because they see it as a "Data Controllers" job, i.e. a menial task, despite the fact that a layer of unnecessary, error prone communication is eliminated.

This is unfortunate, because CAE systems require higher skilled operators than simple database systems, yet traditional designers are expecting data controllers to operate the system on their behalf. This will not work, hence the requirement for an attitude and culture change. Experience has shown that successful implementations of CAE are operated by higher skilled people. (One benefit of higher skilled users is the improved communication between system developer and users due to the more sophisticated understanding of the higher quality users).

### 3.5.3 Differences between users and non-users

One would expect different sentiments from users and non-users, namely:

Non-users would tend to be negative about the technology. They perceive their jobs to be less motivating, needing less skill, of less significance, and having less task identity.

Users would tend to be positive. CAE seems to provide better job opportunities, new procedures are made more important than old methods, "wizkids" are placed in the lime light, etc. This can cause resentment among non-users.

Actual findings differ somewhat from the above expectations:

A general trend was an all round negative feeling because of the perception that designing is a dead end job (whether a user or not). This is interesting, because CAE could be perceived to open new doors, or could be seen as a threat that degrades their job status (already shown to be a false perception). Both old and young see the potential of CAE yet the elderly don't see the opportunities in their own personal future while the young see CAE as a very important part of their future yet other organisational factors (salary, working conditions etc.) tend to play an even bigger role. Non-users sentiments towards the technology also depend on whether they chose to be non-users or were excluded for other reasons. [MGT 02].

Job end goals of users and non-users are the same (e.g. get a design done) but job content and methodology differ dramatically.

### 3.5.4 Age factors

Older people tend to remain non-users and spend more time on supervision, high level creative/conceptual design, and support and maintenance of old designs. They spend relatively little time on the detailed design of new projects. Younger designers spend much time collecting information and are principally involved in the detailed design of new projects. Many old non-users don't see any personal benefit in learning a new "foreign" technology and are quite happy to be delegated to less important jobs. This would be the group nearing retirement.

Younger people tend to use the technology first. The average age of users is 39 while that of non-users is 48 [MGT 02]. (An even larger gap is evident in ESKOM). The chances of someone becoming a CAE user decline by 2% for each year of age (i.e. someone 10 years older would stand a 20% less chance of being a CAE user). Now projects go to users (therefore young people). Although younger people are less experienced overall, they tend to be better educated and more experienced in modern technology. Care must be taken though, to not lose the experience of older people as they are moved aside.

Young and old have different learning capabilities. The young seem to learn faster and rely a lot on memory while the elderly learn by adjusting their experience to new conditions, i.e. learn by analogy.

### 3.5.5 Job security

Generally there are no job reductions (as promised by vendors) [MGT 08]. There is often an increase in staff during transition phases and this extra staff usually remain with the organisation as benefits click into place and production is increased. The perceived job threat seldom materialises if people are prepared to be retrained for new roles. There is a fear by technicians that engineers may take over (and the same between drafters and technicians). This is a valid fear as CAE systems demand higher skilled operators. In South Africa, this "replacement" is negated by the shortage of skilled engineers and the trend to make engineers managers. The very low skilled people do face a real threat and would have to be prepared to advance themselves or face redundancy.

Non-users, and lower skilled people would be first to go in hard times. Those, in which a company has invested much in training, would be retained the longest.

### 3.5.6 Ergonomic factors

These are very often overlooked, yet can cause problems that only surface in the long term. Working conditions of "terminal bound" staff must be carefully considered to reduce health problems (e.g. back pain, eye strain, stressful chaotic surroundings, etc.) Reduction in on-the-job social communication must be compensated for by providing more non-job related social activities. [MGT 22].

### 3.6 Training

Training is absolutely vital for bringing about a culture change and to reap maximum benefits from advanced technology. This is probably one of the most important, yet most overlooked facets of CAE. Training is repeatedly mentioned throughout the "Non-technical" sections of this report. Managers, Users, Developers, Maintainers, everyone interacting with the system in any way must be adequately trained to carry out required tasks. This is even more vital in sophisticated environments such as Integrated CAE.

Whether CAE is used in front-end design or only down stream functions, affects skill requirements. CAE support of front-end design has higher skill requirements. "Operators" will tend to be well educated engineers or technicians. If CAE is only used in down-stream operations (e.g. drafting), then deskilling could take place. Drawing office staff, who previously took an active role in the design function, could become little more than CAD operators / Data controllers, requiring very few engineering skills. As was stated in "Right and wrong reasons for the introduction of CAE" (3.2.2), using advanced technology as a replacement for mental manual tasks seldom justifies the cost of the system. The further up-front that CAE is used, the bigger the benefits, and therefore training becomes even more important.

#### 3.6.1 Skills shortages

Some of the most common skills shortage problems are listed:

- Lack of general computing skills: keyboard use, data management, computer usage discipline and consistency;
- Engineering ability: CAE is not creative and innovative, but is good at data capture, analysis and "what-if" scenarios. Very advanced systems may even attempt to offer some advice in how a design may be improved, but creativity remains a human trait and the computer used only as a high speed tool. Lack of engineering ability cannot be made up for by using sophisticated tools;
- Designs can be greatly improved by iterative refinement. CAE's short design cycle time capabilities are very often not properly or fully utilised due to lack of training;
- CAE encourages rationalisation and the development of a modular product range to reduce piece-part proliferation. In general, engineers are not sufficiently aware of the logistic problems caused by individual designs. Designers generally have insufficient insight into the operation and maintenance problems of the equipment they design. Modularisation and consistency are factors that enhance understanding and maintenance of systems. Modular designing for multiple purposes is an acquired culture that requires some training and a lot of practice;
- Centralised design places an extra communication burden on designers who tend to be introverts, and interpersonal skills and improved communication tend to be lacking due to traditional islanding of functions.
- Formal training (such as at Technicons or Universities) is needed in larger quantities. This is because traditional, single discipline teams, where a trainee would undergo an on-the-job "apprenticeship", are no longer well defined. There may only be one expert per discipline who would be too busy to act as mentor to trainees. The days of master and life long pupil are (unfortunately) disappearing.

### 3.6.2 Specific skills needed

- **Computer awareness:** Simple things like typing skills, basic computer operation, what a disk is, the difference between a PC and a mainframe terminal, etc. Without such very basic skills, further training and use of the system are severely hampered.
- **System Operation:** How to logon, how to use the menus, function key operations, screen forms, etc. What to do when something goes wrong (e.g. a network dip or the database goes down). How to use the provided facilities to query data and produce reports. In other words, how to USE the system.
- **Conceptual model and design functions:** Just knowing what keys to press does not help much if one doesn't know WHY. Design functions and the conceptual model must be mapped onto system operations, menu options, screen forms, etc. This is very important if full use is to be made of the system's advanced features.
- **System Administration:** The person(s) responsible for the administration of the system must know all the required functions and procedures to keep the system running smoothly. These functions include user registration and access profile administration, database monitoring, cleaning out log tables, importing external data, etc.
- **System Maintenance:** The team taking over maintenance of the system must be fully trained in the technical aspects of the system. The procedures to be followed in carrying out modifications, re-testing and placing back into production must be strictly followed. Standards must always be correctly applied. If maintenance is not carried out in a very disciplined manner, system degradation is inevitable. Seeing that approximately 80% of a system's life time costs is consumed by maintenance [SM 06], it must be done properly, and to do this, training is necessary.
- **Training:** In order to train other people effectively, instructors themselves must be well trained. Training techniques need to be adapted to suite the skill and organisational level and personality of trainees.

### 3.6.3 Training methods

Various training methods are available. The most effective method(s) depend on the type and size of the system, the nature of the organisation and the nature of the trainees. Experience has shown that tutor based formal lessons are the best introductory training and thereafter on-the-job hands on training. Once some experience is gained in the use of the system, follow-up, advanced lessons can be presented. (It is advisable to provide a separate training system so that production data is not at risk). Users generally are only interested in learning just enough to get their job done. This is human nature and shouldn't be overlooked by an instructor. Subtle techniques need to be used to make users see the benefits of broader thinking. Other methods such as Computer Based Training (CBT), Interactive Video Instruction (IVI), the use of consultants etc. all have their place, depending on requirements.

### 3.6.4 Continuation of Training

Training needs to continue for a time equal to that of the life span of the system. New staff members and people changing roles (but still remaining as users of the system) need to be trained. It is unreasonable to expect a newcomer to simply climb in and learn the system. Continued, regular, formal training is required. Re-training may also be required after each major modification or enhancement.

## 3.7 Considerations for System Developers

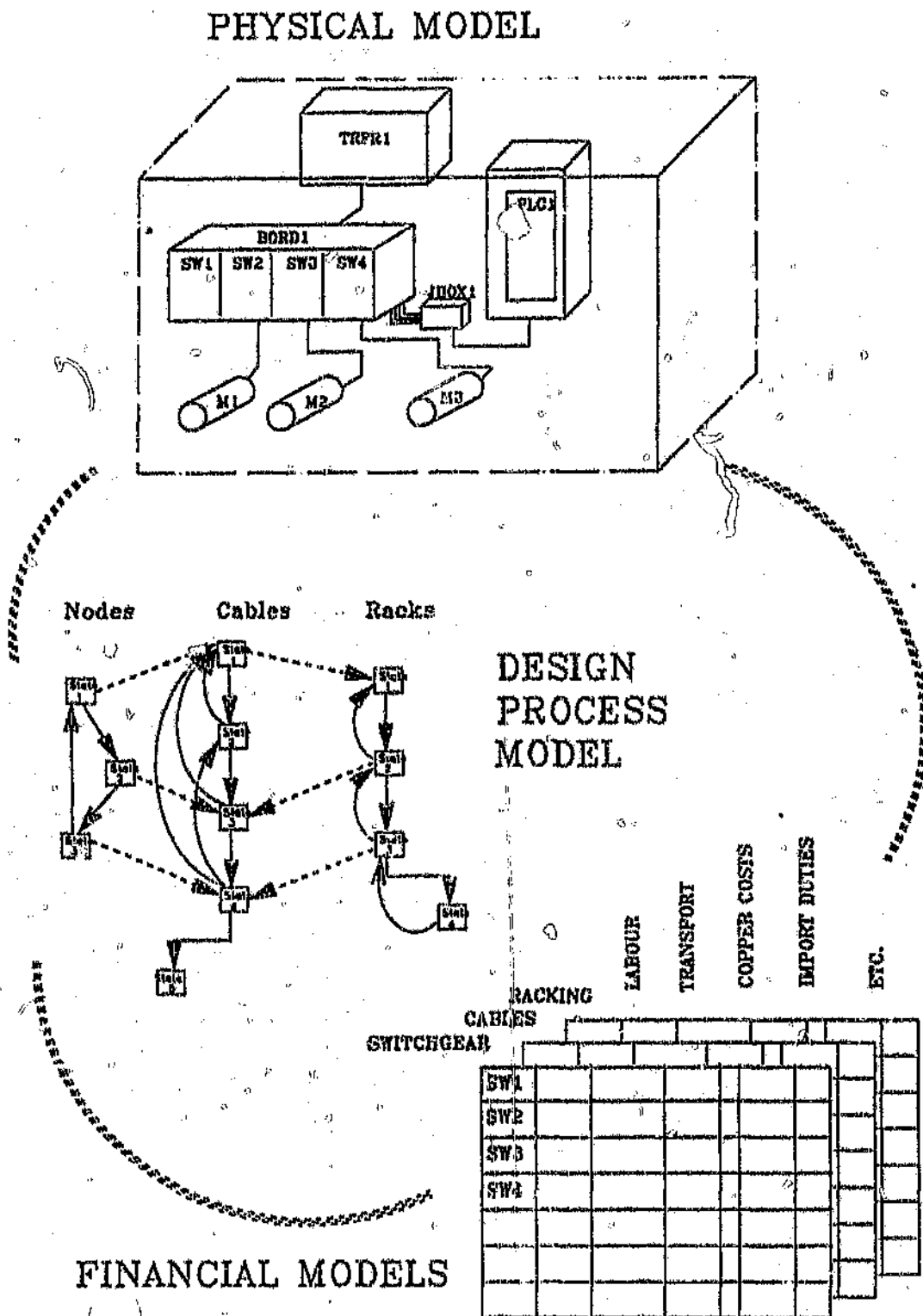
The developers of a system also play a crucial role and understanding of the above non-technical issues is a prerequisite. Generally the people allocated to establishing CAE requirements are experts in their own domains but have little knowledge of CAE technology. It is therefore difficult for them to visualise system requirements when they don't know what is possible and what is not. This is a very frustrating time for system developers because users are generally only prepared to have certain of their mundane manual tasks computerised (for fear of losing control), whereas the developer is attempting to bring about revolutionary changes requiring a major culture change (which does not happen overnight). Computer systems should be designed in a fashion which allows for gradual introduction (i.e. evolutionary, not revolutionary) otherwise rejection of the system is inevitable. As a system is gradually introduced, users begin to experience the benefits (and frustrations of missing functionality) and will, hopefully, gradually begin to alter their behaviour in favour of further advancement. Both sides must continuously make concessions in order to keep momentum.

Management on the other hand must understand the difficulty in analysing current procedures in an unstable, dynamic environment. For example, as soon as the developers have sufficient detail about certain activities, the goal posts are typically moved (sections re-organised, people moved etc.) and large chunks of system requirements are invalidated. The developers, in turn, must account for this and create sufficiently flexible systems that are based on logic and not politics (as logic is absolute, politics fickle).

Developers need to employ "tricks" to bring about the required culture change. "Carrot Dangling" is used by emphasising how a user's vested interests can be enhanced by using the system. If it can be clearly shown that a user will save time and effort at doing something that is particularly frustrating to him, new ideas will be accepted. I.e. the developers must get to know the users quite intimately. Users must also sometimes be allowed to design small parts of the system for themselves. This provides an all too important sense of ownership. Another aspect is to concentrate on young people as they are more adaptable. Identify the key decision makers and try win them over. Others will follow out of fear of upsetting their superiors.

Users often demand that a tool must not restrict or dictate activities. This is true within reason, but one of CAE's strong benefits is rationalisation. This should not be compromised to such a degree that other related benefits are lost, just to please the users.

Development should not attempt to exceed the limitations of the computer platform as this leads to compromised, difficult to maintain software. For example, it shouldn't be attempted to create fancy graphical representations on a text-based mainframe terminal.



S.101

The Physical, Financial and Design Process Models

Figure F4.1.01



## 4 System Models

### 4.1 Introduction

Any information system must be based on some or other "model". [MDL 25,32] For example, an Accounting systems may reflect the Dual-entry Journaling system and a Salary package could be based on a "Salary Profile Model".

Design systems are best based on models that reflect the equipment or system being designed. If the designer can work with the computer tool as though he was working with a model of the real system, then usage of the tool is greatly enhanced. This is because the tool provides the necessary facilities for the designer to think at an optimum conceptual level and not have to keep translating conceptual thought to a lower level suitable to the tool. Consider an electrical system design package for example. A designer thinks in terms of transformers, cables, switchgear and motors. The system must therefore enable the designer to do just that. He should not have to think in terms of database records or lines and curves. The model may well be implemented using such primitive concepts, but the user interface should shield such implementation details from the designer and present him with a consistent, usable interface which facilitates thinking at the correct conceptual level.

Modelling real world situations is difficult because the model chosen is very dependent on the point of view taken. For example: a designer is interested in voltage and current levels, a site engineer may only want to know which contractor is doing what work, a project accountant may see the system as a series of related cost-codes, planners may only be interested in what is done and when it is being done, and not how or why. The approach taken in developing CEEDS was:

- Firstly the system is a Design Aid;
- Secondly a Construction monitoring tool; and
- Thirdly a Contract monitoring tool (Refer to "Financial Models" (4.4) below).

This order of priority has obviously effected the point of view from which the models have emerged. The driving force was to create a suite of models suitable for the design and construction of electrical systems. They are based on the real-life way in which electrical systems are made up. The PHYSICAL Model is the main model and forms the foundation for all subsequent models. In order to control the design process, a DESIGN PROCESS model is proposed. This model would control the sequence and validity of operation being carried out on the Physical model. The costs of the physical equipment contained in the Physical model can be determined by having a "Cost Reference" into an appropriate FINANCIAL Model. Each of these models are discussed in detail below.

Refer to Figure 4.1.01.

- (1) Generalised CAD systems are not optimal design tools because the designer must think in terms of primitive concepts such as lines, curves and surfaces and not in terms of the physical objects with which he is working. The "representation" presented by such a CAD system carries very little inherent information and therefore analysis of the design represented is correspondingly difficult.

## 4.2 Physical Model

The physical model developed for CEEDS was also influenced by certain organisational factors. It must be emphasised that the main thrust of the system was to model power distribution equipment. But the model applies equally well to process control equipment. For example, with power distribution, the "Switchgear Board" is a main component, whereas with C&I equipment, a PLC Cabinet may be the main component. The majority of functionality currently provided favors the power side. A reason for this bias is that the majority of Electrical Design is done in-house whereas the majority of Process Control Design is done by external contractors. One of CEEDS's main functions is the management of power cables and most process control cables (therefore accounting for the well developed Cable Design Functions and the conspicuous absence of other design aids).

Another influencing factor in Model design is the dependency on currently used technology. For example: Switchgear using electro-mechanical control has very different specification attributes to a switchgear using electronic control technology. This effects the model details though, and not the principles.

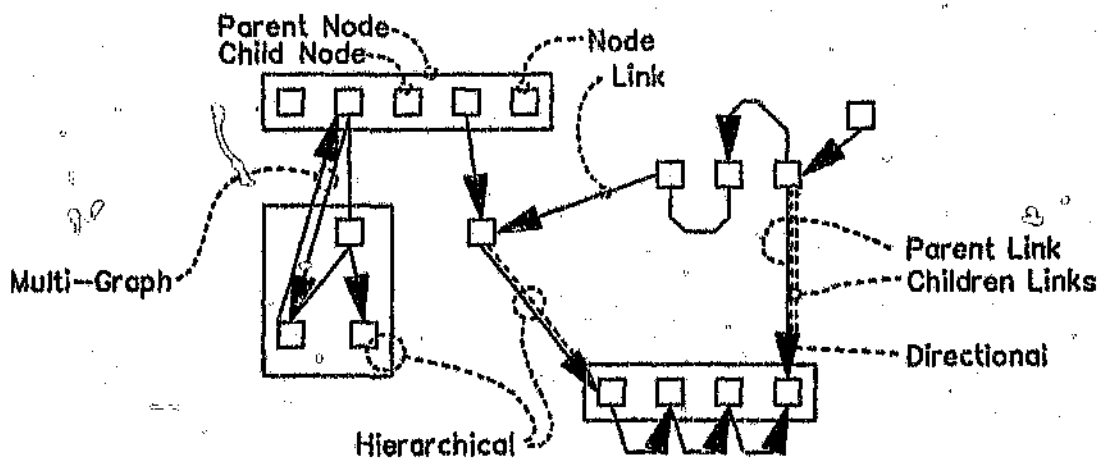
### 4.2.1 Graph based Node-Link Model

The conceptual model chosen is a "Hierarchical, Directional Multi-Graph based Node-Link Network".

- **Graph:** The data structure is a graph structure (as opposed to the simpler tree structure for example) consisting of Nodes (Vertices) and Links(edges). A Graph structure allows any node to be linked to any other node and thereby creating a "Network". A link may only be coupled to one node at either end.
- **Node:** Any piece of equipment that can be visualised as a "box" and has cables connected to it, is modelled as a node.
- **Link:** The cables connecting Nodes.
- **Multi-Graph:** More than one link can exist between any two nodes.
- **Hierarchical:** A number of nodes may be "allocated" to one node, or one node can be seen as consisting of a smaller "Network". Links can also be hierarchical in that a single conceptual link may exist between two nodes but it may consist of a number of physical links (i.e. parallel cables).
- **Directional:** Links can carry directional information. E.g. the normal direction of power flow or signal flow.

Figure F4.2.01 demonstrates the graph concept and Figure F4.2.02. shows how the model is derived from the physical situation to a conceptual representation to a data structure representation.

An example of where multiple nodes are allocated to another node is a Distribution Board (one node) that has many switchgears (each a node) "allocated" to it. Refer to Appendix A4.3 to see how such allocations could be interpreted physically. A link is hierarchical when multiple cables are used for a common purpose, e.g. three single core cables, each with a unique identifier (cable number), are required to conduct a given current. There are three separate physical cables but functionally they are only one link.



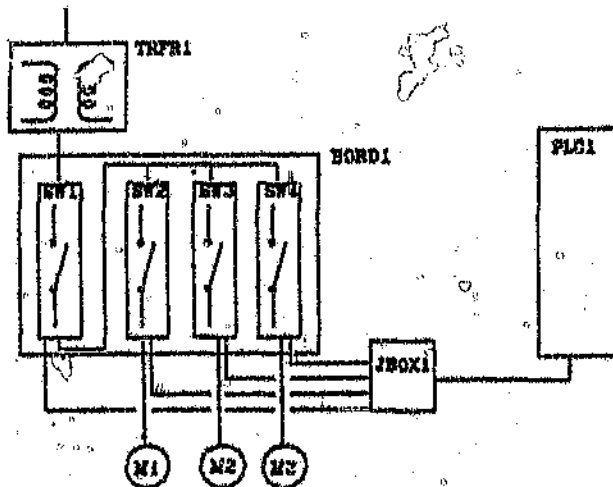
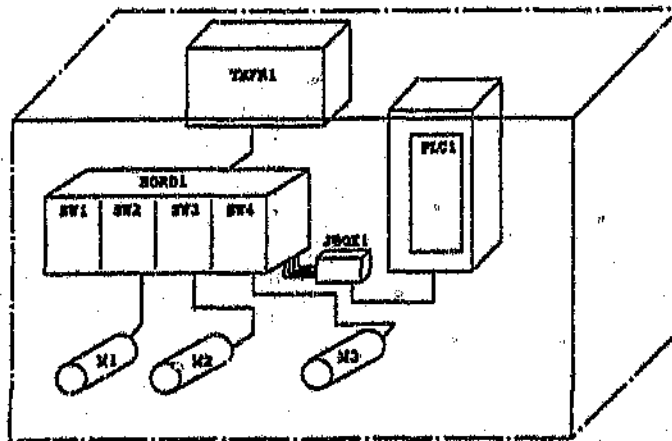
Hierarchical, Directional Multi-Graph Node-Link Network

Figure F4.2.01

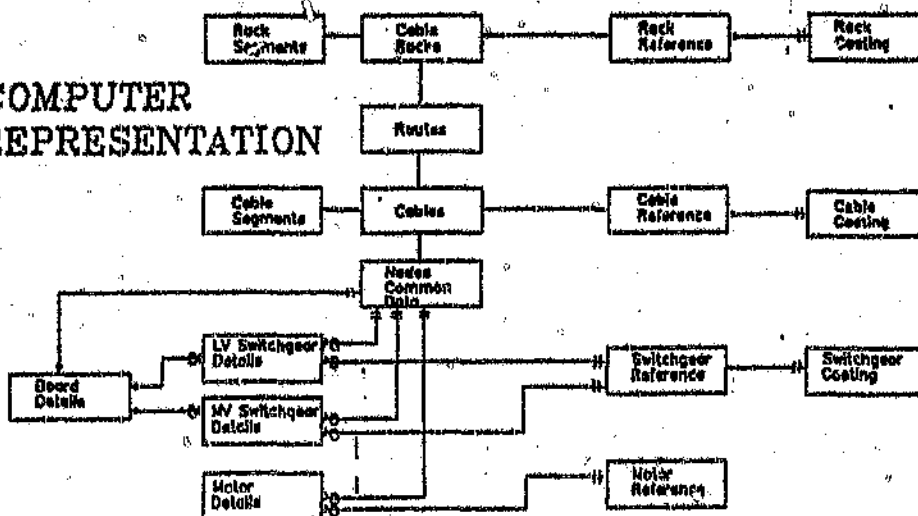
Any operation involving the "link" must be carried out on all members of the link. (For example, all cables belonging to the link should be routed along the same route and not split over various alternative routes). This is achieved by having a "Parent Cable" with "Children Cable(s)" belonging to the parent.

The Level of detail that the model must manage can also vary, and there are various techniques that can be employed for these levels. For example, an electrical network can be visualised having a Macro, Mid-range and Micro connectivity.

- At a macro level, there isn't even the need for actual links. One of the attributes of a switchgear for example would be a "Feed Code". This would indicate the end user of the power controlled by the switchgear. Any nodes in-between the switchgear and the end user (such as junction boxes, transformers, speed controllers, etc.) are ignored. Ring feed systems and alternative paths are ignored by assuming power flow in only one direction (the default direction would be chosen). This results in a tree representation of the distribution network. See Figure F4.2.03. There are uses for this level of modeling, especially for initial design where unnecessary detail clutters and confuses the issue. For example detailed cabling need not be known in order to work out a suitable distribution system or in deciding which loads to allocate to which boards. A representation at this level would resemble a "Single line" diagram;
- At the "Mid-range" level, all nodes and all cables (links) would be modelled. This level is very good at representing connectivity between equipment. A representation at this level would resemble "Cable block" diagrams;
- At the Micro level, individual cores within multi-core cables, and the Terminal strips within Nodes would be modelled. This provides sufficient detail to carry out construction and commissioning of any modelled equipment. A representation at this level would resemble "Termination" Diagrams and detailed schematic diagrams. Modelling down to termination level makes it possible to provide automated support for the design of equipment such as junction boxes, distribution frames, splitter boxes, etc. Refer to Figure F4.2.04.

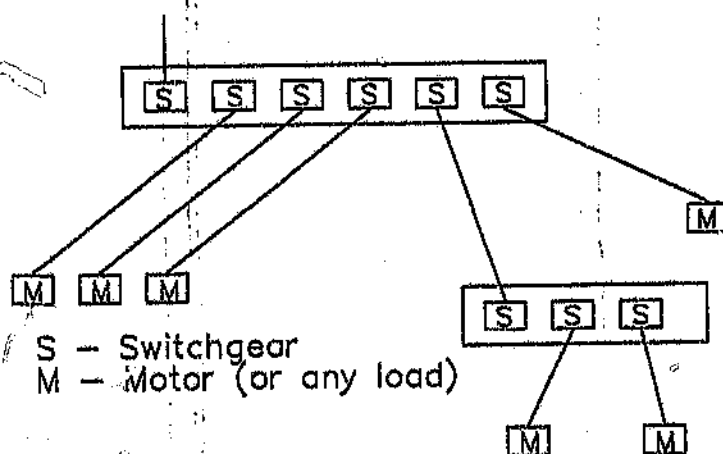


### COMPUTER REPRESENTATION



From Physical to Conceptual to Representation

Figure F4.2.02



## Tree Representation

Figure F4.2.03

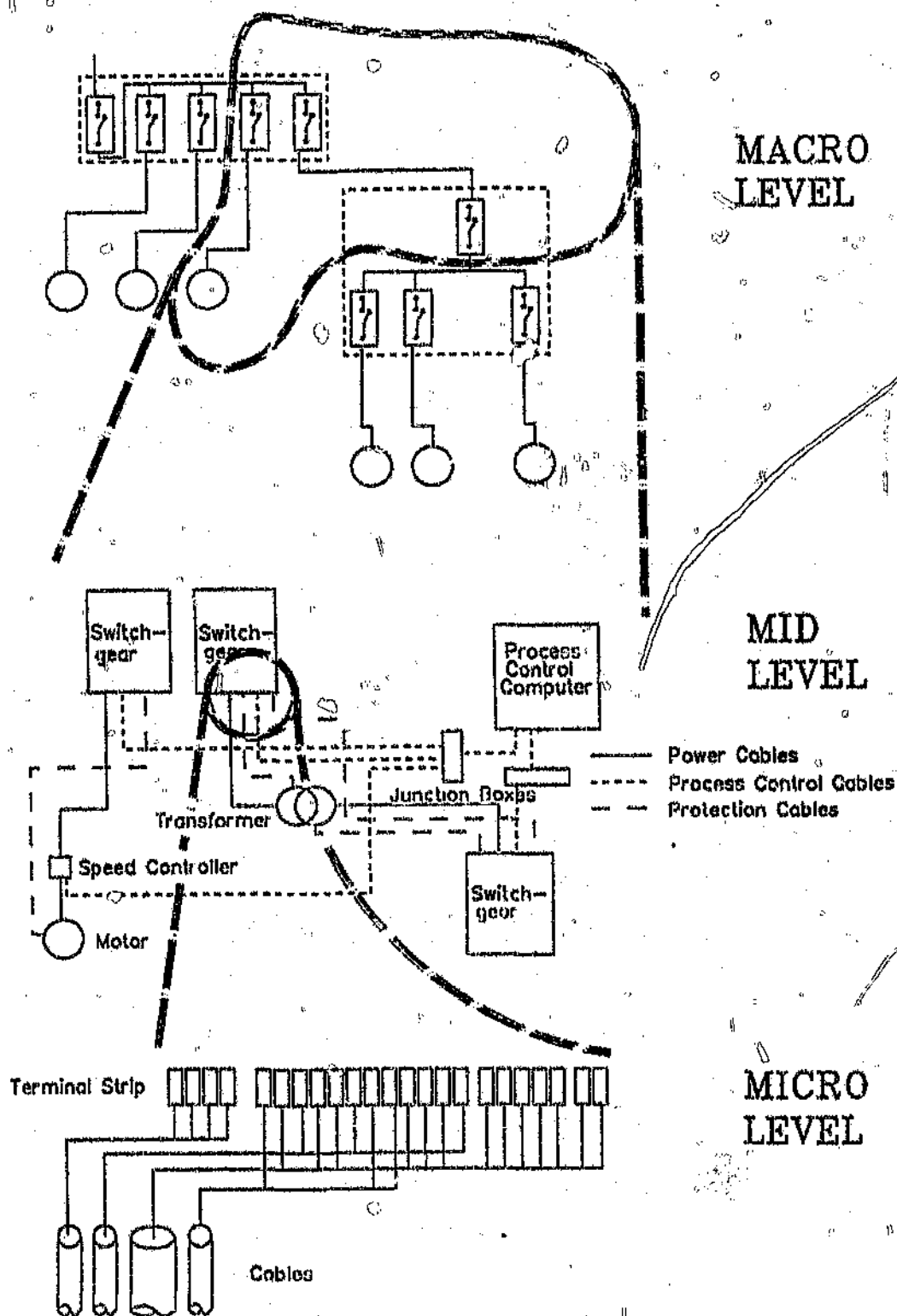
Facilities should exist whereby the integrity of these three levels is guaranteed. For example, macro level connections cannot be changed once cables are coupled, and the cable path (at the mid-level) must not conflict with the Macro connection. It is proposed to guarantee this integrity by controlling the order in which actions may take place. (Refer to the "Design Process Model" (4.3) below). Using the representation schemes described below, a graphical query could be set up whereby multiple levels are represented simultaneously and visual verification can take place.

### 4.2.2 Systems, Identifiers, Equipment Classes and Types

Each Object (Node or Link) within the model must be able to be uniquely identified. This is achieved by giving each object a unique "Object Identification Code". (i.e. every INSTANCE of a node or link is coded to indicate its place in the total system)<sup>(1)</sup>. The underlying database management system ensures uniqueness. Along with this code a natural language description is also given to enable human interpretation of what the code is.

Each object is also given an "Equipment Class", and within that Class, a Type can be specified. See Figure F4.2.05 for an analogy between a node and a person. The Object ID Code (and attached description) describe an instance of a node and is unique while the Class and Type give details about the node itself which are independent of the situation in which the node finds itself.

(1) An international standard coding system has been developed for use on power stations. It is known as the Kraftwerk - Kennzeichensystem (KKS) Coding system. This coding system has been adopted by ESKOM for use on all new projects. The identification code used on CEEDS is code independent to allow for non-KKS "System", "Pseudo", and Dummy nodes/links and to allow the system to be used for older Non-KKS coded stations. Verification of the correctness of codes is achieved by a specific Coding system batch mode verification facility (Refer to "Codification" (5.3.3) under "Design Functions" (5) below).



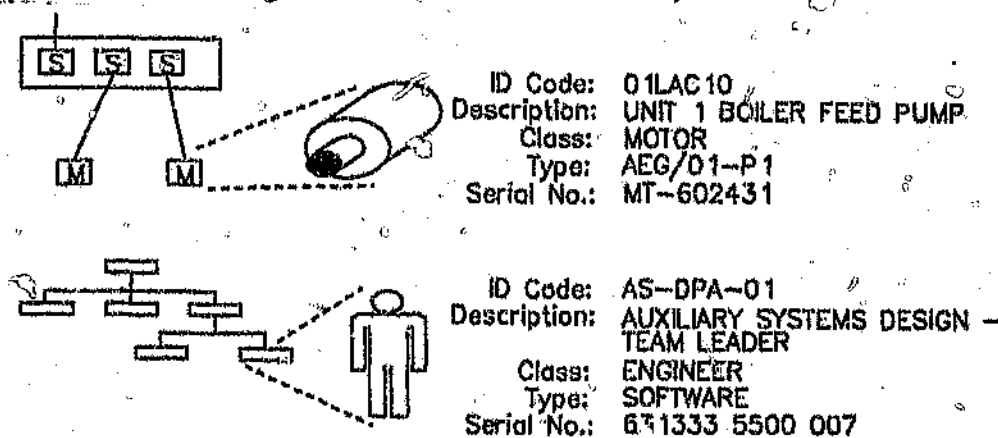
Macro, Mid and Micro Levels of Detail

Figure F4.2.04

For example, a motor of given Type can be used in multiple locations, but each location is unique. There is one other identifier which doesn't play a part during design but is used during maintenance. This is the "Stock Item" number, or Serial number of a given physical piece of equipment. A motor Node Identifier (for example) identifies the node's position and function within a system and does not depend on which specific instance of motor is actually installed during construction.

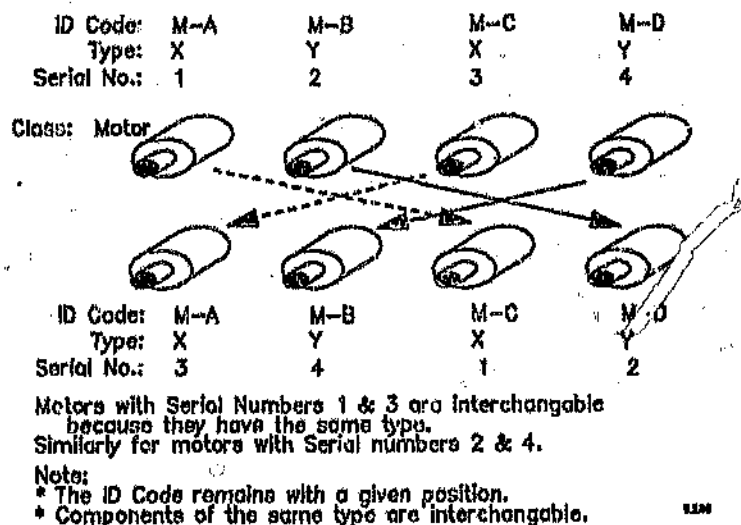
If this motor is removed and replaced with another motor of the same type, the removed motor retains its own "Stock Item Number" but not its Node Identification Number. The new motor (with its own Stock Item Number) inherits the Node Identification number from the removed motor. Both the old and the new motors should have the same Type Code. See Appendix A4.1 for details on the KKS Coding system and on the proposed Class/Type coding system.

See Figure F4.2.05 for an illustrative example.



## Coding Analogy.

Figure F4.2.05



## Coding Example.

Figure 4.2.06

#### 4.2.2.1 Type Reference Tables

Two of an object's attributes are Class and Type (as mentioned above). These two combined could be used to provide a unique reference into "Reference Data". "Type Reference" tables are used to store all data relating to a given Type of equipment. Separate tables are used for each Class of equipment. The data contained in these Type reference tables are those attributes which remain fixed for a given piece of equipment. Those attributes which could vary from instance to instance of an equipment, whose usage are attached to the specific Node (or Cable). By giving an object a Type, all of the attributes contained in the reference table can be "inherited" by that object, and need not be explicitly stored with the object. Some attributes in the reference tables may be "Default" values. These could then either be inherited by an object or overwritten in a duplicate attribute attached to the object. (Refer to "Detail of attributes of various nodes and links" (4.2.4) below).

The attributes contained in a reference table would depend strongly on the Class of equipment whose types are stored in the table. These attributes could include:

- All fixed technical data. For example, a motor has certain properties, physical sizes, connection types, etc. (variable data, such as expected load (not rated load), for a specific node, is attached to the node);
- If the computer environment is suitable, graphical data such as torque/speed curves and base plate layout could be coupled to the reference data;
- Default data (as mentioned above);
- Cost reference(s) into various financial models. One Cost Reference Code could refer to a Financial model that would indicate capital expenditure (contractual) cost for construction purposes. Other Cost Reference Codes could refer to models that indicate spares purchase costs for maintenance purposes. Certain other costs could be directly coupled to a type and not referred to via a financial model. This depends on how complex the cost determination of a type of equipment is. (Refer to "Financial Models" (4.4) below);
- Expected labor time (and costs) to replace, repair etc. for Trade union and work load purposes;
- and there could be many more.

This Reference Information may be common to Design / Construction systems (e.g. CEEDS) and Maintenance systems (e.g. PERMAC). If such Reference Information is available to both systems, the design system need only supply the maintenance system with minimal information (such as Object Identifiers and Reference Codes) because all other data would be available in these centralised facilities.



There are some complications in the use of this reference data. These include:

- Various projects may use different subsets of this reference data. (e.g. the Majuba project may use a different set of cables to the Kendal project);
- The cost references for a given equipment type may be different for the various projects. (e.g. even though both Majuba and Kendal use a certain type of cable, the cost of cables for Majuba may be higher than that of cables for Kendal because of higher transport costs for instance).

Possible solutions include:

- Using separate reference facilities for each project. This is the simplest method but could lead to relatively large scale duplication of data; or
- "Project codes" could be incorporated into the centralised reference system to allow access to project specific attributes; or
- Separated "Look-up" tables could be created and maintained for each project with links into the main reference tables.

For current reference facilities and for details on how switchgear are currently specified, refer to Appendix A4.4.

#### 4.2.2.2 Systems

Various information contained in the model can be grouped into "Systems" as seen from different points of view. Some groupings already identified include:

- **Grouped by Contract:**  
All of the equipment that is to be supplied under a given contract is grouped together. (A document that is produced along these lines is called the "Contractors Electrical Equipment Listing"). This is achieved by allocating to every object, a "Contract Code". This is a link into a series of Contract/Contractors reference facilities containing all relevant information about the Contracts and the Contractors);
- **Grouped by "Work Package (activity)" in a Project Master Plan:**  
For construction purposes, one must be able to determine which equipment is installed and/or constructed, and in what sequence. The total construction is broken down into work packages (which are further broken down in detailed activities). These "work packages" are given a code. Each object in the physical model is then allocated such a code. Refer to the "Design Process Model" (4.3) for details;
- **Grouped by physical functionality:**  
For example: Turbine lubrication oil system, Scott blower system, Mill system etc. The KKS coding system used as identification for objects within the model has an inherent functional breakdown. Refer to Appendix A4.1 for details;
- **Grouped by physical location:**  
Each area/volume within a "building" is given a "Structure Code". Each object in the model is allocated to the area in which it would be located by attaching such a Structure Code to the object. In this way, lists of

equipment found in a given physical location can be produced. (The structure code is part of the KKS Coding system. Refer to "Extensions to the model" (4.2.7) below and to Appendix A4.1 for more details on Structure Codes).

With the exception of the Contract Code, all of the coding systems used in the groupings above are hierarchical in nature. (Refer to Appendix 4.1 for an explanation of how a hierarchical code works). A grouping can be zoomed into or out of by specifying a more or less specific search criteria. For example: a complete building may have the code of "11B", a large hall in that building has the code "11BAU" and a given floor area in that hall has the code "11BUA01". One node may have a structure code of "11BUA01", another a structure code of "11BUA02" and yet another a code of "11BAC03". All three would be included in an "11B" grouping, only two in an "11BUA" grouping and only one in an "11BUA01" grouping. An example of the use of the functional grouping is how a distribution board can be "imploded" to represent the combined loads that it feeds. (Refer to "Design Functions" below).

#### 4.2.3 Representational Schemes

On a conceptual level (i.e. human thought level) a "model" representing a real world situation can be visualised. There are any number of ways in which individuals could visualise such a model. Some visualisation schemes are identified and discussed. The computer on the other hand must be able to store, manage and manipulate this "model". A computer does not have the visualisation capabilities of a human and consequently a far more well defined and rigidly structured representation scheme must be utilised. Various schemes are briefly investigated and the scheme on which CEEDS is based is discussed in a bit more detail.

##### 4.2.3.1 Conceptual Representation

The mental "picture" that a person would build for himself of this "model" is highly individual and very dependent on that individual's experience and point of view. A software engineer could visualise it as a series of icons joined by lines, an electrical engineer may see it as electrical equipment joined via cables in the form of a schematic diagram, a mechanical or civil engineer may visualise it in three dimensional space as a series of "boxes" occupying space and the cables as lying on cable racks. In order for a CAE tool to be as effective as possible, the representation schemes offered to the user should mimic his own internal mental picture as closely as possible. Three categories of representation have been identified:

- Textual;
- Two Dimensional (2D) Schematic (Symbolic);
- Three Dimensional (3D) Layout (Solid and/or Symbolic).

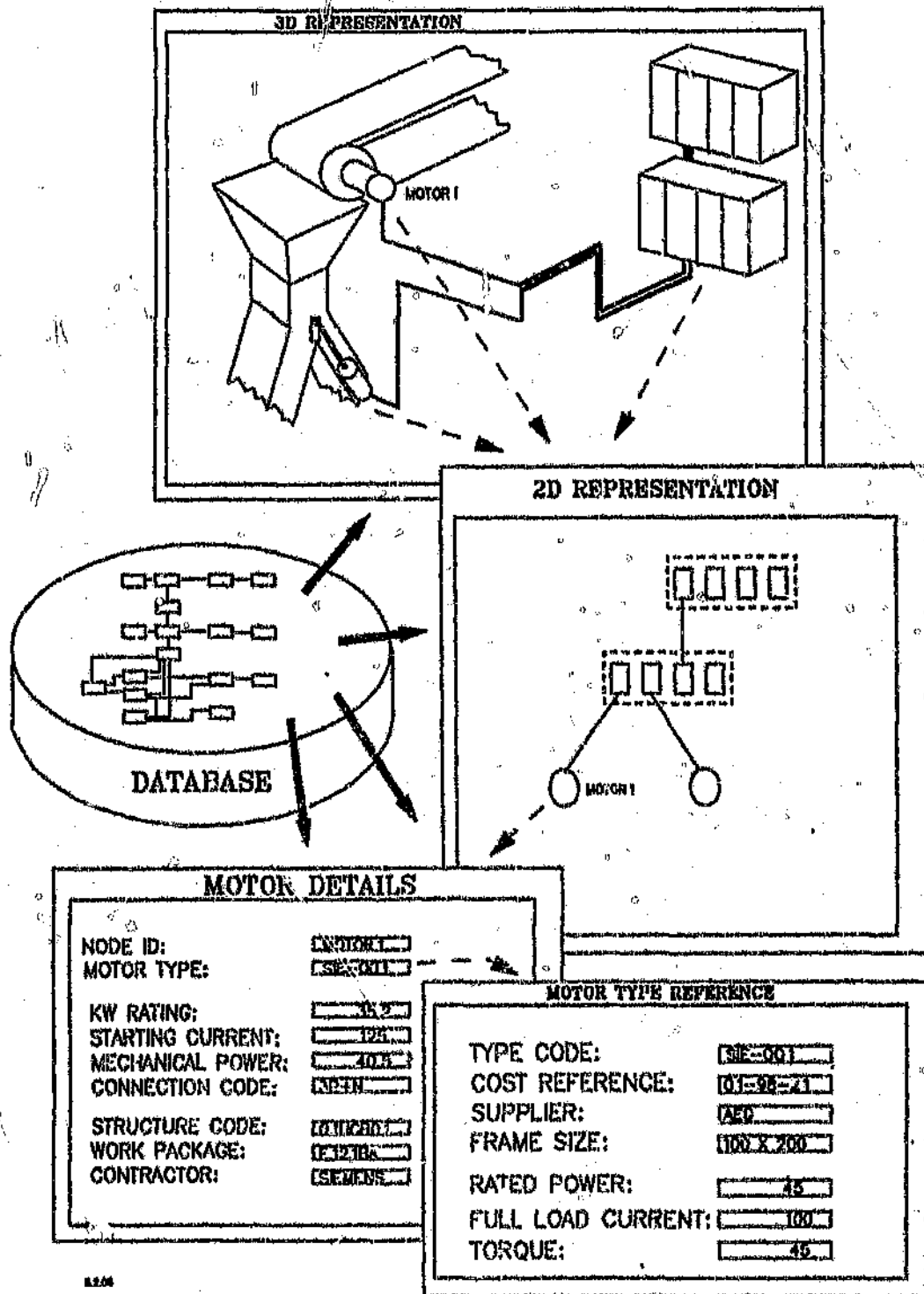
Each category has its place in a system as they present varying degrees of detail and the nature and characteristics are different in each. For example, a 3D layout would probably not be used to calculate board loading, whereas a 2D schematic representation could serve the purpose and details of a component can only be viewed through a text "screen", etc.

A very important concept to realise here is that these representation schemes are only mechanisms to view the model. All information is still stored in the model. The representation carries no information of its own (except if data is added to the representation and a report produced, refer to "Textual and graphical representations" (5.5.5) and "Configuration management philosophy" (5.5.3.2) below). All three schemes should be fully accessible from within any other one. I.e. if a set of equipment is being visualised as a 2D schematic, the user should be able to point to one or more objects and obtain textual data or a 3D representation. Similarly from within text and 3D representations. Multiple representations should be able to be viewed simultaneously. For example, if a text "window" describing the attributes of an object is being displayed, a further text view of the reference data attached to the equipment type should also be able to be viewed if so desired. Refer to Figure F4.2.08 for an example. [SC 07]

Modifications to the model via any one representation scheme should be evident to all other representations because these representations are not explicitly stored, they are created from the model. For example, if a node is added via text facilities, it would appear in both the 2D and 3D representations (if it falls within the selection criteria used to generate a representation view). Similarly for all other schemes.

The exact display would depend on how the representation is generated. There should be a number of 2D display algorithms available to produce various formats of representations. [MDL 00,33,34,37,39,40] Three dimensional representations can be created by using the actual X,Y,Z coordinates of equipment nodes. All nodes selected for viewing within a 3D representation, that have no X,Y,Z coordinates would lie on top of each other at one location (for instance, a default of 0,0,0 can be used). In this way non-digitized nodes can be detected. Text screens are the simplest format in that either pre-specified screen forms are presented, or flexible query facilities are used. Refer to "Design Functions" (5.3) below for details of representation modification facilities, (e.g. zoom, pan, isometric etc.) and for model manipulation functions (e.g. create, move, delete, etc). Display attributes can be used extremely effectively to aid the design process. Colour, intensity, blinking, line styles etc. could be used to depict and aid visual identification of any number of situations. For example, a rack design could be displayed in 3D in which colour is used to depict the different types of rack (MV racks are blue and LV racks are red for instance). In the same representation, colour could depict rack loading (green 50%, yellow 50% to 80%, red 80% to 100%, white 100% for instance).

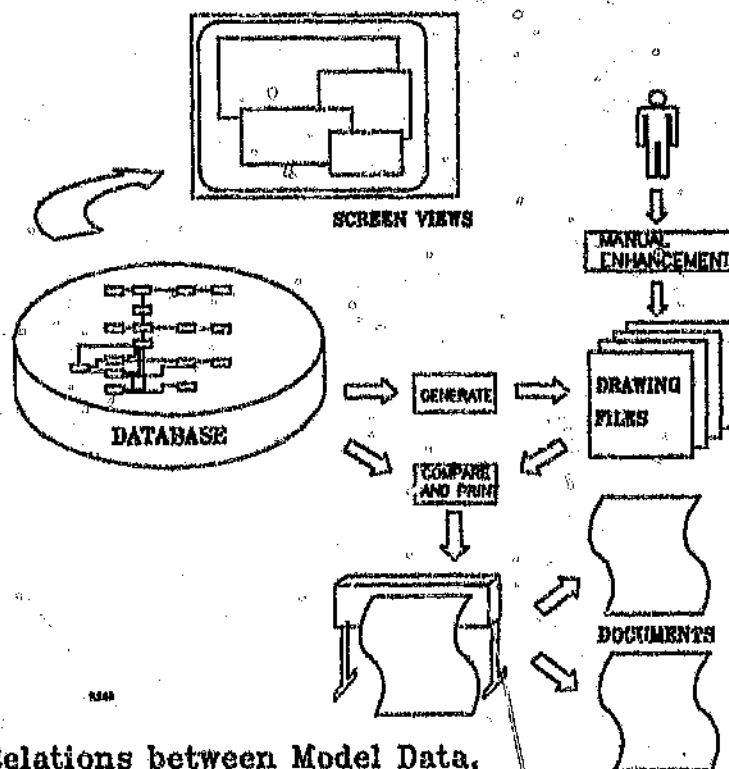
How these representation can be created and how they relate to other graphical representations must be investigated. Firstly, the graphical representations which are generated for viewing during design, tend to be simple icon based representations and not highly detailed. This is because they are usually created to portray desired data only. They are simply views of the model and don't exist in their own right. They are also totally unsuitable for production documentation (e.g. schematic drawings for construction purposes).



Relations between the various  
Screen Windows (3D, 2D and Text)  
and the underlying models

Figure F4.2.08

It is currently visualised that a set of Production documents containing a fixed graphical representation will exist in parallel with the model. These documents would initially be created automatically, but are then manually re/arranged for aesthetic purposes. These documents are not totally independent from the system in that objects depicted on them must exist in the model before they can appear on a document. One problem is that an object can be deleted from the model and others added without first consulting all attached documentation. It is possible therefore for a production representation to "get out of synchronization" with the model. It will be possible though, to verify such documents against the model for integrity, before finally producing a hard copy. The verification routine could be included as part of the hard copy production facility to prevent incorrect data being represented on a hard copy. It may sometimes be desirable to have out of date information on a document because the document might represent the state of affairs at a given point in time and a hard copy of this may only be produced at a later stage. In the meantime changes may have taken place that cause current data to contradict that depicted on the hard copy. This situation should not arise in a well coordinated team, but real life must be accounted for. Once a hard copy is produced for distribution, the actual document can be revision controlled and archived (in paper as well as electronic media). Refer to Figure F4.2.09, for a depiction of the relationship between model views and documentation being produced from the model. Refer to "Textual and graphical representations" (5.5.5) and "Configuration Management philosophy" (5.5.3.2) for details on this revision control.



Relations between Model Data,  
Views of the Model and Documents.

Figure F4.2.09

One last aspect of model representation is the link to a 3D plant design model. The ultimate situation is one in which the Civil Structure Model, the Mechanical Plant Design Model and the Electrical Model are one and the same thing. For example, if a motor is identified as being needed on a conveyor system, as soon as it is created on the Plant Design CAE system, it is available to the Electrical CAE system. Similarly for the Civil Structure CAE system, as soon as a beam is designed or a room designed, these structures are available for the mechanical engineers to begin their plant design and the electrical engineers can begin their Cable Racking Design. The external control of such a system is an issue that needs to be negotiated in great detail between all parties involved. Agreements must be made between relevant parties as to when data from one design activity is stable enough to be used by another. Although the CAE system will enable continuous incremental adjustments to the underlying models, people need non-changing base information with which to work. The synchronizing of milestones and base dates will be a planning and management function. Refer to "Non-technical Issues" (3) above.

Currently the Electrical system (CEEDS) is a stand alone system into which data is imported from Mechanical and Civil sources (in electronic and paper based formats). This data is used to create local 3D models for electrical purposes (such as Rack Design and positioning of equipment in 3D space). To ensure that the local model remains up to date with the civil and mechanical information is an arduous manual task. The Electrical Model (on CEEDS) and the 3D model (on the CAD system) are also separate so 3D information is imported into the electrical model from the CAD for cable routing purposes. This situation should be changing soon with a change in computer environment (Refer to "Computer Environment" (6.2) below).

#### 4.2.3.2 Physical Database Structures

The physical database structure depends on the software being used to store the model structure and information. There are a number of standard, well developed Database schemes available as commercial products and a number of less well developed schemes available as "Artificial Intelligence Knowledge Representation" schemes. [MDL 43]

Classical Hierarchical and Network Database schemes were not even considered because the more modern Relational Database scheme is far superior. [MDL 41,42]. Other even more modern schemes such as Object Oriented (OO) Databases and Frame Based Knowledge representation schemes were investigated from a theoretical point of view and both are firstly, very similar in concept and secondly, are very strong contenders to replace the current scheme. [MDL 17]. There are other techniques that attempt to use the relational scheme, but in a non-normalised manner, to give it some of the capabilities of a Frame Based Scheme. [MDL 18]. It is felt that a fully fledged Frame Based System should be used if such functionality is required, rather than "bending" a less suitable scheme. This practice of "bending" leads to the model manipulation functions having to be more complex as some knowledge of the "bent" scheme is required by each function. This in turn leads to longer development times and an increased maintenance burden that is seldom justified by the added functionality of the "bent" scheme. Technically, OO databases would be better suited for the requirements of the models described here but the choice of scheme used was not based upon which one would be technically the best for the job, but on the availability of a suitable commercial product.

The ORACLE Relational Database Management System was chosen for various technical and organisational reasons. (The next release of ORACLE, namely version 7, has numerous OO features, so the evolutionary move from relational to OO should be quite smooth if the same product is used). The advantage of having OO features at the database level is that certain functionality that actually belongs with the data can be incorporated in the database. Whereas the relational model, that only allows data storage (and not procedure storage) forces all the data modification routines to contain such functionality. (Refer to "Logging Facilities" (5.6.2) below for examples of such functionality).

CEEDS uses the relational scheme for model storage. The current scheme is not optimal and improvements following this investigation are tabled for the next major overhaul. The current principles upon which the Database Structure is based are:

- Nodes, Cables and Racks are modelled separately and slightly differently. They are all "objects" but Nodes consist of many Classes whereas Cables and Racks are already each a "Class";
- Common attributes of nodes are stored in one table, with separate tables for the detailed attributes of each class (with the exception of the miscellaneous class which contains only common data);
- Separate Sets of Reference tables for each Class, (Refer to "Type Reference Tables" (4.2.2.1) above);
- Financial Model Tables, (Refer to "Financial Models" (4.4) below);
- Various "Dictionary Tables" containing Coding reference information are separate from the main model, but verification takes place against these tables, (Refer to "Design Verification Checks and Model analysis" (5.3.2));
- Contractual Documents Tables, (Refer to "Contractual Documents" (5.5.3)); and
- Various "System" tables:
  - System's Software configuration, (Refer to "Software Configuration Control" (5.4.3) below);
  - "User Interface" structure, (Refer to "Man Machine Interface(MMI)" (5.2) below);
  - "Standard Reports" structure, (Refer to "Standard Reports" (5.5.2) below);
  - "Model Structure" Data. (Refer to Appendix A5.1 for an example of "Model Structure" Data).

Refer to Figure F4.2.10A & B for a graphical representation of the relational model and to Appendix A4.5 for a detailed description of the current CEEDS Database Structure.

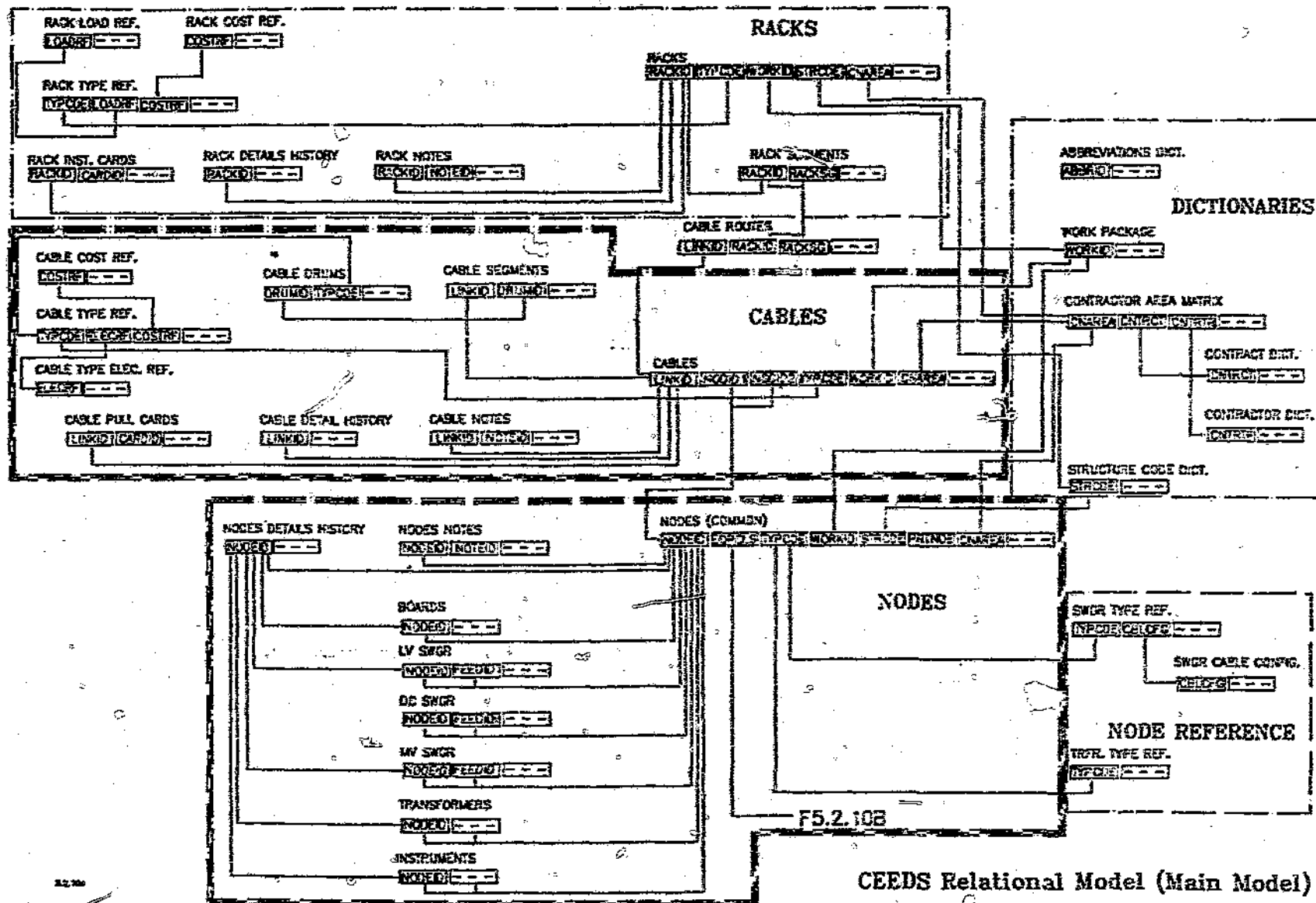
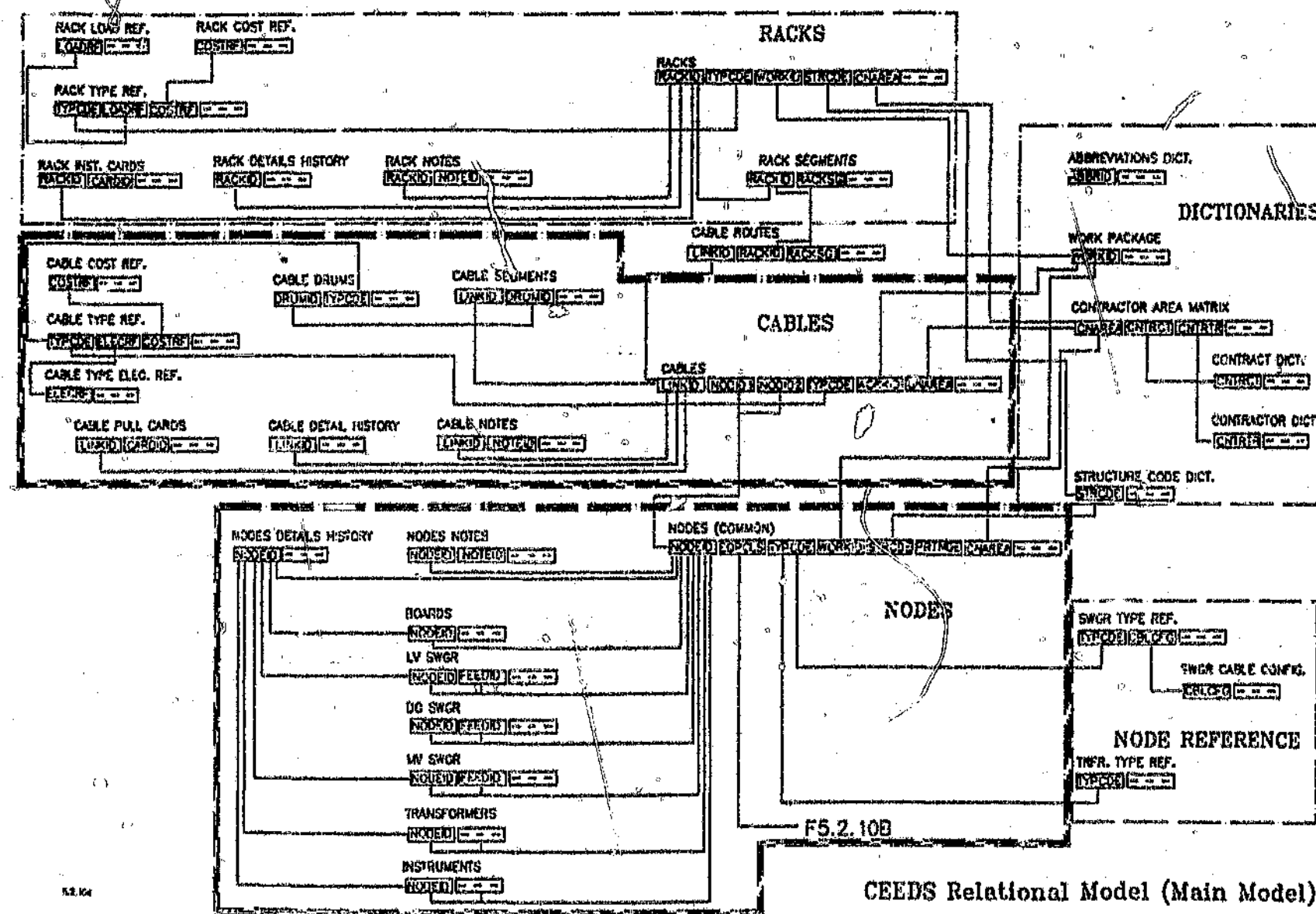


Figure F4.2.10A

CEEDS Relational Model (Main Model)

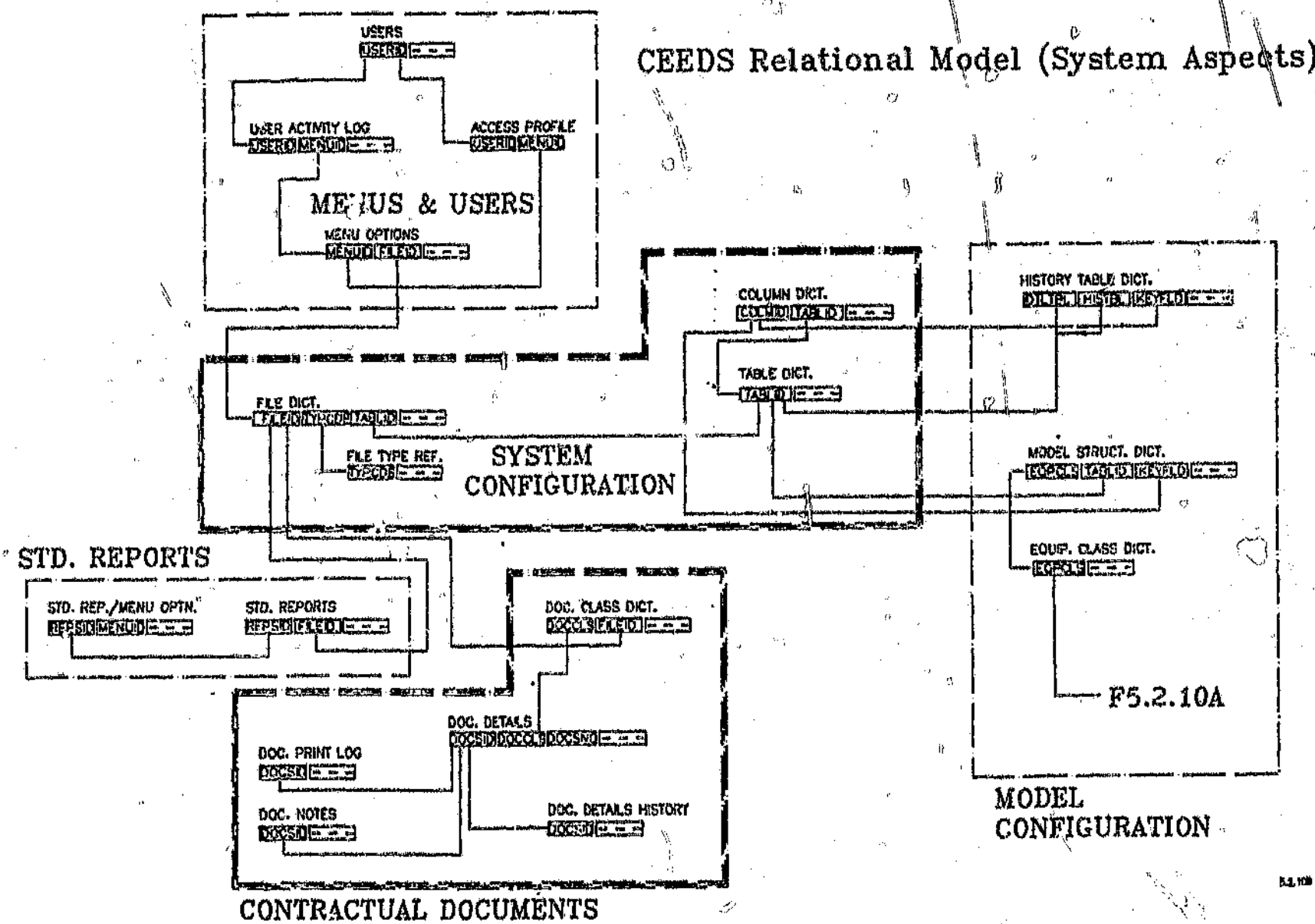




CEEDS Relational Model (Main Model)

Figure F4.2.10A

Figure F4.2.10B



The exact storage of data on disk is a detail which is left up to the database management system, but those involved with system tuning may need to understand some of these details in order to gain optimum performance from the database.

#### 4.2.4 Detail of attributes of various nodes and links

There are certain categories of attributes defined for any object. There are attributes that are:

- Common to all objects;
- Specific to a given class of object (i.e. Details); and
- Reference attributes inherited from the reference tables (dealt with above under "Type Reference Tables" (4.2.2.1)).

Nodes and cables are handled slightly differently from each other. This is because there is essentially only one Class of link, namely Cables whereas there are many Classes of Nodes (e.g. motors, valves, switchgear, process computers, junction boxes, batteries and many more). The principles though, are equally applicable to both nodes and links, as Objects.

##### 4.2.4.1 Nodes

Common Node Attributes include:

ID Code and Description(s);

Class and Type;

- All attributes which define a system (Refer to "Systems" (4.2.2.2) above);
- Common Administrative attributes such as various statuses, created and last updated dates, source of data indication, remarks, etc.;
- Physical attributes such as X,Y,Z coordinates of a node's location as described by the 3D aspect of the model;
- Common model attributes such as "Parent Node", for example, the Board to which a Switchgear is allocated, is its Parent Node;
- Power requirements such as Full Load current, Power Rating, etc. (These attributes are currently regarded as common attributes, but, considering complications listed below in "Subtleties of Power Consuming Nodes" (4.2.6), it is proposed to model them as details belonging to those Classes that have such attributes;
- Construction feedback data such as date commissioned and handed-over;
- Notes. Each object can have any number of free format text notes attached. These are mainly there to provide the designer with a flexible facility in case the provided attributes are insufficient to allow the model to deal with a specific situation, but could be used for any other purposes.

For example the designer may want to indicate that a certain motor is to be painted a non-standard colour;

- **Graphical representation attributes.** These would be attributes such as icon shape, default screen location, default colour, line styles and weighting, etc. Currently these are not supported due to the lack of a graphical interface. (Refer to "Computer Environment" (8.2) below). Many of these attributes would be overridden by the display algorithm to highlight certain conditions. For example, one representation would colour each Type of Cable Rack a different colour while in another display, the colour a Rack takes on, could be determined by its % filling.

Detailed attributes are those attributes that differ from one Equipment Class to another. I.e. they are those characteristics which make motors different from switchgear for instance.

Detailed Attributes include:

- Those technical attributes that could be variant between two instances of the same equipment type. These would include, for example, various settings, the current rating of a switchgear's fuses, a transformers normal tap position, etc.
- Those attributes that, when combined with cost rates found in the financial model, would contribute to the total cost of an object. For example, terminal sizes on a switchgear (larger terminals are probably more expensive) or whether a switchgear has local current and voltage indication or not. For a link, its length would fall in this category. (Refer to "Costs directly coupled to the Physical model" (4.4.1) below).
- Those costs directly attached to a specific instance of an object. (Refer to "Costs directly coupled to the Physical model" (4.4.1) below).
- Class specific Administrative attributes,
- Class specific Model attributes. The ID of the item being supplied by a switchgear (Feed code), circuit number within a Board, etc.
- Class specific links to data other than the Reference data attached to Type.

#### 4.2.4.2 Links

As Links have only one class (namely cables), all attributes are common. Attribute categories are similar to those found in nodes, and quite a number of attributes are common between nodes and links. Some of the system grouping attributes applicable to nodes have no meaning to links. For example, a Structure code is meaningless, as a link could traverse any number of areas while linking one node to another. Currently, construction feedback attributes for links are far better developed than those for nodes. (This is because one of the main functions of CEEDS is to manage the cable contract). If the system is used to model links of a different class (such as optical fibre links, for instance) then link attributes would be structured the same as for nodes.

#### 4.2.4.3 Other Attribute Considerations

Currently only a small set of equipment classes are being modelled. These are Switchgear, motors and transformers. (This once again indicates the power bias within CEEDS). Any equipment not specifically modelled as an existing Class is regarded as "Miscellaneous". The Class called "Miscellaneous" has no details or Type and therefore no Reference attributes, only common attributes. Refer to Appendix A4.6 for other equipment classes being considered for separate modelling.

The detail of attributes applicable to the "Model Extensions" namely Cable Racking and Cable Drums are dealt with in "Extensions to the model" (4.2.7) below.

#### 4.2.5 Special Nodes and Links

In order to model certain real world situations and to enable design to take place with minimum data, "System objects and Pseudo objects" are required.

System objects are nodes and links required by the model to ensure consistency in the absence of real data and to make the model compatible with equipment labeling requirements (as laid down in the KKS rules, refer to Appendix A4.1). Examples are; when a cable is required to be coupled to one piece of equipment, but the exact node to which it is coupled on the other end is not yet known, then it is terminated to the "FLOATING" node. The model requires that a link has nodes at both ends, so a "System" defined node is used until the real node is known. The link is then disconnected from FLOATING and connected to its real node.

Pseudo objects are nodes and links that are created in order to make the model work in certain real world situations. To the model they appear as real objects but in real life they don't actually exist. Examples are: When one node is coupled to another via a bus coupling, there is no real cabling involved, but there is a link. A "Ghost Cable" is therefore created to link these two nodes. (This could be handled by creating a Link Class called Bus-Bars, but there are so few such occurrences that the extra effort is not justified). A Switchgear has an attribute "Feed Code" to indicate the piece of equipment being fed from that switchgear. Normally only one piece of equipment is fed, so one Feed Code is sufficient. There are cases though where multiple loads are "Daisy Chained". All of these nodes are "allocated" to a Parent node. This parent node then becomes the feed code for the switchgear. The parent node does not actually exist and is therefore a Pseudo Node. Refer to Appendix A4.3 for more detailed examples.

#### 4.2.6 Subtleties of power consuming nodes

When the analysis for CEEDS was first done, a node that was capable of consuming power was modelled in quite a simple manner. Now that real data is being inputted into the model, a number of inadequacies in the model have arisen. (Currently "work-arounds" such as the use of Pseudo nodes are being used to overcome the model's shortcomings. Refer to Appendix A4.3 for more detailed examples).

Some of these Subtleties are:

- Currently the load current, electrical power rating, power factor, etc. are quoted at rated full load. (These values should be available through reference tables). Subsequently cables and transformers are being sized according to these full load ratings. But the situation often arises where, for example, a 50 kW motor is used in an application that only requires 35 kW. This could be to use the same motor type as required for other applications that do require 50 kW, in order to reduce the proliferation of spares. Cables sized on the full load rating in this situation would thus be over sized. Similarly power factor correction could be incorrectly calculated. Such situations would currently be handled by attaching suitable notes to the specific nodes;
- Currently provision is made for only one set of ratings, i.e. only one power rating, starting current, etc. There are cases of components that could have a number of sets of ratings, multiple speed motors for instance. For cable sizing purposes, only the maximum of these ratings is necessary, but there may be other design considerations that do require such information. Similarly a given node may have multiple requirements such as Main AC power, Auxiliary AC and Auxiliary DC power. (This situation is currently handled by creating pseudo nodes to receive each supply, then grouping them into the real "parent" node);
- When board loading and power balancing are calculated, power nodes are classified as either continuous load, intermittent load or standby load. Board loading is not a simple arithmetic sum of all loads, because not all loads would be on at the same time, or started at the same time. Using the above loading classifications and corresponding diversity factors, a probabilistic loading profile can be determined for a board. This is where the problem arises, because some (usually critical) loads may have a chop-over system with multiple power sources. One source is its normal continuous feed while another is its standby feed. So a "schizophrenic" node appears because to one supply it should be seen as a continuous load yet the other supply wants to see it as a standby load. (This situation is currently handled by attaching notes to the relevant switchgear indicating whether they function as a continuous or a standby supply. This is not a satisfactory situation because cable sizing may be effected by the nature of the load, and the sizing algorithm cannot read notes).
- Currently, a node has a "duty" which describes it as Continuous, Intermittent or Standby. The Transformer sizing activity would make use of such duty information to determine load profiles. Cable sizing on the other hand also requires a "duty" description of sorts in order to determine the desired volt-drop limits. For example, cables feeding intermittent loads can be sized to allow a 5% volt drop, volt drops on main transformer feeds may not exceed 1.5% and on motor feeds, 3%. A problem arises if the "duty" used for transformer sizing is also used for cable sizing because a Motor load and a transformer feed are both Continuous but have different volt drop criteria.

#### 4.2.7 Extensions to the model

The main thrust of the Physical Model is to model an electrical network. Some extensions have been added to the model to facilitate Cable Racking design and Cable Drum Management.

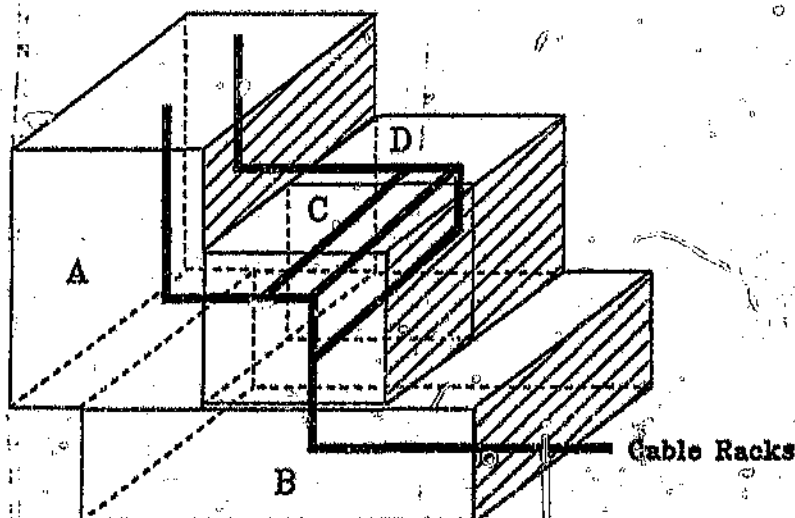
#### 4.2.7.1 Cable Racking

Cable Racking is required by the electrical model to determine the physical lengths of its links. Links carrying power need to be "sized" (i.e. Cable Type determined) and the number of parallel cables, the core area and insulation/armouring type are very dependent on the distance between a supply point and its load. (Signal carrying cables may also need special attention for long distances, e.g. extra screening, insertion of repeaters, etc). This distance is determined by designing "routes" between nodes in a 3D model. Cable Racks and support structures are then placed along these routes to facilitate the laying of a cable between these nodes.

Racks are made up of straight segments. These individual segments are modeled as nodes and the joints between them, as links. Cables are routed and laid on these racks and in the process, rack filling and mass loading can be determined by considering the physical properties of the cables currently allocated to any given rack.

CEEDS has four main functional areas, these being "Nodes", "Cables", "Cable Racks" and "System Administration". So one can see that the modelling, representation, management and construction of the Racking networks are a very relevant part of the complete system. To deal with the Racking model in too much detail would only clutter the main issues. Many of the principles which are applicable to the electrical model are common to the Racking model, although 3D representations play a greater role in the Racking Model. Refer to Appendix A4.7 for details on Rack Modelling.

One aspect that is of interest though, is the modelling of Volumes and Areas as objects. These areas/volumes are identified by Structure Codes, refer to "Systems" (4.2.2.2) above. Racking design is broken into "Chunks" that correspond to volumes defined by these structure codes. When a cable is to be routed between two nodes that are not in the same, or even in adjacent volumes, it becomes difficult to find suitable "in-between" volumes through which to route the cable. Currently this is being done manually (Refer to Appendix A5.4). If the volumes are modelled as nodes and links are created to represent connections to adjacent volumes, then a network of Volumes can be built up. [MDL 16]. In this way, a searching algorithm can be used to find suitable routes between far removed volumes. Refer to Figure F4.2.07 for an example. There are other reasons for modelling these volumes. If a representative point (e.g. centre) can be determined for all volumes, then equipment allocated to that volume (by attaching a Structure code to the node) can inherit the X,Y,Z coordinates of the representative point. In other words the node is "loosely" positioned within a volume. This could allow other design processes, that need a node's location, to continue with the estimated position until actual placement takes place. It would also be possible to verify the coordinates of equipment already positioned in 3D space for integrity purposes. This would be achieved by checking that the nodes' X,Y,Z coordinates fall within the limits of the volume that they are allocated to.



From a Rack Connectivity Point of View, the following connections exist between volumes:

A - C  
B - C  
C - D  
A - D

No direct paths exist between A - B and D - B.

**Relations between Volumes to find suitable Routes**

**Figure F4.2.07**

#### **4.2.7.2 Cable drums**

This is essentially a real world extension to the model. Cables are delivered to site on drums. Each drum is registered with the system. Some of the attributes would be "Drum Identification Number", "Cable Type", "Cable Delivered Length", etc. A cable is designed on the system and an instruction is issued to the contractor to pull a cable according to the details given by the system (e.g.: pull a cable of Type T, with a theoretical length of X, from Node A to Node B). The contractor then removes cable from one or more drums to carry out this task. Once the job is complete, feedback from site would indicate the number and length of cable segments used to construct a given cable. For each of these segments, the Cable Drum from which it is removed is also given. The people managing the cable contract are now in a position to detect any significant discrepancies between design length and actual length. They are also able to determine how much cable should still be available on any given Cable Drum. If the calculated available length and the actual available length differ significantly, then appropriate investigations can be activated. Other data fed back into the system, such as number of joints (also calculatable), and termination types (e.g. what glands were used), etc., make very accurate cost estimates possible by referencing the relevant Financial Model (Refer to "Financial Models" (4.4) below and Appendix A4.10 for more details).



Refer to Appendix A4.3 for a description of how a Pseudo Cable drum is used to manage some real world situations such as the re-use of cables).

#### 4.2.8 The Deletion of Objects

The Deletion of Nodes or Cables from the system, although being functions, are very dependent on the model (Refer to the section on "Design Functions" (5.3) below). The model restricts the deletion of objects based on "Real Life" rules. For example, a cable cannot be deleted from the system if it is already allocated to a Rack. It would first have to be un-routed before deletion can take place otherwise the Rack loading would be false. A node cannot be deleted if it is being fed from a switchgear. The switchgear's feed code must first be nullified before deletion is possible, otherwise the switchgear contains false information in that it indicates that it is feeding a non-existent node. No node can be deleted if it has one or more cables coupled to it. Either the cables must be un-coupled and re-coupled to another node (such as "FLOATING", refer to "Special Nodes and Links" (4.2.5) above) or the cables must be deleted. A very elegant software solution was found to enforce such rules without having to explicitly state the rules. (Refer to Appendix A4.2 for details). There are other non-physical criteria that could prevent the deletion of objects, refer to the "Design Procedure Model" (4.3) below for details.

Refer to Appendix A4.3, for some interesting "compromises" to the model in order to manage real world problems.

#### 4.2.9 Signal Model

The Physical Model described above is mainly used to create a configuration of equipment on a physical level. Analysis of power is possible because firstly it is relatively simple and secondly, there is a one-to-one relationship (in most cases) between a model node and a consumer of power or a power control circuit such as a switchgear. Control signals tend to be far more complex because signals are transmitted throughout the network of physical nodes and links to control complex systems consisting of many nodes. It may be possible to superimpose a Signal Model over the Physical Model if individual cores of signal cables are modelled (The micro level mentioned in "Graph based Node-Link Model" (4.2.1) above). Technology complicates matters in that multiple signals can be sent down a single link (e.g. Time domain Multiplexing), so the superimposition between the requirements for a Signal Model and those offered by the Physical Model are not simple. (This aspect has been briefly investigated but the majority of the design of control systems are done as turn key systems by external contractors. There was therefore insufficient impetus to drive further).

## 4.3 Design Process Model

### 4.3.1 General discussion

The Physical model discussed above contains technical, spacial and administrative attributes. Financial attributes are covered under the Financial Model below. So far nothing has been mentioned about the all-too-crucial dimension of time. The Physical model presents a static picture of the current state of affairs. There are two aspects regarding the question of time and sequence and these are:

- **Project Planning:-** This gives Time and Date information and high level sequences of design / construction activities. The Project Plan is very project specific and looks at the total project life cycle from feasibility studies to final hand-over;
- **Detailed Design Procedure (Sequence):-** This governs the fine detailed sequences that need to be adhered to in order to carry out a design. It is project independent, is based on logical requirements and has only sequence. No duration, time or date information is contained in this model.

There is obviously a relation between these two aspects. "Chunks" of the Design Procedure Model may be referred to by a single activity on the Project Plan. For example, an activity may read "Design cables for Unit one Emergency Supply Board". The actual sequence that is to be followed in doing the design would be detailed in the Design Procedure Model. In other cases, an activity on the Project Plan could reflect a single activity in the Design Procedure Model. For example, "Print Cable Pull Cards" on the Project Plan is also an activity contained in the Design Procedure Model. Refer to Figure F4.3.01.

### 4.3.2 Project Planning

As was mentioned in "Systems" (4.2.2.2) above, a project is divided into work packages by means of a Work Breakdown Structure (WBS). These packages are determined by factors such as:

- When must a certain activity start;
- What are the dependencies between activities, i.e. what must be completed before something else may start;
- When is access to an area available;
- When are the necessary support structures available, etc.

Each of these packages can be hierarchically broken down further. There are dependencies between tasks that must be established, durations of task must be estimated and dates established and calculated. It is not intended to go into the details of setting up PERT charts etc. as they are well established procedures. It is also not intended to delve into this aspect too deeply because there are existing facilities and all that must be established are interfaces. Refer to "CAE at an Engineering discipline level" (2.1.2) above.

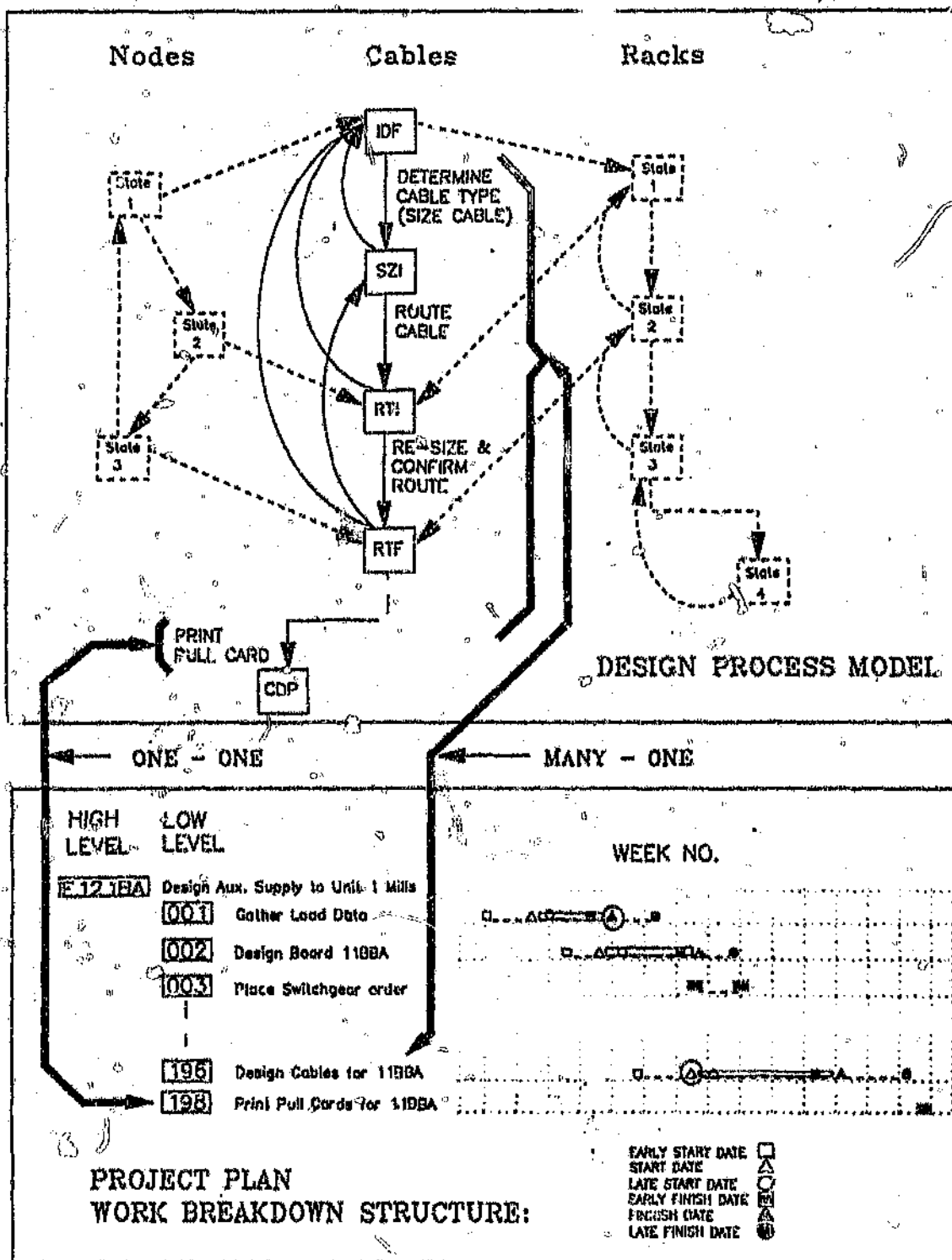
What must be determined is how this Project Plan ties up with the physical model. All that is required is to attach a "Time Tag" to each object in the physical model. This Time Tag is the name/code of a Work Package/activity in the Project Plan. If any time and date related information is required for a given Node, its "Time Tag (Work Package Name)" can be obtained from the Physical Model and the Project Plan subsequently queried for details about this Work Package. In this way the project plan is largely independent of the physical model, from a software point of view. They are strongly dependent on a conceptual level though, because the whole idea of the Project Plan is to facilitate the construction of the design contained in the Physical Model. High Level Planning usually takes place before detailed design and is determined by many factors of which detailed design is only a small aspect. Other aspects include the availability of finance and resources, the urgency for the facility, political climate, etc. Low level planning, on the other hand, is highly dependent on the design and is usually initially estimated based on past experience and is then updated as the design proceeds. Similar to the "Work Package Tags" attached to Objects in the Physical Model, the Low Level Project Plan "Model" would have "Equipment Tags" attached to the Work Packages. For example: an activity could read Install Motor "10LAC10", where 10LAC10 is the identification of a node in the Physical Model. The "Linking" between these two models is not a simple one because many objects in the Physical Model may be referred to in one or more Work Packages and vice versa. This is an area where further investigation would be required if tight coupling is required for integration purposes.

(Refer to Appendix A4.6 for details on how this is currently managed in CEEDS and for limitations imposed as a result of the models not being tightly coupled.)

#### 4.3.3 Design Sequence

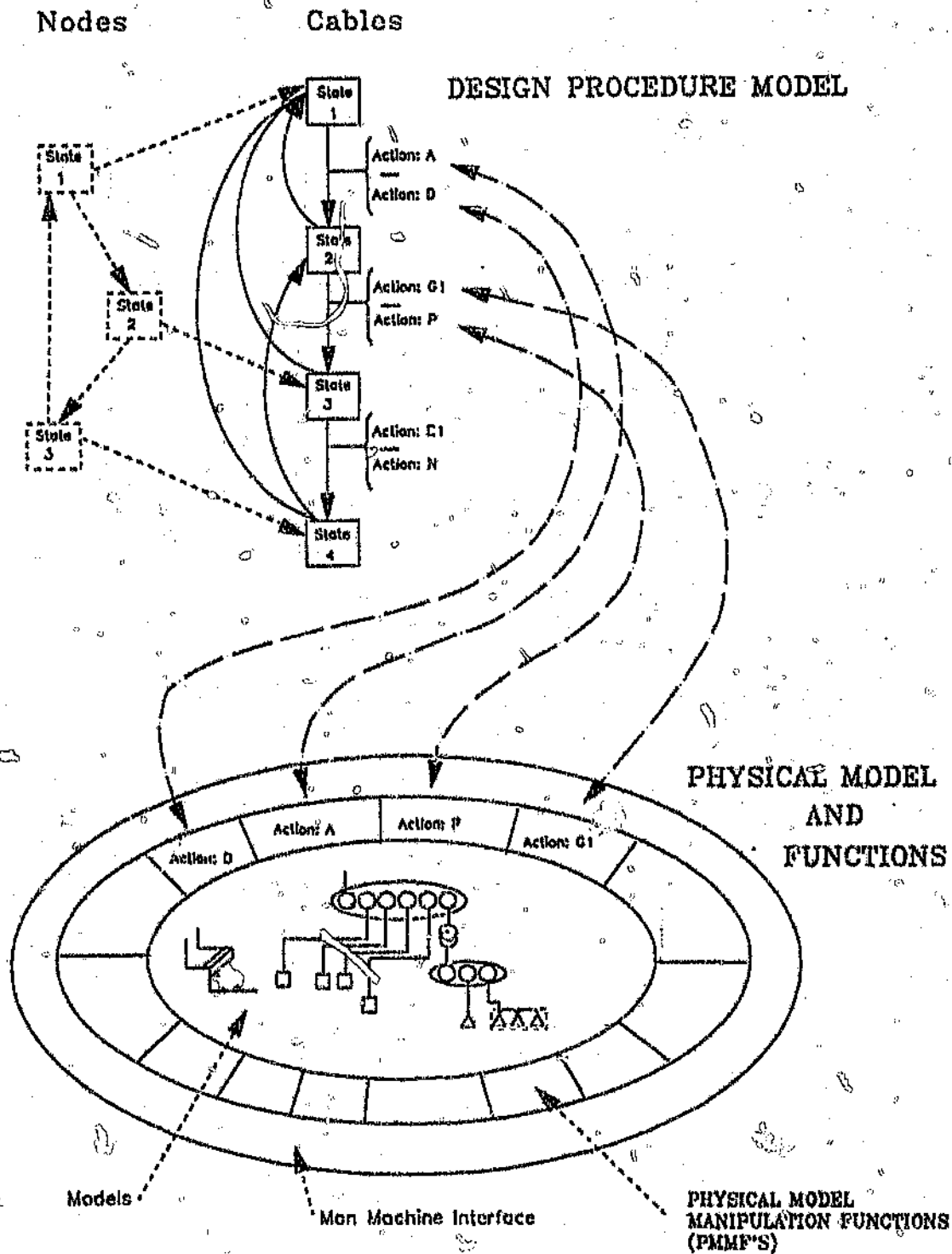
There are many model manipulation functions surrounding the Physical model. (Refer to the chapter on "System Functionality" (5)). These enable a designer to carry out his design by building up and manipulating the Physical Model. (Refer to Figure F4.3.02 for a depiction of the relationship between the Design Process Model and the Physical Model Manipulation Functions). If there is no control over which functions are used, and in what order, there can be no design support from the system and the system can therefore not guarantee the integrity of the design. The intention of the Design Process Model (DPM) is to enable software enforced control of the manipulation of the physical model. There are a number of reasons why a design sequence should be determined in a formal manner. These include:

- determinism is required if a computer system is to be used to control the design process;
- manual procedures are based on a solid foundation (backed up by computer based tools);
- it is known exactly what needs to be done and in which sequence, in order to achieve given results;
- it allows for very tight progress control and monitoring; and
- it provides the system with the ability to ensure design integrity of a very high degree.



Relationship between the Project Plan and the Design Procedure Model.

Figure F4.3.01



Relationship between the Design  
Procedure Model and the Physical Model.

Figure F4.3.02

The technique utilised is one in which all objects in the Physical Model have various Statuses attached to them. These statuses depict the state of the object regarding a particular design procedure. For example there is a Design State, a Coding State, perhaps a Construction State. A Cable could for instance have a Design State of "Identified" followed by "Initial Type Determined" followed by "Initial Route Determined" etc.

The Design Procedure model depicts:

- The valid states an object can find itself in;
- valid sequences of states, i.e. an object can only progress from any given state to a limited set of other states;
- the actions that need to be carried out on an object to get it from one state to the next; and
- the dependencies between states of various inter-relating objects, i.e. what pre-conditions need to be met before actions can be allowed to move an object from one state to the next.

When a Physical Model manipulation Function (PMMF) wants to carry out a task on the Physical Model the following protocols take place:-

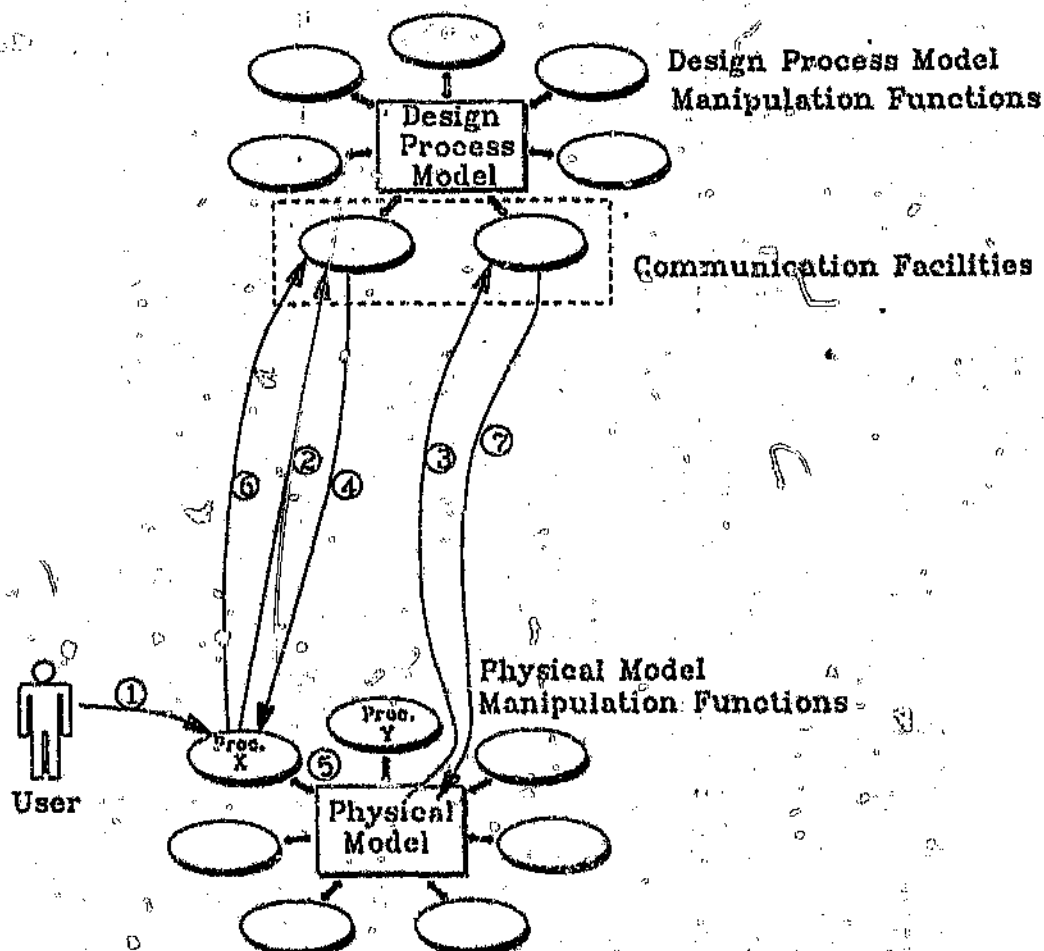
- The PMMF firstly identifies the object(s) to be manipulated;
- It then informs the DP model (DPM) that it is "Function X" (its function ID) and also passes on the list of objects it intends to manipulate;
- The DPM then queries the objects to see what their current statuses are;
- The DPM then check its State Dependency Information to see whether the PMMF is allowed to carry out its function on the objects;
- If permission is denied, the function is informed of the reasons, and the PMMF backs off;
- If permission is granted, the PMMF attempts execution of its function;
  - If execution is unsuccessful, the PMMF offers reasons and backs off;
  - If execution is successful, the PMMF informs the DPM.
- The DPM then in turn queries its internal information to determine what the following states should be and updates the objects with this new state.

Refer to Figure 4.3.04 for a description of the communication protocol between the DPM and PMMF's.

#### 4.3.3.1 Representation of the Design Process Model

The proposed Design Procedure model is conceptually a State Dependency Graph indicating how a certain State attribute of a piece of equipment (or a cable or rack) moves from one state to another depending on its current state and on activities carried out on the piece of equipment.

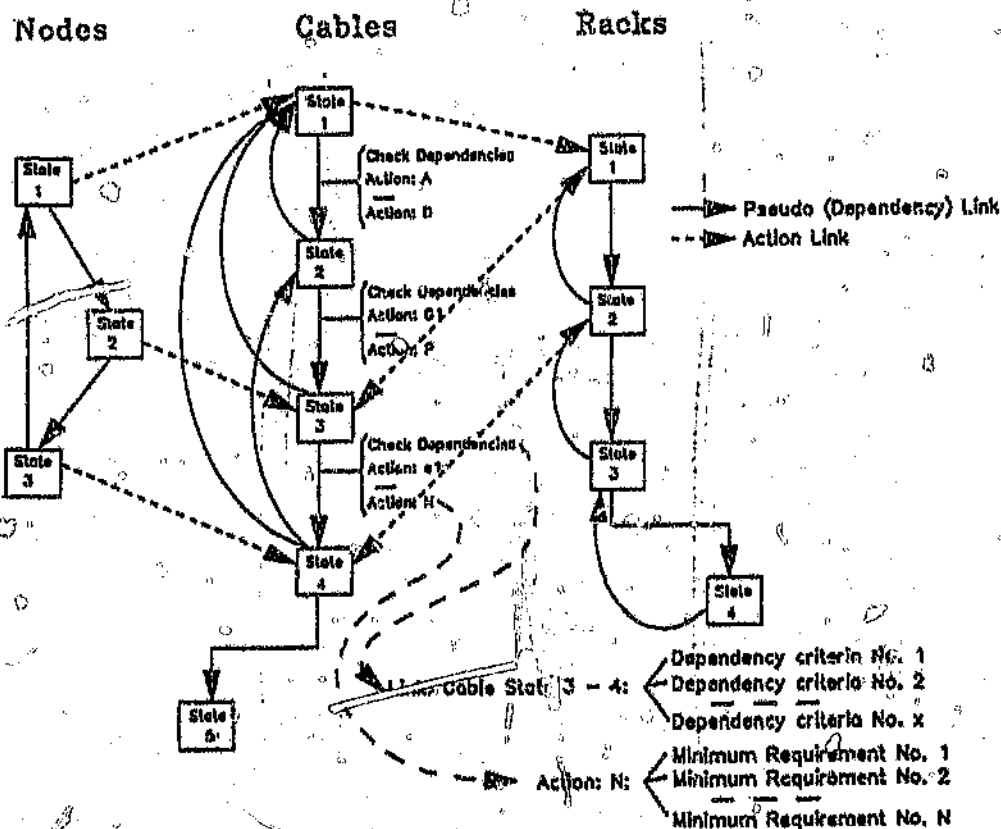
Refer to Figure F4.3.03 for a depiction of such a graph. The State Dependency Graph consists of Nodes (Vertices) and Links (Edges) (as with any other graph).



- 1) User selects objects to be worked on and activates a given function (eg. size these cables).
- 2) DP Model wake up, I am Process X and want to work on this list of equipment.
- 3) The DP Model then queries each of the objects to determine their current state. Then for each object with its state, it determines if the requesting process is allowed.
- 4) The DP model then grants permission to Process X to carry out its operation on valid objects only.
- 5) Process X carries out its functions.
- 6) Process X informs DP Model of its success in executing its function (either successful, partially or not at all).
- 7) If successful, the DP Model determines the next state for the objects and updates the state attributes of the object.

The Communication Protocol  
between the DPM and PMMF's.

Figure F4.3.04



In order to progress from one State to the next, all dependencies must be valid and all the necessary Actions carried out.

### State Dependency Graph for the Design Process Model

Figure F4.3.03

The Nodes depict State, for example "Identified", "Initial Type Determined", etc.

The links depict actions that are required to be carried out in order to get from one state to the next. There can be multiple links between nodes, these indicate alternative ways of getting from one state to the next. (For example, to get a cable from "Identified" to "Initial Type Determined" differs between power cables (that must be sized) and Process Control Cables (that are possibly selected from a standard). The Actions coupled to a link can also be supplied with a list of "Minimum Data Requirements". (Refer to "Design Automation" (4.3.3.2) below).

Many actions would be simple "status/dependency checks" to determine prerequisite states of other related equipment. Graphical representations of the DP model would depict two types of link. The one type has already been described above (called "Action Links");



the other type being "Pseudo links" (deciding dependency on the states of other equipment (called "Dependency Links"). Dependency links are not explicitly modelled as they are included in "Dependency Check" actions, which are part of Action Links.

An example of a "Dependency" link (or "Dependency Check" Action) is; a Cable's ID Number is partly derived from the Equipment Nodes to which it is connected. If a designer attempts to carry out the "Allocate Cable Number" action to get the Cable ID from Dummy status to Approved Status, the code statuses of both end Nodes must indicate that they are in "Approved" status otherwise the action cannot be executed. There is thus a dependency on the states of other equipment. For visualisation purposes, this dependency can be indicated by a link but is physically carried out by action(s).

The actions depicted by the Design Process Model are limited to computer processes. Manual contribution is the supplying of data. If an action is triggered (e.g. by a designer) and insufficient data is available to carry out the task, it aborts and informs the operator of what data is missing. In other words, the DP model controls the internal operation of the tool. It has no physical connection in any way to the Project Plan Model described above.

By having a stand alone DP model, "Physical model manipulation functions" need only be designed to be able to converse, in a standard, pre-determined manner, with the DP model. The DP model decides on the validity of an action and the sequence of events to follow. The majority of software servicing the Physical Model is therefore NOT dependent on the design process. CEEDS does not currently have a stand alone DP model as described above. A sub-section of the cable design DP model is hard coded into relevant Physical Model Manipulation Functions. This leads to the very undesirable maintenance problem of having to modify many programs if there is any change in design philosophy that affects the states and dependencies between states. Refer to Appendix A4.8 for details on the current system for Cable Design (not construction).

#### 4.3.3.2 Design Automation

One of the very exciting possibilities opened up by the DP model is the idea of "Automated Design".

A design process can be started up by a designer who triggers a function via a menu option or a screen based manipulation. If the DP model permits the execution of this function, and execution is successful, the DP model is in possession of the necessary information to trigger subsequent processes. If execution fails, the DP model may contain alternative actions to be carried out. These subsequent and/or alternative actions can be automatically triggered by the DP model itself, with no human intervention necessary. The topic of minimum data requirements mentioned previously, now becomes an issue. Triggered functions can only execute if their minimum requirements are met, otherwise they abort, so a chain of functions would continue as long as there is enough data to work on and would stop as soon as it "ran out of data".

One method of getting the chain of events re-started is by providing the necessary data and then manually re-triggering it. Another method is to have a continuously active background "Minimum Data monitor". This "facility" has knowledge of each PMMF's minimum data requirements. (These minimum requirements should be attached to the functions themselves and not to the Design Procedure Model). It continuously monitors the Physical Model for equipment statuses and currently available data. It then checks the DP model for valid actions following on from current states. It then checks for those functions' minimum requirements and compares them to the data available in the Physical Model. If it detects that sufficient data is available, the Minimum Data Monitor can trigger the relevant functions. The functions then start up in the normal manner and the chain will continue for as long as sufficient data is available. This function is similar to the "Truth maintenance facilities" found in some of the more sophisticated Expert System Shells. With the incorporation of AI assisted decision facilities, very powerful design automation facilities can be developed.

#### 4.3.3.2 Some points to note

The DP model MUST support iterative design. Only the design of very simple systems follow a linear sequence from start to finish. The reversal of states to previous states must be possible and must be seen as normal. Up-front estimations, followed by iterative refinements are necessary to prevent the classical "Chicken and Egg" problem. For example, Cable Rack design depends on Cable design which in turn depends on Cable Rack design.

The designers of the DP model must always guard against having to know too much information up front. The guiding principle should be that a design must be able to be taken as far as possible on absolutely minimum data.

One facet that must not be overlooked is that the determination of exact states may not be all that easy. For example, How can the situation be managed where specific attributes of an object could be in various states. For instance, the Power Rating of a motor could be an estimate, a preliminary value or a final approved value. The degree of confidence of one attribute could affect attributes of related equipment. For example, the cable designed to supply the above motor cannot be determined with 100% certainty before the attributes on which it is based are all known with 100% certainty. The certainty with which some data is known, is very important. This is because the probability of re-work on a piece of equipment once it is manufactured and/or installed could be determined using an empirical formula or a set of rules. A cost could be attached to such work and provision made to either counter the situation by insisting on more accurate data or accepting it and making the necessary financial arrangements. This aspect of determining the "Goodness" of available data has only just started receiving attention and much work still needs to be done. (See Appendix A4.11 for a proposed method of handling this system).

Administration of DP model is a very critical activity as absolute consistency must be ensured. The mechanism of automated design is reasonably robust because of its dependency on minimum data requirements, but if the DP model is inconsistent, it would be difficult to predict what could happen. Graphic visualisation and manipulation of the DP model would be a great aid. The DP model and the Physical Model are very similar from a software point of view and functionality could be shared. The DP graph is alterable by adding or deleting states, altering "dependencies" and altering allowable actions between states. (Academically, this point of having one model controlling the modification of another could be taken to ridiculous lengths because "Rules" could be established for the modification of the DP model, and rules about how to modify these rules etc.)

#### 4.4 Financial Models

In order for CAE systems to perform a role in real life design and not just academic/theoretical design, they need to have a financial component. Optimal design is seldom the best technical system, but the best value for money system<sup>1</sup>. Certain facilities are required, such as:

- A facility whereby a cost<sup>2</sup> of the design as contained in the model can be determined. (Refer to "Limitations on Physical Model based Financial Facilities" (4.4.1.3) below);
- Facilities that are able to give an indication of the cost of Re-Engineering (e.g. drawing updates, re-distribution, etc.) as a result of a change to the model; and
- Other facilities that aid in the financial management of a project.

The first two categories of cost are able to be derived from the model, the third category would require extensions to the system.

##### 4.4.1 Costs directly coupled to the Physical model

Certain costs are able to be derived directly from the Physical Model while others have no direct relation with the model but are still contributory to total cost. Refer to "Limitations on Physical Model based Financial Facilities" (4.4.1.3) below. The costs that are of interest to a new project are Capital Expenditure costs and an indication of Re-engineering costs.

- 
- (1) There is a delicate balance in obtaining "best value for money". What might seem to be the best initial solution from a capital expenditure point of view may turn out to have higher life cycle running and/or maintenance costs than an initially more expensive solution).
  - (2) Notice: "A" cost, not "THE" cost, because firstly, there are different types of costs such as Engineering/Design costs, initial purchase and installation costs, replacement (spare parts) costs, etc., and secondly, any of these costs can be made up of a number of components, some derivable from the model, others not.)

#### 4.4.1.1 Capital Expenditure costs

These are costs that can be directly allocated to a piece of plant. These costs can be divided into three categories:

- A. Those that can be allocated to the Class and Type of equipment and are fixed. For example, a certain type of motor may cost a given amount as a component. Irrespective of where on the plant such a motor is used, this component's cost is the same.
- B. Those allocatable to the Class and Type of equipment but are dependent on certain physical or technical attributes that are explicitly contained in the model, for a given instance of that Class and Type. For example a cable's cost is influenced by its length and number of joints.
- C. Those that are dependent on attributes, not contained in the model, for a specific instance of the equipment. For example, the motor in (A) above may, in one specific instance, require a special rust proof coating whereas in other instances this extra is not required. Rust proofing may not be an attribute catered for in the model.

Categories A and B are costs which can be accessed through the Class specific Type Reference Tables. (Refer to "Type Reference Tables" (4.2.2.1) above). Other costs specific to an instance (Category C) would need to be allocated to the specific object in the physical model or would not be able to be accounted for by the system. (The scope of the systems modelling capabilities must be limited at some point).

Factors which make up the Component cost are divided into categories such as Material, Transport and Labour. These in turn are further categorised into Local Content, Overseas content, import duties, etc. The exact structure is very contract dependent. (Refer to "Limitations on Physical Model based Financial Facilities" (4.4.1.3) below). The Financial models are "Hooked" into the Class/Type Reference Tables by means of a "Cost Reference". All costs and charge rates are attributes of such a Cost Reference. A Cost Reference is attached to the Type of an object. There may be more than one cost reference in the Type Reference Tables. Design and Construction projects would access the "Capital Expenditure" Models while Maintenance systems may access "Spare and Replacements" Models.

There need to be sufficient attributes describing an object so that, if used with the Financial Model attributes, a cost can be determined. For example: a cable's cost is dependent on its Type, length, number of joints, termination types at each end, etc. Each of these attributes must exist as part of the model for cables and the financial model must contain corresponding rate prices (such as cost per meter, cost per joint, cost per specific type of termination etc.).

Plant Component costs can now be determined and therefore grouping plant into Systems could give indications of "System costs". For example, the Turbine system costs so much, all the equipment in a given room costs something else, etc.

#### 4.4.1.2 Re-Engineering Costs

Exact Re-engineering costs as a result of a change cannot easily be extracted (unless the cost of design activities is available some how), but an indication of the extent of the change can be obtained. For example, if an ID Code is changed, a report could be extracted to indicate the number of documents in which this code exists and thus the number of documents that may require a new revision and re-distribution. If a cost can be allocated to the changing and re-distribution of a document, an indication of the cost of the change could be determined.

#### 4.4.1.3 Limitations on Physical Model based Financial Facilities

The total cost of a project is dependent on many factors external to the direct costs associated with the physical plant. Hence, the emphasis on "A" cost above, and not "THE" cost. Unless a CAE system incorporates complete financial facilities for a complete project, the costing capabilities are limited to those that can be extracted directly from the Physical Model. For instance, if a cable trench needs clearing as a result of rain and mud, and this activity is charged for, there is no easy way of linking such a cost to the physical model. Costs incurred as a result of bad weather, strikes, penalties, re-work, damage repair, etc. are all real costs but cannot always be directly coupled to the model.

Another limitation of the Financial Models in general are that it is very difficult to make them generic (such as the physical or design process models). There is a very strong dependency on specific contracts and pricing structures. For example, the Cabling contract may have very different costing practices to the Switchgear contract.

#### 4.4.2 Other Facilities

These facilities have not been investigated in detail but it is anticipated that the following facilities may be useful in an integrated system.

- A priori facilities such as Budgeting, forecasting, pricing, estimating etc.;
- Historic facilities such as Invoice verification, comparison to budget, actual cost versus forecast costs, etc.; and
- Time coupled information: forecast rate of invoicing, cash flow over a period, etc.

Integrated spreadsheet facilities could be provided for "what-if" type analysis.

The financial facilities on CEEDS are currently limited. Expanded requirements are being investigated, but, CEEDS is first and foremost a design aid, not a comprehensive financial system. For a more detailed account of equipment currently and shortly containing Financial facilities within CEEDS, see Appendix 4.10.

## 5 System Functionality

### 5.1 Introduction

This chapter discusses aspects of the required system functionality and those areas that show promising potential are highlighted. It should be noted that the facilities mentioned here are based on functions currently implemented on CEEDS and on anticipated functions within a computer environment that supports graphical facilities. There are probably countless other facilities and functions that could be provided to meet a very wide scope of user requirements. The areas of Design and Internal requirements have been concentrated on, and the areas of Financial Facilities and Construction Facilities are briefly discussed.

### 5.2 The Man Machine Interface (MMI)

This is the mechanism by which users interact with the system. If the MMI is not easy to learn and is not consistent in the way it operates, then the system will be difficult to use and may result in system rejection. Consistency is an absolutely vital criteria. Available computer environments place a restriction on the user interface facilities that can be provided. For example, CEEDS is implemented on a Block Mode, non-graphical IBM Mainframe with very limited screen display capabilities. Screen display is limited to normal monochrome text with high/low intensity only. This obviously limits functionality and effective visualisation capabilities of the system. Modern systems offer vastly improved facilities (Refer to "Proposed Future MMI" (5.2.6) below). It is not intended to go into the details of the psychology and ergonomics of MMI design. These areas are well researched and documented. [SC 03,04,05,06,08,14] Besides, MMI capabilities are usually defined by the capabilities of the Commercial Software chosen to do the job.

#### 5.2.1 Menu facilities

Menu driven interfaces are preferable to command driven interfaces because humans' recognition capabilities are usually superior to their recall capabilities. Menus can either be text based (with a short description of the option) or icon based with a graphical representation of an activity, or both. Icons have the advantage of being language independent, but are sometimes difficult to decipher if the representation is not obvious. There are many types of menu styles, pop-down, pop-up, expanding, scrollable, point and click, enter selection, etc. Once again the methods chosen will be strongly dictated by the chosen software environment.

Important Menu features include:

- The user should not be able to get lost. There must be some implicit (e.g. one menu expanding into the next and so on) or explicit (e.g. menu name/address) method of knowing where one is in the Menu Structure;
- A globally available facility should exist with which to query the menu structure so that desired options can be found without having to flounder around a massive structure trying to find where a particular option is;
- Menu navigation should be quick, easy and obvious;

- The menu options themselves can give an indication of expected behavior. For example on CEEDS the symbol following the text description of any option gives an indication of expected behavior. The following conventions were used.
  - "... " - a further menu is to follow;
  - " " - a screen form is to follow;
  - " " - a program is to be triggered for execution;
  - " " - a report is to be expected. For example, the option "General Data Queries..." would lead to a further menu;
- If possible, there should be multiple ways of selecting an option, e.g. type in option number, or first letter of option, or highlight with arrow keys or point with a mouse;
- Access profiles should be built into the menus. A fully dynamic menu structure would only display those options to which a user has access. An alternative is to use an indicator, for example, a marker, colour differences, etc. to indicate usable options. It is preferable to display all options and use an indicator, because a user can then become familiar with what else the system offers even though he may not have access. A menu option may also be invalid as a result of the state of the system and not because of restricted access, or a combination of both reasons. Different attributes should be used to distinguish between those options to which user access is denied and those that the system has restricted;
- Menu structures should be kept as "shallow" as possible. Rather group more options into one menu than have many menu levels with few options;
- Grouping of options should also be logical so that related options are found in the same vicinity as each other. The previous point and this one could be contradictory so a happy medium may need to be found;

### 5.2.2 Data manipulation facilities

In a textual system, data manipulation usually takes place via "Screen Forms" or "Dialog Boxes". Some manipulation could take place in a graphical manner, refer to "Design Functions" (5.3) below. Design of such "Dialog Boxes" is important to prevent user confusion. [SM 20,22,23,24]

Relevant issues in the design of Data Manipulation facilities include:

- The contents (i.e. object attributes, their descriptions and dialog/messages) of a screen/window/box should be selected to depict the function performed by the screen. For instance, screens allowing data updating usually contain only relevant fields and usually only display one record at a time whereas screens for viewing only, would contain more information and usually depict multiple records so that comparisons can be made;
- A user should be able to detect which fields are updatable, and which aren't by a display attribute. For example, updatable fields could turn red while all others remain some other colour;

- Model integrity should not be jeopardised by data manipulation via a screen form. For example, an object's total data may be stored in a number of tables. A particular screen form may only be looking at one such table. The "key" field connecting all these distributed attributes together must not be alterable from within any Screen Form, and similarly a record should not be able to be deleted from only one table. Refer to "Design Functions" (5.3) below and to "Physical Database Structures" (4.2.3.2) above;
- A user should be able to detect whether a screen gives him view only, insert only, update only, delete only or any combination of these privilege levels. The naming convention used on CEEDS indicate the function of a screen. The last letter in a screen form name, which is displayed with the form, would be V - View only, I - Insert only, D - Delete only, U - Update only and M - Maintain/Full privileges. Refer to Appendix A6.1 for more details;
- Validation of data entered in a screen is discussed in "Design Verification Checks and Model analysis" (5.3.2) below
- Screen form facilities are also dependent on the technology being used. These vary from tabbing between simple text field through to "devices" such as sliders, dials, gauges, push buttons, radio buttons, switches, check boxes, selectable lists and many more.

### 5.2.3 Reporting Facilities

See "Documentation Production Facilities" (5.5) below for details of reporting facilities. From a Man Machine Interface point of view, there are a few requirements for the production of reports:

- Except for user defined reports, all other reports should be accessible via a displayed list. The user should not be required to remember commands and report names;
- There should be flexible query facilities on the list of all available reports so that a user can find a suitable report by simply querying in a "key word/text retrieval" manner;
- If a report expects parameters of any sort, the user should be made aware of this fact before running the report;
- When a report prompts for a parameter:
  - the message should be clear;
  - It should contain an example of expected input;
  - it should give an indication of allowable format and length of the variable being prompted for;
  - it should contain and indicate a default if the absence of a value causes the report to have no meaning; and
  - there must always be the option to quit, and return to a previous activity from anywhere in a series of prompts;
- Screen dumps should be possible;



- A report should be able to be viewed on screen before being submitted for printing. This is to prevent paper wastage by printing possibly unwanted reports;
- Printing facilities should be flexible. A user should be in the position to choose from a number of possible printing locations and on various types of printer/plotter. For example, people on the construction site should have local access and printing facilities. Large printouts should be routed to high speed line printers where as, small query type printouts should be printed locally on PC printers for instance.
- The status of printing facilities should be able to be monitored and manipulated. Printer queues should be able to be viewed and jobs must be able to be re-prioritized or dropped (within security privileges, for example one user should not be able to drop another user's job from the queue).

The last two points are obviously dependent on the Computer Environment being used. Graphical reporting facilities are entirely dependent on the Computer Environment. Refer to "Computer Environment" (6.2) below.

#### 5.2.4 Query facilities

Refer to "Utilities" (5.6.4) below and "Data manipulation facilities" (5.2.2) above for details on text based query facilities.

Some of the requirements for Graphical Query facilities are described in "Design Functions" (5.3) below, but no details have yet been investigated. This work is currently underway.

The basic philosophy that should be applied is that all data, excluding confidential data, must be available to be queried in a totally flexible and easy to use manner.

#### 5.2.5 Help facilities

There are no clear lines between conventional Help, guidance about how to use the system, what to do when an error occurs, giving advice about what to do next, etc. Explicit help facilities are therefore only partially useful. What is being investigated for CEEDS is the use of context sensitive Hypertext documentation facilities. These facilities would also be able to link into relevant reference tables and "Error message / Corrective Action" Tables. The user interface is given some intelligence in that it is aware of what the user is doing and in some cases would be able to assist a user in carrying out a given function. For example, if a user unsuccessfully attempts to size a cable, the corresponding "topic" in the Hypertext Documentation is presented, the Cable sizing function's analysis of why the attempt failed, is simultaneously displayed and depending on the reason for failure, possible actions can be recommended. The "Help" facility is therefore not a separate part of the system but a fully embedded function. Refer to "System Documentation" (6.5) below. A user should always be able to access help of some sort, even in the event of a fatal abortion of a current activity. The system should be robust enough that one process aborting does not affect the access to help information.

### 5.2.6 Proposed Future MMI

Currently CEEDS has a very limited MMI. The models described above are very difficult to visualise in terms of records and tables, but are easily visualised when viewed graphically as nodes and links. Refer to "Design Functions" (5.3) below. When a computer representation matches a user's "Mental Picture", i.e. matches his own conceptual model, then usage of the system is greatly enhanced. Modern workstation environments have extremely powerful MMI features including Multi-window, multi-tasking, click and point, and all the other fancy features of modern Graphical User Interfaces (GUI's). The details are available in many references. [SC 00,11,13].

### 5.2.7 Other factors

The MMI should be modelled within the system in a manner similar to that in which the system is modelling an electrical network. In other words, MMI components such as menus, screen forms, reports, help screens, etc., are all considered as objects. In this way, software maintenance is kept to a minimum as new facilities can be created by developing them in isolation and then simply "registering" them in various places in the system. Integrity enforcement and query facilities are available to enhance and determine the soundness of the system in much the same way as that in which the system attempts to ensure the integrity of the models it manages. Refer to "Documentation Production Facilities" (5.5) below.

## 5.3 Design Functions

One of the main functions of a CAE system is design support. The facilities described here are those required for carrying out the design process. These are the Physical Model Manipulation Functions (PMMF's) discussed in the "Design Process Model" (4.3) above. Facilities needed for creating and maintaining the model, verifying and analysing the model and ensuring Coding correctness are discussed. Most of the facilities are required, irrespective of whether the MMI is text based or graphical. The discussion presented below is assuming the availability of suitable graphical facilities.

### 5.3.1 Creation, modification and deletion of Model Objects

The model consists of objects falling into the categories of Nodes and Links. Cable Racking design is ignored for the purpose of this investigation. Principles, and details applicable to the Racking Model are very similar to those described below for the Electrical Model. The functional mechanisms presented to the user for the manipulation of these objects **MUST BE AT THE OBJECT LEVEL** and **NOT** at the internal database structure level. In other words, if a designer is creating a node, he must do just that. One operation. He must not have to create several records in several different databases so that when combined, they make up the attributes of the node. Similarly for deletion. When a node is deleted, the function carrying out this deletion **MUST** take care of all administration at the database level in order to ensure the continuous integrity of the model. There should also be the facility for making "temporary" changes, running an analysis, then "rolling back" to what the design was before the changes were made. In this way "what-if" type studies can be done.

### 5.3.1.1 Nodes

Node creation would take place by activating a menu option, or carrying out appropriate actions with a mouse, to present a "Node Creation Details Box". The Node will have to be given an ID Code, a description and a Class. The creation function will then firstly verify that the ID Code is unique. If so, it will secondly query the "Model Data Structure" database to find out where the relevant insertions into the underlying database are to take place. These insertions will then take place. In most tables (excluding the one in which the description is stored), only the ID Code is inserted (along with the Date Created and Last Update fields). The decision as to where the insertions are to take place is decided purely on the Class of the node. See Appendix A4.2 for details on this "Model Data Structure".

The moving of a node only has meaning in a graphical representation. Moving a node in a 2D representation, would be to improve the display, whereas in a 3D display, the node is actually placed at new coordinates, i.e. physically moved. Moving a node may imply the moving of a link, see below.

Node deletion would take place by selecting a specific node and activating the "Delete Function". This function would firstly determine whether it is "physically" possible to delete the node, i.e. if the node is deleted, would the integrity of the model be in jeopardy. It would then determine if the statuses of the node allowed it to be deleted. If both checks are affirmative, all occurrences of the node are removed from the system, along with all directly coupled attributes. If either condition was not met, the node would not be allowed to be deleted until all requirements had been met.

Adding data to a node, and manipulating the data attached to the node can take place in a number of different ways. For example, simply moving a node in the 3D model would change its X,Y,Z coordinates and could change the Structure in which it finds itself (i.e. the attached Structure Code). If such changes were done via a text based screen form, the next time the 3D model is displayed, these changes would be apparent. A node could be "allocated" to another parent node by simply "picking" it up and "dropping" it onto the new parent node. Most other data would be manipulated via text based screen forms.

### 5.3.1.2 Links

Link creation can take place by manual action or links can be automatically generated.

To create a link manually, the appropriate menu option is activated and a "Link Creation Details Box" would be displayed. The ID Codes of both end nodes will have to be supplied as well as an ID Code for the link. Certain default data will be created along with the link (e.g. default length = 92m, the average length of cables on previous similar power stations). The create routine will have to verify that the node ID Codes are valid and that the Link ID code is unique. An alternative graphical way would be to place the mouse pointer on one of the nodes, hold the button down and drag the pointer to the other node and release the button. The same "Link Creation Details Box" will appear, except that the node ID codes (as well as directional information) are now already supplied and all that needs to be supplied is a Link ID Code. If the system is supplied with a set of "Link name generation" rules, even the Link Name need not be entered. In this way the system takes care of all lower level detail.

Cables for certain pieces of equipment can be automatically generated. For example, the required cable configuration for a standard switchgear can be obtained from the relevant Reference tables. A switchgear has a Type, the Type has a given configuration of cables. For example: a switchgear of Type "11AA/03" requires cable configuration "A4". Cable configuration "A4" consists of one 40 core process control cable Type "XYZ", one 2 core Field trip cable Type "ABC", one Motor Thermal Trip cable Type "ABC" and a 3 core power cable, type unknown (to be determined by sizing). As soon as the switchgear is supplied with its Type, it is possible to automatically generate all links associated with that switchgear. If the destination of the cable is known (as is the case with the power cable) then the link can be automatically "coupled" to the correct node. All other nodes are coupled to the FLOATING node. (Refer to "Special Nodes and Links" (4.2.5) above). As soon as the correct destination becomes known, the link is "un-coupled" from the FLOATING node and coupled to its correct node. Link creation can be left until the Macro level of connection has been finalised for a section of the design. All required links can then be created in one batch. The automatic creation of links is limited in that it is based on macro-connectivity. As a result, in-between nodes would have to be inserted manually. (Refer to the "Graph based Node-Link Model" (4.2.1) above and to Figure F4.2.04).

If a node is incorrectly coupled to another node via a link, it must be possible to "un-couple" the link from one node and "couple" it to another. Textually this can be achieved by querying for the link then replacing one node ID Code with another in the "Cable Details Box". Graphically it can be achieved by selecting the link (in which case display attributes highlight "handles" for the link itself and for either end. To un-couple and re-couple the link, the pointer is placed on the end "handle", the mouse button pressed down, the handle is dragged to its new location and the button released. The underlying software takes care of all database manipulations required.

Power cables need to be sized in order to determine their Type. One of the important factors determining cable size is length. Volt drop along the length of the cable must be within an acceptable limit. Initially a cable is sized at a default length to get an estimate of Type. The cable is then routed using this Type. The physical properties of a given Cable Type would determine whether a rack is suitable or not. Once an estimate route is found, the route length and thus the cable length, should be more accurate than the default length. The cable is then re-sized using the more accurate length. If the Type changes, the route is queried to determine whether it is still suitable for the new Type. For example, if the new Cable Type has a larger diameter, it may over-fill a section of the current route. If the route is no longer suitable, the cable will have to be un-routed and re-routed. This iterative process continues until the optimum solution is obtained. One of the interesting features of the CEEDS Cable Sizing algorithm is that cables are optimised on cost and not just technical criteria. Refer to Appendix A5.3 for details on the Cable sizing algorithm used in CEEDS and on some factors still requiring investigation.

Cable Routing involves finding the shortest suitable path between the physical location (X,Y,Z coordinates) of the end nodes along the Racking Network. The un-routing facility removes a cable from a route and adjusts the rack filling accordingly. Refer to Appendix A5.4 for details on the Cable Routing facilities used in CEEDS.

Currently, the sizing and routing facilities are manually "prepared" and run. Once the design process is better understood and controlled, the iteration described above could run automatically.

The success of cable deletion depends on the status of the cable as defined in the Design Process Model. For example, if an instruction has already been given to a contractor to pull a certain cable, or the cable already physically exists on the plant, then that cable cannot be deleted. If a cable is able to be deleted, the deletion routine must take care of all low level activities. For example, if the cable was already routed, it will first have to be un-routed before it can be deleted. otherwise the rack % filling is compromised.

#### 5.3.1.3 Importing and Exporting of Main Data

A CAE system involved in the design of large multi-discipline and multi-contractor projects cannot operate in isolation. There may be large portions of a design that are done externally to the CAE system. In other words, the source of data may be widely spread and the CAE system must be in a position to import such External Nodes and External Cables data. The system must be in a position to check for minimum data requirements and for duplication with Objects already existing within the main system. If the Contractor supplying the information does not submit a schematic drawing of some sort, then sufficient "Indication of environment" information must be supplied with an object. This is so that the designer using the main CAE system is able to correctly couple up these external objects.

CEEDS currently supplies external sources of data (e.g. contractors) with a PC based data capturing facility that is an inherent component of the complete CEEDS system. Data captured on such "satellite" systems are supplied to the Main system via magnetic disk. The format is obviously compatible with the Import routines and all other necessary routines to compare and verify, log delivery, etc.

The system should also be able to export data for use by other systems. On CEEDS, exporting falls into two categories. Data can be exported from the system in a format suitable for direct use by the PC based satellite systems. This is so that contractors can have their data "handed" back to them for updates. These updates can then be re-imported. The second category actually falls within the General Query Facilities. Any data can be exported onto magnetic media in any possible format. (The same does not necessarily apply to importing data. Facilities for importing random formats must be created on an individual basis. They are not an inherent part of the system. Only the data that is compatible with the system's data structure could possibly be imported).

#### 5.3.1.4 Graphical Query capabilities

Text based queries are described in "Utilities" (5.6.1) and elsewhere. A few graphical manipulation features that would be required for graphical based design are considered below.

Firstly, it must be possible to display only the specific area of interest. With very large designs, it would be almost impossible to present all detailed information at once. The "specific area" must be totally flexible. For example, a designer may only be interested in the Medium Voltage Distribution for Unit 1 of a power station, or a specific board and all its power related details, or all LV boards, but only showing Process control cables, no power details, etc. The "scope" of display must be able to be established by setting level of detail as well as specifying which equipment is to be displayed.

The display must be allowed to be added to (inclusion) or removed from (exclusion). The inclusion function will present flexible query facilities so that the user is able to select whatever equipment he wants by whatever criteria he wants. Exclusion is achieved by selecting the required objects on the display and "excluding" them. Note that exclusion is not the same as delete. Exclusion simply removes an object from the display. Selection is achieved by the "point and click" method, by surrounding the desired objects with a flexible "frame", or by using the flexible query facilities.

Hierarchical (nested) nodes must be capable of being "expanded" to show their internal nodes or "contracted" to hide internal detail. This idea of Expansion and Contraction can be taken to a higher level. For example, a board and all the loads connected to that board should be able to be contracted (or collapsed) into a single "temporary pseudo" node which represents the entire load of the board. The opposite should also be available, i.e. the expansion (explosion) of a "bundled" node into its constituent pieces. In this way a designer is able to concentrate on specific data without being swamped in unnecessary detail.

Other standard graphic facilities such as zoom, pan, fit screen, windowing etc. should be available. Standard 3D display facilities such as isometric views etc. should also be available. Refer to [SC 15] for details of such facilities.

Liberal and flexible (customisable) use should be made of display attributes to enhance visualisation capability.

### 5.3.2 Design Verification Checks and Model analysis

Verification checks can be placed into two broad categories:

- Those that ensure that the necessary level of data integrity is maintained in order to prevent software problems at the database level. These verification checks MUST be software enforced and are not able to be by-passed; and
- Those that ensure design and design process integrity. It would be desirable for some of these checks to be software enforceable (e.g. a cable cannot be deleted once an instruction has been given to a contractor to pull the cable). While other checks would be counter productive if enforced, by dictating that too much must be known about something before it can be inserted into the system.

There is no clear line between these two categories. The one extreme could be called "Data Integrity checks" and the other extreme, "Design Analysis".

There is also a delicate balance between how much "on-line" (data entry) verification is used as opposed to "batch" verification (after bulk entry).

- On-line verification slows a system down, but detects errors as soon as they are made. They also tend to be simple checks because complex checks would take too much time and make the system unusable. Obviously the computer environment plays a factor in this speed/verification tradeoff.
- Batch verification on the other hand has no effect on system speed as they can be run during low usage periods. They can thus be far more comprehensive than on-line checks.
- Another factor is to allow for flexible iterative design, therefore having as little on-line verification as possible, but the user must not be able to place the main backbone of the model in jeopardy.
- "On-line" verification is also divided into "Hard" and "Soft" verification. Hard verification will totally prevent an action if it is incorrect, while soft verification will only give a warning, but will not prevent the action.

#### 5.3.2.1 Verification Checks: (Data Integrity checks)

The system should be able to take care of itself regarding data integrity. In other words, if it is attempted to delete a node, the system should be able to determine whether it is possible or not. If it is possible, the system must then carry out all the necessary actions without having to rely on the user to remember to carry out any further actions, that if not carried out, could leave the system in a state of compromised integrity, which in turn could lead to catastrophic failure if not corrected.

Verification checks can be divided into Syntactic and Semantic checks:

- Syntactic checks are used to check on the validity of the format and range of data fields. For example, only digits, "+", "-", and a decimal point are valid for fields expecting a number value. Other cases may be limited to specific characters in specific positions. KKS rules dictate such a format specification. Refer to Appendix A4.1. Some cases require a range specification, for example a phase angle may only have meaning between 0 and 360, any other value would be invalid, or a Switchgear could only be "LV", "MV" or "DC".

- Semantic checks are more difficult to carry out. They are checks that would determine whether an action, or a connection, or some other criterion, **MAKES SENSE** in the given context. For example, deleting a cable from the model when it already exists in the real world does not make sense. A semantic check would be used to detect such a situation. Another example would be if a switchgear specifies that a 4 core power cable is required (3 phase + neutral), but the equipment it is supplying power to only requires a 2 core DC supply. Semantic checks are essentially used to carry out procedural and design validity checks.

Some mechanisms to carry out the above verification checks include:

- Checks integrated into the user interface (screen forms). These include:
  - hard coded range and format checks;
  - "This value must exist in a dictionary somewhere else" type checks. For example: the ID Code of a Node that is being typed in the "Cable Details" screen must exist in the system as a node otherwise that code cannot be accepted by the system. This type of check is often supported by a facility whereby the other "dictionary" can be queried, a value selected and this value automatically inserted into the relevant field, i.e. "lookup" facilities;
  - Mandatory data entry, i.e. the form forces entry of data in specific field(s) before the system can accept data from the form;
  - Dependency Checks: For example, if one field has a specific value, then another must be filled in according to a given criteria;
- Database level checks such as mandatory data and Unique indexes;
- Background activity checks, such as those that would be used by the Design Process Model (Refer to the "Design Process Model" (4.3) above). A transparent background activity is triggered to carry out a check or it is continuously monitoring activities and "pops" up with a suitable message and action when the conditions for a check are met;
- "What is in this table but is not in that table" type checks. The use of intersections, unions, and other set operations are useful for this type of check. For example, this type of check would be used to detect unrouted cables without relying on a status indicator for instance. This is achieved by producing a list of all cables in the Cable Details Table that are not in the Cable Route Table.

There are many batch type data integrity checks that can be carried out to determine the "healthiness" of the model. Refer to Appendix A5.5 for a list of current and proposed Batch checks for CEEDS.

### 5.3.2.2 Analysis

Design analysis is at a higher level of checking than simply checking for data integrity. It is not intended to delve into the detail of any specific analysis, as these are self supporting topics in their own right, but they do require mentioning.



These checks would include statistical analysis of the "goodness" of a design and numerous technical analysis activities including:

- Power Flow analysis;
- Dynamic Network Stability analysis;
- Power balancing and board loading analysis;
- Signal Path checks (for C&I equipment); etc.

Design "Goodness" analysis would consist of determining the percentage of data available, compared to what is ultimately required and also the confidence with which this data can be used. (Refer to Appendix A4.11). Missing data reports are also useful in aiding in the design process.

The current philosophy with CEEDS is that if a package that carries out an analysis function already exists, it would interface to that package rather than trying to re-invent the wheel. For example, a package called "Power System Simulators" (PSS/E) is a highly sophisticated load flow package used in the organisation. CEEDS must now supply an interface to this facility.

Mechanical CAE systems have their equivalent analysis checks such as finite element analysis, 3D collision detection etc. Civil engineering systems would have facilities such as volume determination of irregular shapes (e.g. a dam or a mine dump), the quantity of fill required to build part of a road, stress analysis of concrete structures, etc. Electronic systems would include, for example, analysis of thermal distribution on a printed circuit board (PCB) or within an integrated circuit (IC), simulation facilities, etc.

It is the ability of the system to carry out such analysis on the "model" that differentiates CAE systems from simple database systems or Computer Aided Draughting systems. If these analysis "routines" are fully interactive and are able to make full use of graphical features, then very powerful CAE systems are possible.

### 5.3.3 Codification

In any information system, liberal use is made of "Codes". One of the factors influencing the ease of use of a system, is the ease with which "codes" can be remembered and/or looked up. There are various schools of thought affecting coding.

- One extreme is that codes have no meaning at all and are only used by the database system to facilitate reference links to the details attached to those codes, and the role the codes play is usually transparent.
- The other extreme is that a code is made up in such a manner that meaning is inherent in the code itself and the code is created according to a set of rules.

The first method is fine if access to data is always via the information system, but as soon as the code may be used externally to the system, possibly within manual methods, then the second method is preferable. The philosophy adopted by CEEDS has tended to favour the second method.

Many codes used by a CAE system, will also be used by a number of other systems. If a set of codes must be accessible to a number of systems, then corporate wide dictionaries must be established. The responsibility for the contents of these dictionaries must be placed with a centralised body, not with any one of the major users. This is to prevent the modification of these centralised reference facilities in such a manner as to suit only one player, while disadvantaging others. (Refer to "High Level Development Strategy" (2.4) above for a development approach to centralised facilities).

These centralised dictionaries fall into two categories:

- Generic Reference Data and
- Project Specific Data.

Generic data would include Coding Systems Key Parts and Code Creation rules whereas Project Specific would include for instance, a "Master Record Index" of all currently identified valid and correct Object ID Codes.

#### 5.3.3.1 The codification of Object ID's

The system does not do code verification upon entry of codes because the exact code of a piece of equipment is not always known up-front. In this case "Dummy" codes can be used that may or may not adhere to a coding systems rules. These dummy codes can then be replaced by approved codes as soon as they become available. The administration of this function must be very carefully controlled to prevent the entry of the real code while the dummy code still exists. This could lead to the situation where there are two individual objects are identified to the system, but they are actually one and the same object. Another reason for not enforcing a specific coding system's rules is that the system may also be used to model plant that adheres to other Coding systems. The current philosophy is that the system allows the entry of any alpha-numeric Object ID code as long as it is unique. All the Object ID codes in the system can then be verified, in batch mode, against the rules of any given coding system.

On all new CEEDS projects, the codification of Object ID's is dictated by the KKS Coding system. (Refer to Appendix A4.1). For verification of KKS codes, use is made of a series of "Key Part dictionaries". Validation of a code against these Key Parts involves firstly checking the format of the code (i.e. do numerical parts contain only numbers, and do each of the alpha parts contain a valid character) and secondly, checking the hierarchical relationship between different parts of the code. This procedure cannot guarantee the correctness of a valid code, but it can detect codes that are definitely invalid.

A more sophisticated check involves the comparison of the English description of a plant item against what the code depicts the plant item to be. Key words are attached to key parts of the code. A "Percentage correlation" check can be run to detect whether certain keywords appear in the description or not. The closer the correlation, the more likely that the code and the description are correct.

### 5.3.3.2 Other codification

Object ID codes are but one facet involving centralised lookup and verification facilities. Other "Code Dictionaries" include:

- An Abbreviations Dictionary containing all official abbreviations and alternative abbreviations for use in creating short descriptions or where long descriptions cannot fit into the prescribed width. For example "MOT" or alternatively "MT" are used for "Motor". MTR would be an invalid abbreviation;
- A Contract Area codes Dictionary containing all valid contract areas into which equipment can be grouped;
- A Structure Code Dictionary containing all valid Area and Volume (Room) codes. This dictionary is actually the same as the "Object ID Master Record Index" mentioned above, because Areas and Volumes can also be seen as "Objects". (Refer to "Extensions to the model" (4.2.7) above);
- Documents are given an Identification code. All valid codes are registered in the "Drawing Information System" dictionaries. These dictionaries make up part of the documentation configuration management information system which is available on a centralised basis. Documents may have another identification scheme internally to a specific CAE system (Refer to Appendix A5.2) but as soon as they are intended to be used externally, they must be "registered" with the centralised facility and be given an official Document ID;
- Class and type codes are currently not available on a centralised basis but are being investigated on a corporate level. Refer to Appendix A4.1 for a possible scheme. The "dictionaries" making up class and type codes are actually the Type Reference Tables discussed in "Systems, Identifiers, Equipment Classes and Types" (4.2.2) above.
- Many other miscellaneous codes exist and are either stored in "Dictionaries" or are hard coded into the relevant software. If there are more than about five codes involved, and if there is a chance that a code may change or be deleted, or others be added, then it is advisable to create a dictionary. Software can then query the relevant dictionaries rather than having codes hard coded into routines.

### 5.3.4 Artificial Intelligence (AI) Technology

AI technology can play a vital role in future CAE systems. Design is an inherently innovative and creative activity. Conventional software technology is unable to "mimic" such behaviour but modern AI techniques can come a lot closer to carrying out tasks that appear to require intelligence. Some ideas are discussed, but a detailed investigation has not been done at this stage. There are a number of areas where it is envisaged that AI technology will play major roles.

- **Model representation:** Instead of using the relational model, an Object Oriented / Frame based representation could be used. This will give the facilities working on the model the opportunity to carry out far more sophisticated functions such as "inferencing". It is currently felt that the "Design Process Model" (4.3) may need a richer representation scheme than that offered by the Relational Model;
- **Integrity Checking:**  
Background Integrity checking programs can be made much more efficient by using AI search techniques. Instead of using "blind" exhaustive checks where much duplication and unnecessary work is carried out, thus slowing the system down, "Intelligent" checking can be done that eliminates or minimises this extra burden;
- **Design aids:**  
"Expert Systems" can be consulted to offer advice about design activities and embedded systems can actually automatically carry out other design activities. For example an expert system can be used to carry out automatic switchgear design. The selection of a suitable switchgear is based on certain rules, the available range of equipment, attributes of the load being fed and the environment in which it finds itself. The "rules" consist of a "first principles" part (deep knowledge), and an "experience" part (shallow knowledge). The deep knowledge is obtainable from equipment catalogues and "calculations". The shallow knowledge is obtained from experienced switchgear designers. If the rules are adequate, a large proportion of switchgear should be able to be selected automatically, thus relieving the designers of an essentially mundane task. The designers are now able to concentrate more on that smaller proportion that have special conditions. Design optimising within constraints is another area where AI techniques excel.
- **Design Analysis:**  
Far more sophisticated analysis of a design can be carried out using AI techniques. Things like "Reliability analysis" become possible.
- Other areas that have not even been identified yet...

## 5.4 Construction and Financial Functions

The natural progression from design, is construction. Construction moves a CAE system from the comfortable confines of a design office into the real world of dirty hands, deadlines, contractors, legalities, finance, etc. The requirements for a design aid system differ substantially from those of a Construction aid system. Construction system requirements have not yet been analysed in detail, but a few aspects that play a role have already been identified:

- **Construction oriented information:**  
The system should be in a position to provide "Bill of materials" reports, "Material Delivery schedules" etc. These would aid contractors in their buying of equipment and materials. Some equipment, special cables for instance, must be imported and long delivery times are a reality. The sooner estimates can be made about material and equipment requirements, the sooner contractors can start ordering and purchasing. Design information should be presented in a format useful for the people actually involved in building and commissioning systems.
- **Site feedback:**  
Certain important attributes of objects can only be obtained after construction has taken place. These attributes are used by subsequent processes, invoice control for

example, and must be obtained from site. For example, an actual cable may consist of a number of lengths all joined together to make up the design length. Each of these pieces may have come from different Cable Drums. This information must be fed back into the system to facilitate cable cost determination and Cable Drum Management. The actual length may differ significantly enough from the design length to warrant investigation. The reasons for such differences must be made known to the system to facilitate modification of the model to reflect reality. The model serves no purpose if it conflicts with the real situation. This site feedback can also be used as an indication of construction progress. The CAE system can be expanded to administratively support the complete commissioning and hand-over process.

- **Financial functions:**  
Involving control, budget variation analysis, financial reporting, etc. are all functions that can be supported by the CAE system.

CEEDS is currently mainly a design system with Site Interface capabilities limited to cable and racking attributes being fed back for invoice control. There is massive room for expansion in the areas of Construction and Finance. Those aspects are currently receiving attention.

## 5.5 Documentation Production facilities

Besides aiding in the design process, one of a CAE system's main functions is the production of Documentation for Design, Contractual and Construction purposes. The types of reports that can be extracted from the system are only limited by the data and data structures within the system. Technical, Managerial, Administrative, Audits, Bill of Materials, Construction aids and many more should be able to be extracted from the system in an easy and consistent manner. On CEEDS, three levels of documentation production facilities were identified. These are:

- General Reporting facilities (High Flexibility, accessible by all users);
- Standard Reports (Lower Flexibility, access is controllable to a certain degree); and
- Contractual Documents (Fixed Format, restricted access).

### 5.5.1 General reporting Facilities

The general reporting facilities are totally flexible and a report's format and content are decided on, created, maintained and administered by end users. Users are able to create and store their own report "programs" on an individual basis. These can then be run by the users whenever they so desire. Each registered user is allocated a small amount of personal disk space for the storage of such report programs (and for other purposes). The administration of this space is a user activity, not a system administrator function. (Refer to "Utilities" (5.6.4) below for details). All data, except confidential data such as passwords, are available for query purposes. On CEEDS, access to a flexible query product (Oracle's Query Management Executive, QMX) is available from every menu. In this way, the ability to query any desired data is never more than a few key strokes away. There is no need to continuously traverse large menu trees to find the necessary screen query facilities.

### 5.5.2 Standard reports

There are some reports that are useful to a wide variety of people and are produced on a reasonably regular basis. These reports are pre-written and made available to users via a "Standard Report" facility which is essentially a "select from the list" type facility. On CEEDS, most menus have access to a "Report List" via a "Reports..." menu option. Access to these menu options are controllable in the normal manner so that certain reports will only be accessible by certain groups of people. (Refer to "Security within an application" (5.6.1.2) below).

A Report is considered as an "Object" by the system. It has :

- an ID (Namely the File Name/File Type of the report program),
- a description (that would appear in the "Reports Lists" above),
- a set of instructions (that would appear in a separate "window" of the report list if requested) and
- an optional set of "hard parameters" that can be passed to the report program. (Soft Parameters would be prompted for by the report when it is activated, while hard parameters are set by the system administrator. Hard Parameters are seldom used but the facility exists if needed). Parameters enable reports to be flexible in the data they contain.

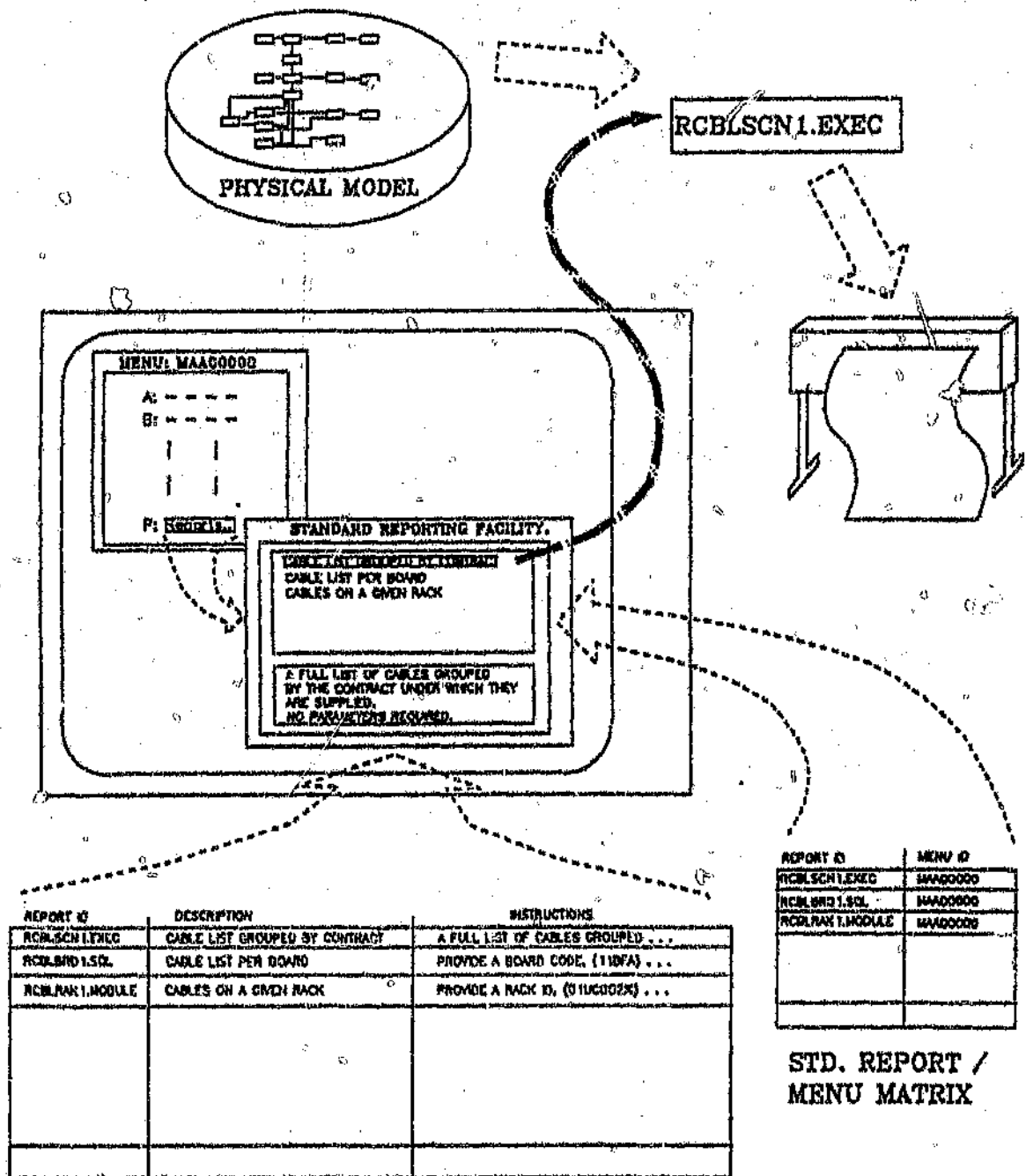
A table of Standard reports exists which contains all registered reports, with the above attributes. The names of these reports are validated against the software configuration management system. In this way it is impossible to register a report that has not already been registered with the software configuration system, thereby improving system integrity "Software Configuration Control" (5.4.3) below. An associative table exists in which reports are "allocated" to be available from one or more menus. In other words, a single report can be made available through the "Reports..." option on many menus, or can be restricted to only one. When allocating reports to various menus, the system ensures that the report exists and that the menus exist, once again ensuring system integrity. Refer to Figure F5.5.01. Query facilities exist on both the "Reports List" and the "Reports/Menu allocations list" so that a user can see what reports are available and where to find them. It then depends on the user's access profile as to whether the report can be run or not.

### 5.5.3 Contractual documents

This third category of documentation produced by a system is the most stringently controlled and has the strictest access out of all the categories. These documents act as instructions to contractors to carry out work. If there is inadequate control of the issuing of such documents, cost implications could arise as the result of re-work being done by the contractor because of incorrect information contained in these documents. Manual procedures must exist for the administration and distribution of these documents.

The "Contractual Documentation Production Facilities" provide the following:

1. Each document is required to be registered with the system;
2. Issue and Revision control are carried out;



STD. REPORT DETAILS TABLE

E.5.51

## Standard Report Facilities and Reports as Objects

Figure F5.5.01

3. Print logging is done;
4. Electronic archiving of previous revisions is carried out;
5. Re-printing of any previous revision of any issued document can be done; and
6. "Rollback" of an incorrectly issued (but not yet distributed) document. (Previous Revision numbers are re-instated and the copy of the document in the archive is deleted).

The last five issues are what differentiate Contractual Documents from Standard Reports.

#### 5.5.3.1 Modeling a document as an object

Contractual documents are modeled as "objects" in a manner similar to Standard reports. The only difference is that a Class of document is defined to which the software program producing the document is attached. Each instance of a document is identified by a combination of a Document Class and a Search Criteria (not a file name/type as with Standard reports). Other additional attributes include "Drawing Number(s)" for reference to other Documentation Control Systems (Refer to "CAE on an Engineering discipline level" (2.1.2) above). In the case of CEEDS, these Drawing Numbers are links into the "Drawing Information System" and the project specific documentation system.

For a detailed description of how CEEDS manages Contractual documents refer to Appendix A5.2. This is an edited extract from the Hypertext Documentation System being supplied with CEEDS.

#### 5.5.3.2 Configuration management philosophy

Configuration management of documents always poses interesting problems. Hybrid manual and CAE design systems have complicated the matter even further. This is because Manually produced documentation, which includes traditional CAD drawings, and MODEL BASED Integrated CAE system produced documentation cannot be viewed in the same light. Refer to Figure F4.6.03 for a comparison between traditional and model based configuration control (also known as Version oriented and Change oriented configuration control).

Traditionally, a document is the SOURCE of its own data, i.e. the data contained on a document is stored ~~in~~ the document. The document and the data it contains cannot be separated. The data contained on one document could also be contained on other documents. If such a place of data is to change, it is up to the manager of the documentation to firstly know all the places a piece of information may reside and secondly to arrange that the data is changed in all those places. Very strict procedures must be adhered to, to ensure data integrity, but it remains a manual task. There are no absolute enforceable mechanisms to ensure that all changes are made in an error free manner. Each time a number of changes are made, another revision of the document is issued and distributed. Changes made since the last revision are summarised on the document and once again there is no guarantee of correctness. Previous revisions can be reviewed by removing them from storage (if such a facility exists).



With a MODEL BASED CAE system, the model is the source of all data. Documentation is produced by extracting relevant data in a given format and placing it on paper. The document is no longer the source of data, it is simply a paper based representation of certain aspects of the model. A piece of information may once again appear on any number of documents, but if it is to be changed, there is no need for the change to take place on all documents. The change takes place in only one place: the source of data, namely the model. A Document has a dual nature in that it consists of the actual report and a "Change Summary" report. The actual report contains the technical data as at the date it was produced. The Change Report contains a listing of all changes that have taken place on data contained in the actual report between the dates of the previous revision issue and the current revision issue. Refer to Figure F5.5.02. This change data is extracted from Historic Data. Refer to "History Facilities" (5.6.3) below. When a document is produced, the actual report is archived in electronic format for later retrieval. In this way any previous revision of a CAE document can also be reviewed. There are facilities available for producing Micro Film versions of the documentation directly from magnetic media. Change reports can always be re-generated by simply supplying them with any two dates. This is as long as the History data is still on the system. In this way comparisons can be drawn between any two versions of a document.

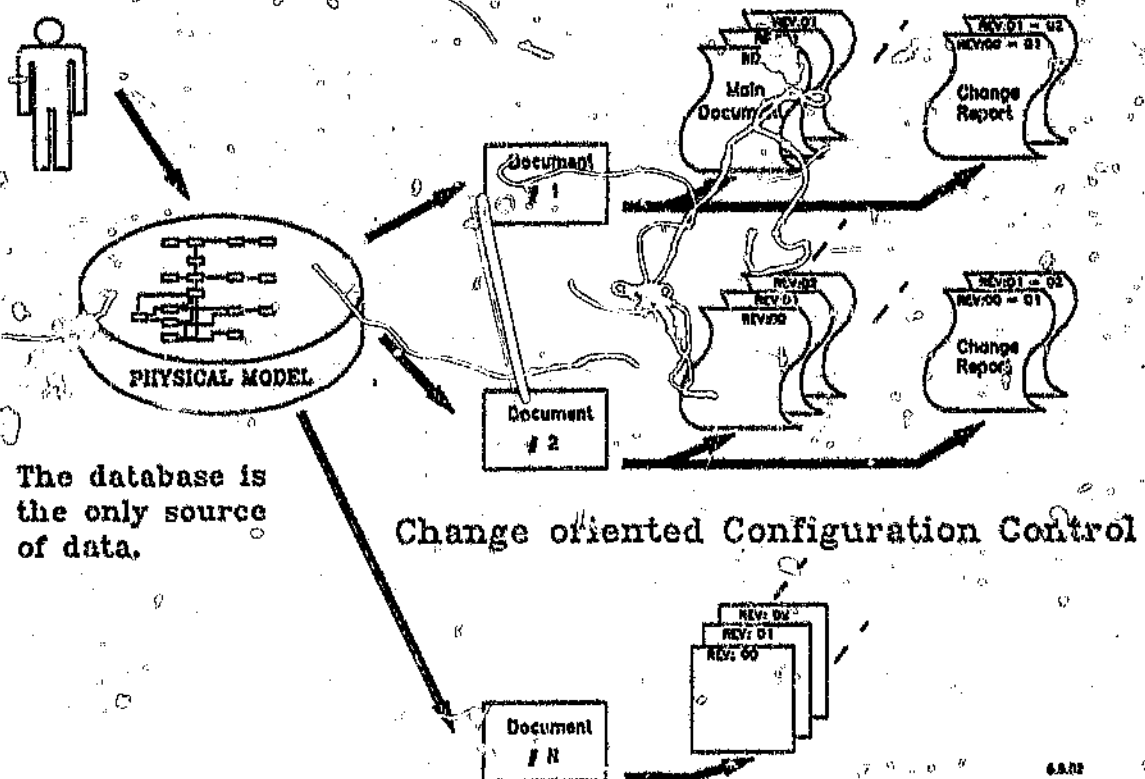
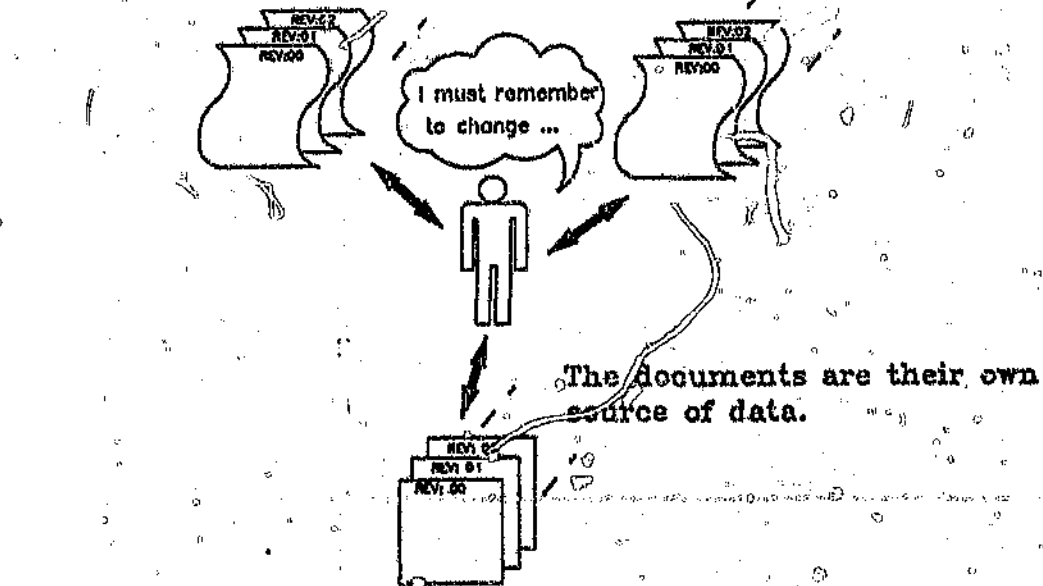
For details on how CEEDS carries out Contractual Document Configuration Control, refer to Appendix A5.2.

#### 5.5.4 Report Design

The contents and format of reports are very important if effective usage is to be made of the report. Firstly it must be decided whether the report is for specific and/or general use. General reports tend to contain quite a lot of similar data, but in a simple format, whereas specific reports tend to be fairly complex and extract data from a large number of places. Exactly what data appears on a document is also important because the reader of the document must be given adequate information for the document to be meaningful and non-misleading but not so much that important data is swamped with large amounts of irrelevant data. Layout and print attributes (such as colour and bold printing) play a part in highlighting important data and in giving the data a meaningful structure. For example, switchgear schedules are printed in vertical blocks to mimic what the physical board would look like. Efficient use of paper surface must be considered to prevent paper wastage.

Company standards must also be adhered to for all contractual and official documents. Printouts are prefixed with a standard "Administration" page which clearly contains the document title and any other required administration (such as names, approval signatures, etc.) Printouts must contain a common header/page numbering and a date for all pages. This is because if a lengthy printout is torn, the parts can at least be sorted. Graphical plots must contain the necessary title blocks, signatures, etc.

## Version oriented Configuration Control



Comparison between Version oriented  
and Change oriented Configuration Control.

Figure F5.5.02

### 5.5.5 Textual and graphical representations

The decision as to whether a graphical representation is required, or whether a text based report will suffice depends entirely on the nature of the information. (Arguments are similar to those presented in "Representation Schemes" above).

Currently CEEDS directly produces only text reports. Data for some reports can be exported to CAD programs to automatically produce a graphical representation. Facilities for automatic drawing generation are currently being investigated on the CAD based Racking Design Facilities. There is currently no control over CAD generated drawings containing data that is also in CEEDS.

The concept of automatic data extraction and drawing generation for graphical documents, followed by manual touch-ups to produce a production quality document are currently being investigated. The situation could arise that a merger of Model Based CAE configuration control and Version control of documents containing their own data, could arise. (See "Configuration management philosophy" (5.5.3.2) above). This would happen if Graphical documents were stored separately from Model Data but would be able to be automatically verified for correctness against model data. In this way transparency is lost but Flexibility is gained. It is foreseen that this amalgamation would be the interim situation until all documentation can be fully generated by the system.

## 5.6 Internal Functions and Utilities

### 5.6.1 System Security

There are various levels of security regarding the use of computer applications. These are:

- Physical access to computing facilities;
- Logon IDs and passwords to gain access to a machine/network;
- Logon IDs and passwords to gain access to an application;
- Access, within an application, to specific functionality;
- Access to databases with varying degrees of privileges; and
- Access to certain columns and/or rows within a database with varying degrees of privileges.

Security arrangements are very dependent on the computing environment being used for the CAE system. The following discussions are based on the platform that CEEDS is currently implemented on. Refer to "Current Platform" (6.2.1) below.

#### 5.6.1.1 Access to computing facilities and applications

Integrated CAE systems covering a very wide field of activities should provide easy and sufficient access to computing facilities. Access should be distributed to provide the best possible working conditions. Some teams may require to work in close unison and should be situated in close proximity (e.g. a communal terminal room), while others prefer to work in the privacy of their own territory. Both situations have merit and equipment distribution depends on available resources. Experience has shown that at least one set of centralised facilities is required where a number of people are able to access the system simultaneously, remain within earshot of each other and have the use of support facilities such as writing boards, overhead projectors, etc. These facilities are used for training, design reviews, close cooperation activities (such as design activities that affect one or more interfaces between discipline groups), etc.

Logon procedures to a specific machine or to a network should be simple, yet secure. A person should be a registered user with a personal password. The registration of users to access the computer facilities is usually handled by the administrators of these facilities (who are not necessarily the administrators of a given Application). Depending on the particular system, access to a particular application should again be as simple as possible. A person should also be a registered user of an application with an application ID and password. The registration of people as users of a given application are managed by the Application Administrators. (Refer to "Security within an Application" (5.6.1.2) below). These registrations should be on an individual basis, not a group basis. This is so that the activities of a individual can be logged and any malicious activities can for instance be traced to Joe Bloggs and not simply to Accounting. To get to the CEEDS application, a user is presented with a menu from which he is to select a particular environment. Once selected, he logs onto the specific environment by giving his individual ID and password. The system then presents him with a menu of applications for which that individual is registered. An application is then selected, in this case CEEDS. Logon to the application could be transparent using the already provided "Environment ID and Password" or could be explicit where an application ID and password must be provided. CEEDS uses the latter method.

### 5.6.1.2 Security within an application

Once inside CEEDS, all facilities are accessed via menu options. Access to a given option is dependent on the individual who has logged on. Each user is registered in a User Table with his User ID, a Password and personal details. Users are able to maintain their own personal details, but cannot query or modify anyone else's personal data. Each user is also given an "Access Profile" that determines which facilities may be used. The access profile is dependent on the function that the person carries out on the system. For example, a cable designer may have "view only" access to switchgear design data but may have full access to the cable design facilities and data. Each user is given a "Default Access" that allows them to View all (non-confidential) data and produce certain reports, but no access is given to any data modification facilities. The system administrator must determine what facilities would be required to carry out a given task. Access to these options is then given to the person delegated to carry out that task. This access profile is achieved by means of an associative table linking a User ID to any number of Menu Option ID's. (When editing a user's profile, the system ensures that the menu option exists and that the user exists. Integrity is thus ensured). For details on Menu Option ID's, see "The Man Machine Interface (MMI)" (5.2) above.

Report facilities are available whereby a user is able to print or plot his own menu access profile in tree form. A list of all users with access to a given option is another useful report. Refer to "Logging facilities" (5.6.2) below for details on "User Activity" logging. Query facilities are available on the User data but are restricted to Names, Telephone numbers and Location. All other data is considered confidential.

### 5.6.1.3 Database security

The degree of security built into access to the databases depends on the security philosophy and facilities provided by the Database Management System. ORACLE provides excellent and comprehensive security. [Ref ORACLE database administrators manual]. CEEDS has only made use of one layer of security at the database level. All facilities that are able to modify data, do so directly on the main tables. Query facilities are based on a set of "Synonym" tables that are based on the main tables but have Viewing privileges only, i.e. data cannot be modified via query facilities. The philosophy adopted was that if a person has access to a menu option in which given data can be modified, then ALL data accessible via that option is modifiable by that person.

If it is deemed necessary, very sophisticated security in which access can be limited to certain columns and/or rows within a given database, based on some or other criteria, with varying privilege levels, can be coupled to an individual. These privilege levels consist of privileges such as read only, update only, delete only, insert only, etc. or any combination of these. There is a proportional relationship between the sophistication of security and the administration required to maintain integrity when altering privileges and when adding or removing users. If there is no security, there is obviously no administration. Totally flexible security requires a great deal of administration. The developers of an application would need to determine the appropriate level based on risk, threat and audit requirements.

### 5.6.2 Logging Facilities

Any information system that supports activities that have contractual/legal and/or financial implications and where disputes may arise, must have sufficient audit facilities. Log tables can also serve a number of other functions such as data gathering for trend analysis, detection of system problems, the re-establishment of data inadvertently or maliciously incorrectly changed, etc.

The following logging activities were identified for CEEDS:

- Design Data Change Capture (Refer to "History Facilities" (5.2.5) below);
- External Data Import/Export logging;
- User activity logging; and
- Report and Contractual Document Print logging.

Logging activities should be totally transparent to the user.

#### 5.6.2.1 External Data Import/Export logging

CEEDS relies heavily on the gathering of information from other disciplines (especially Mechanical Engineering, who administer the mechanical designs where motors, valves, etc. are identified). Ideally, the electrical design system should be integrated with other design facilities but most mechanical design is done by external contractors and their data has to be imported<sup>1</sup>. PC based "external data capturing" facilities are part and parcel of CEEDS and are distributed to relevant contractors in order for them to capture their equipment onto magnetic media in a format specified by CEEDS. These disks are imported into CEEDS. The importing is logged and the date, disk number, contractor and user are captured. This is to prevent any disputes about when information became available on the system. The disk number is stored along with each piece of equipment on that disk as a "Source of data indicator". Equipment within CEEDS with a common disk number (or grouped by some other criteria) can be exported in the same format as the imported data. The exporting of data in this manner is similarly logged. (Refer to "Importing and Exporting of Model Data" (5.3.1.3) above).

#### 5.6.2.2 User activity logging

Those menu options which are classified as "Transactional" (i.e. those options where data modification can take place) are logged every time they are activated. The Menu Option ID, the user, date and time are logged. (See "Man Machine Interface (MMI)" (5.2) above). Besides providing a record of who is doing what on the system, this log data can be very useful in determining System usage profiles over a period of time, either grouped by user or menu option.

---

(1) Inter-organisational Integrated systems are even further into the future than inter-divisional CAE systems.

Analysis can provide many useful indications on user behaviour; for example, an up-swing in activity just before a design freeze date despite the necessary data having been available for a much longer period can be detected. The functions provided by the system can also be analysed to determine those that are used and those that aren't. On CEEDS, a log entry is also made each time a user logs on and logs off. In this way, the time spent logged onto the system can be determined (although this tends to be meaningless, as it is not a measure of time spent actually using the system)<sup>1</sup>.

#### 5.6.2.3 Report and Contractual Document Print logging

Currently General and Standard Reports are not logged. Contractual documents are logged each time they are printed. (Refer to "Documentation Production Facilities" (5.5) above). The Document ID, revision number, date and user are logged. This provides a full audit of contractual documents provided by CEEDS. The document logging on CEEDS could be improved. It may be desirable to log the number of copies produced and the number of pages used with each print. In this way overall paper usage can be determined and users with excessive paper usage can be detected and appropriate action taken.

#### 5.6.2.4 Administration of log data

All the above log data are stored in various database tables. The number of records could get quite large and periodic cleaning out would be required. CEEDS provides administrative reports that detail the number of records in these log tables. Based on these reports it can be determined whether cleaning out is required.

Cleaning out is done by supplying a date to the "Clean out Log Tables" facility. All records (for a specific table) older than this date will be exported to an automatically named flat file. These records are then deleted from the database. This flat file is then archived, and will eventually spool onto tape. The naming of the file is similar to that for contractual documents, except that another mechanism is employed to ensure uniqueness if the same file is created more than once within a day. This is because there is no revision number to ensure uniqueness as with the contractual documents. A standard report provides the administrator with the number of records older and newer than a given date. The choice of the specific date is a delicate balance between desire for as much on-line information and the cost of disk storage.

Facilities should be provided to retrieve and reload archived data back into their respective databases if complex analysis requires database type query facilities on the data. Otherwise viewing of the flat files should suffice.

---

(1) One aim was to prove an interesting observation, namely that the loudest critics of the system were the people who had used it the least.

### 5.6.3 History Facilities

If the changing of attribute values could have design, contractual or financial implications, it is advisable to provide audit facilities. CEEDS has a "History Facility" that captures changes in data into various History Tables. The following information is captured along with each change:

- ID Code of object on which the change took place;
- The name of the specific attribute that was changed;
- The previous value of the attribute;
- The ID of the user who made the change; and
- The Date and Time that the change was made.

Some change "rules" include:

- If the ID Code of an object is changed, the ID code in the History table is that of the new code, the attribute name is "ID Code" and previous value is the old ID Code.
- If an object is deleted, the ID code is that of the deleted object and the previous value is "DELETED".
- If a non-manually triggered function is responsible for changing of an attribute, the User ID is the name of the Function.

In this way a complete history is built up of all relevant data regarding a change.

There are a couple of factors influencing decisions as to what change capture facilities are needed, what data needs such facilities and what must be done with the data once captured.

#### 5.6.3.1 Facilities Required

Firstly, the change capture facilities should be able to be "switched on and off" in an easy manner. They must also be able to be applied to extensions to the system without requiring software modifications. This implies a "History Facilities Dictionary" of some sort. The History data must be able to be queried in as flexible a manner as possible to extract any desired information. It must be possible to archive old data and then recover that data if necessary. The facility should be totally transparent to the user.

The mechanisms used would depend on the computer environment available (Refer to "Computer Environment" (6.2) below). It would be preferable to have such facilities built into the database manager because then, irrespective of where the changes are made, they will be captured. If the facilities are built into the Physical Model Manipulation Functions (PMMF's), it would be possible to bypass them by modifying the data via facilities which don't have History facilities built into them. Another problem with having the facilities built into the "user interface" is the increased complexity of these facilities and the resulting additional maintenance burden.



CEEDS has had to resort to building the facilities into all data modification software because the necessary functionality was not available at a database level. (The latest pending version of ORACLE has Object-Oriented Database support and much enhanced functionality at the database level. These History facilities could then be built into the database management system, thus relieving all interface software of this burden).

Refer to Appendix A6.1 for details on CEEDS History facilities.

#### 5.6.3.2 What data to capture

What data changes need to be captured obviously depends on the nature and importance of the data, the difficulty with which it was obtained and most importantly, the severity of the consequences if it is changed and the old data is not available. On CEEDS it was decided that all technical attributes attached to an object, excluding those in reference tables, would be auditable. All data regarding Contractual documents is also auditable.

Reference information is not provided with these facilities. Batch type integrity checks are provided to detect and correct for changes in reference links. The system doesn't allow a non-existing reference link to be used. But, if a valid reference link is used and attached to an object, then the reference link is deleted, the object will not know. This situation will be detected by the above integrity checks.

All records in all databases have a "Date Created" and a "Last Updated Date" which are automatically maintained. These are very useful in detecting where and when changes have taken place and in synchronizing related information. If certain data was supposed to be frozen past a certain date and its Last Updated Date is at a later date, a conflict of interests could arise.

The same considerations need to be given to History data regarding the level of on-line data versus the cost of storage and retrieval as what is given to other Log data (Refer to "Logging Facilities" (5.6.2) above). Another factor affecting History Data is that it is needed to produce "Change reports" for Contractual Documents (Refer to "Documentation Production Facilities" (5.5) above). History data MUST remain in the database for at least the length of time required for one revision change of all documents accessing the data in question.

#### 5.6.4 Utilities

The utilities provided with a system depend largely on the computer environment and the desired level of functionality. Utilities discussed here are those provided by CEEDS.

##### 5.6.4.1 Flexible Query Facilities

The system must be relatively strict in the control of data modification facilities. Exactly the opposite philosophy should be applied to data query facilities, namely:

- there must be sufficiently flexible facilities to be able to carry out ANY possible query;

- users must be able to create, print and store fully customisable reports;
- facilities should allow for the creation and storage of "personalised macros" for the production of reports;
- they must be able to extract data into flat files in any possible format, and (at least) down load to PC's and therefore printers, disks, tape systems, etc.

Refer to "General Reporting Facilities" (5.5.1) above. Use is made of the QMX (Query Management Executive) ORACLE Product, which provides a function key driven, "graphical" query by example interface which is very easy to learn and use and yet extremely powerful. Access is also provided to the Structured Query Language (SQL) Facility for more complex reports [SC 16].

#### 5.6.4.2 Queries about the system itself

The system should be able to be queried in an easy, flexible and consistent manner for:

- finding a menu option that carries out a desired task and descriptions of what it does, i.e. query system functionality;
- the database structure for all tables and then all columns within these tables, i.e. query the model database structure;
- what standard reports are available, and where they are accessible from;
- who is responsible for training, administration, system development, maintenance, error logging, etc.; and
- a development history of when major updates were made and what they entailed.

#### 5.6.4.3 Communication facilities

Communication facilities are required for:

- the administrator to inform all users of modifications, intended down periods, happenings, etc. Such messages should be displayed when a user logs onto the system.
- individual users to communicate with the administrator and to each other. Inter-user file transfer facilities, or the e-mail system on the same machine as CEEDS are used for inter-user communication.

#### 5.6.4.4 Others facilities

Other general facilities include:

- Standard Operating System File Maintenance utilities for the maintenance of personal files, report procedures, note files, report data, etc.;
- A standard text editor (e.g. XEDIT) for writing procedure files, final touch-ups to reports, etc.;

- A general print facility that allows
  - selection from a list of report data files to be printed;
  - viewing of the report data to be printed;
  - printing any number of copies to any valid network printer;
  - the ability to set a default printer ID;
  - temporary override of the default printer ID;
  - a printer management facility to view printer status, view its queue, drop reports from the queue etc.

#### 5.6.5 Mass update facilities

There may be times when mass updates are required. These tend to be database type manipulations and not model related. For example, a contractor has defaulted and the contract has been taken over by another contractor. The contract code may then need to be changed for all equipment allocated to the previous contract. Another common update is in correcting descriptions, for example, in short descriptions some people may have used "MTR" and others "MOT" for Motor. A mass update can be done to change all occurrences of "MTR" to "MOT" for instance.

Currently such updates are done by appropriately trained people only because they are done through the "Back Door" by using Structured Query Language (SQL) commands directly on the database. Audit facilities are by-passed. (This is because the detection of changes are built into the normally used interface, not the database. Facilities are becoming available in very modern databases whereby audit facilities will be managed at a database level. In such a case it doesn't matter how the data is changed, it will be logged).

## 6 Software and Development / Implementation Aspects

### 6.1 Introduction

Most of the discussion so far has centered around the use of the CAE system and the system itself. This section discusses the designing and building of the CAE system. There are vast similarities between designing and building a Power Station and designing and building the CAE support system. Many of the engineering principles such as requirements analysis, user involvement, configuration management, etc., are all very similar. The major factors in the building of CAE systems are:

- the computer environment;
- other development resources; and
- a development methodology.

The development strategy is also important. Questions like:

- "Do we do the development in-house or do we hire a consulting company?",
- "Do we buy off the shelf products and attempt to integrate them or do we start from scratch?",
- "How much involvement from other islands is required at this stage?",

and so on, must be answered up-front. To go into "battle" without a well-thought-out strategy is to invite defeat before the battle has even begun.

### 6.2 Computer Environment

The more powerful the computer environment, the more powerful the CAE system can potentially be. A Computer Environment includes both hardware and software. Hardware features contribute mainly to speed, accessibility, graphics capability, storage space, interconnectability, etc. Software contributes mainly to functionality, user friendliness (usability), flexibility, modelling ability, etc. Much emphasis has been placed on the importance of non-technical issues. Technical issues also play a major role. For example, a serious hardware failure in a full production CAE system, can bring almost all activity in an organisation to a standstill. How many times have the words "I can't do anything, the computer's down", echoed through quietened halls. Software failures (bugs) tend to cause users to lose confidence in the system. A single report that gives incorrect answers can undermine many man-hours of good work. A comparison will be presented between the current CEEDS platform and the proposed platform to highlight the above hardware and software issues.

#### 6.2.1 Current platform

CEEDS is currently implemented on an IBM mainframe using the VM operating system and the ORACLE database system. Access is either via "dumb" terminals or via on-line PC's running terminal emulation software.

**Advantages:**

- Country wide access by ESKOM personal, including construction sites;
- Hardware and software maintenance are supported by full departments as well as the hardware and software vendors. Administration and maintenance of the mainframe environment is a massive task requiring massive resources;
- Support facilities such as "hot-lines" and "fault control" are available;
- Reasonably good security, full backup and recovery facilities run by operators, disaster recovery and contingency planning are covered;
- 24 hour access, with operators and support staff on standby;
- Very little capital expenditure on the part of users;
- The availability of development support staff and project maintenance teams;
- Very large storage capacity.

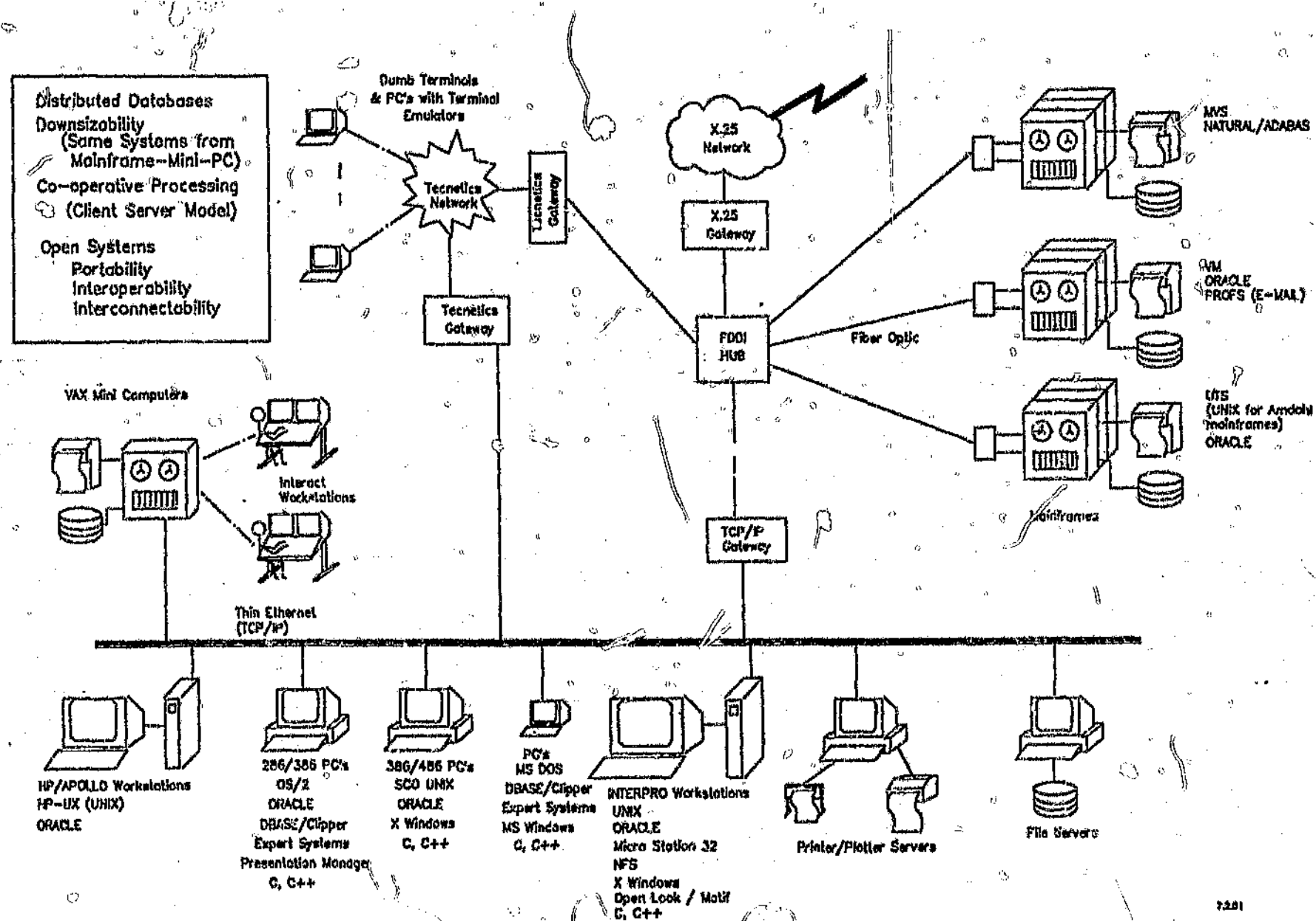
**Disadvantages**

- Running costs, equipment rental, CPU Time, storage costs,
- Users have very little say in what software is to be made available<sup>1</sup>, i.e. use the facilities provided or make alternative arrangements;
- the normal bureaucracy surrounding large systems;
- limited screen capabilities: Monochrome Block Mode text facilities only;
- limited printing capabilities (limited character set, no graphic characters are supported);
- The mainframe uses the EBCDIC character set and PC's use the ASCII character set. PC programs using mainframe data must be written with the differences in mind;
- NO graphics capability;
- Difficulty in connecting to other machines, e.g. the CAD VAX minicomputers and stand alone workstations;
- reasonably slow, especially at peak loading times;
- the availability of suitably qualified staff;
- there are many projects competing for resources and time, and politics becomes entangled in the process of determining priorities.

---

(1) ORACLE is written in "C" and was originally developed for the UNIX environment. The only suitable language on the VM Machine was FORTRAN. Halfway through development, some compatibility problems between ORACLE and FORTRAN were discovered. They could not be solved by using "C", but no "C" compiler was available, so ungainly work-arounds had to be used. The evaluation of a suitable "C" compiler was completed almost a year later. FORTRAN is also very limited in its ability to build and manipulate complex data structures such as trees and linked lists which were required by some programs. These programs were developed on PCs and data transferred between mainframe and PC, a very clumsy way of doing things.

Figure F6.2.01



Software and Development / Implementation Aspects

Page 39

Proposed new CEEDS Computer Platforms

25 x 10

32 x 10

### 3.2.2 Proposed future platform

The proposed future Computer Environment for CEEDS consists of a series of Ethernet LANS connecting the Mainframe, Workstations, standard and high performance PC's capable of supporting graphics, printing and plotting facilities. UNIX will be the predominant Mainframe, workstation and high end PC's operating system. All applications running on the UNIX environment should be running on top of X Windows and should have a consistent user interface (e.g. Open Motif or conforming to CUA) [SC 09,11,13]. Smaller PC's should be running under Microsoft windows (DOS) or Presentation Manager (OS/2).

The idea is to use the mainframe as a data server in which all the model data will be stored. The high performance graphic workstations will be used for most design activities. PC's and terminals will play their normal role as mainframe terminals. PC's could double as low performance graphic workstations. The database will run as a fully distributed database over the network. X windows (and other standard products such as the Network Filing System (NFS)) will provide all the necessary facilities for running fully distributed applications across the network. Refer to Figure F6.2.01

One important factor is that Organisational standards must be established and all equipment coupling to the system must adhere to these standards. These Organisational standards must be based squarely on international standards such as those advocated by the Open Software Foundation (OSF), International Standards Organisation (ISO) and the Institute of Electrical and Electronic Engineers (IEEE).

Other factors include the provision of sufficient support infra-structures. Repairs, insurance, guarantees, replacement parts, hot-lines, maintenance fees, license fees, running costs, etc. must all be taken into account. It must be decided whether to lease or purchase equipment. Are costly maintenance agreements necessary, are software product upgrades necessary? etc.

This platform does not yet exist in ESKOM, but a number of moves are being made in the right direction. Ethernet LANs are being installed, workstations are being purchased, UNIX is being evaluated as a universal operating system, etc. The following are some perceived advantages and disadvantages.

#### Advantages:

- Very much more powerful facilities. The environment offers such rich and flexible facilities that few restrictions should be encountered;
- Necessary software can be purchased and loaded onto locally owned facilities without being required to carry out normal bureaucratic procedures associated with the mainframe;
- Local system administrators would experience far more autonomy in the management of "their" piece of the system.

#### Disadvantages:

- The chronic shortage of people skilled in such an environment;
- the massive learning curves experienced by people entering such a new and complex environment;

- If individual departments now own their own equipment, they must either establish a local support infrastructure, or enter into agreements with corporate wide support facilities. Such facilities are not automatically available simply because one is a user of the system;
- Security administration increases.

Plus all the normal advantages and disadvantages associated with centralised and distributed facilities.

Essentially, the proposed system advocated centralised data access using distributed functionality. Functionality is placed where it is able to execute best.

### 6.3 Development resources

The three main resource requirements are:

- Well trained people;
- the necessary computer environment (Covered in "Computer Environment" (6.2) above);
- Time.

Cost is inherent in all three.

#### 6.3.1 People

People with the necessary training can be obtained via a number of sources.

- In-house:

In-house training of suitable people is a long term solution that has a number of benefits which include:

- cheaper in-house expertise;
- loyalty;
- developed products have a more stable, secure life time;
- minor enhancements and modifications can be made at low cost.

Some difficulties are:

- often, people displaying the potential of being good software engineers, are promoted into management positions; or
- after receiving expensive training, a person finds himself marketable enough to demand remuneration packages that cannot be met by larger organisations, and the company loses skills.

It is also difficult to attract suitable potential for employment because of the great demand for software skills.



- Professional/consultational services:
  - as much skill is available as an organisation is willing to pay for. These skills don't come cheaply though;
  - Companies often have an Information Technology (IT) department (previously known as Data Processing (DP) departments) and use must be made of these facilities as they are often cheaper than external consultants.
- Academic and Research Institutes:
  - Universities and technical colleges are always on the lookout for projects. Especially those with industrial application and backing. Professional researchers with very high skill levels can be "employed" through universities to carry out parts of a project for which in-house skill does not exist and for which it would be too expensive to use consultants.
  - Organisations that have bursars at such institutes should make use of this potential resource.

#### 6.3.2 Time

The time required of users to be involved in development of the system and in training to use the system must be catered for (Refer to "Resource planning" (3.3.2) above).

The time required to actually carry out the work of creating software facilities is very often under-estimated. This is an inherent danger of software project management because there are so many unknowns that time estimates are very often way off the mark. Another danger that project managers must be aware of is the misconception that the more people thrown at a task, the quicker it can be done. This may be true for tasks like digging ditches, but complex tasks requiring a high degree of inter-communication have a optimum level of people, where after, the communication burden becomes counter productive with an increase in team members. [SM 07]

The requirements for specific facilities must be identified sufficiently in advance because, all too often, facilities are only developed when it is too late. The "We want it yesterday" syndrome must also be avoided to prevent developers being pressurised into carrying out "quick and dirties". This practice may solve immediate needs, but it leads to large scale maintenance problems and system degradation that could be very costly in time and money to fix up at a later stage.

## 6.4 Building and Maintaining the System

### 6.4.1 Design methodologies

Every experienced software project manager has developed his or her own styles and methodologies of going about developing software. The exact methodology decided upon depends on many factors such as:

- The level of expertise within a development team;
- The size of the development team;
- The size and nature of the project. (Very Important);
- The intimacy with which the developers interact with the users and the users' management;
- The cultural advancement of the users (in a computer sense);
- Whether it is an in-house or contracted out project.
- The availability of suitable CASE tools;
- The available computer environment;
- and many more.

A few alternative approaches are briefly discussed (See [SM 04,09,10,11,15,16,17,18,19] for more details on methodologies). There are two extreme approaches, with a continuum between. One the one hand, it can be attempted to establish detailed design specification before actually creating any software. The other extreme is to do all development by rapid prototyping. An optimum approach for a specific project would lie between the two extremes somewhere. The main factors in deciding where to pitch the methodology are, user capability (i.e. do the users know what they want), and the availability and sophistication of prototyping tools. The project manager must decide on the "best" approach.

The approach taken for CEEDS was slightly different to normal because the development was driven as a speculative venture. There was very little desire by users to get involved because of the normal resistance to change (Refer to "Non-Technical Issues" (3) above). Attempts at large scale Joint Application Development (JAD) sessions were abandoned very soon because the total scope of the project was too large to be covered in any one session and the number of people involved was too large. There was also little drive from the users to actually get something done. Most viewed the exercise as a waste of time and money. A new approach was utilised whereby key pro-active individuals were identified and very close working relationships were established between the development team and these individuals. A combination of establishing reasonably detailed requirements and prototyping for demonstration purposes was put into action. One of the difficulties experienced was that the nature of the task, namely design support, is actually an extremely interlinked task. There are so many dependencies that the "pieces" chosen for analysis and prototyping had to be reasonably large. Quite a bit of re-work on already developed sections had to be done to account for details discovered during the design of a later section.

Prototyping has some very important advantages. Amongst these are:

- Users are better able to understand and visualise proposed solutions;
- Some of the scepticism can be eliminated when claims by developers are actually shown to work. (This requires quite a high quality of prototyping though);
- New ideas are triggered in users, and user involvement improves as a result.

Getting users to set up their own detailed specifications is also a technique that encourages deep involvement and understanding on the part of the user. This can only work though, if the user has the desire to get the job done.

One last aspect that requires discussion is the use of CASE tools. On CEEDS a stand alone CASE Tool was used. A graphics based PC CASE package was used to do all detailed design. [SM 25] The design was then coded and implemented on the Mainframe. The PC based CASE tool was suitable for initial design. Once coding began, small adjustments had to be made in order to get the system working and design omissions were catered for directly on the mainframe. The CASE documentation and the real design soon started drifting apart. Despite all efforts, drift will occur, especially with a prototyping approach. The dual effort soon becomes quite a burden and the CASE documentation is abandoned till "later" as the systems "self-documenting" capabilities become usable. There are CASE products that are based directly on a database system, for example ORACLE has a CASE product that is able to eventually create the database structure and limited processes. The advantages of an integrated CASE tool is that the CASE tool is incorporated as part of the system. It should be able to be adapted to perform the necessary configuration management functions. In this way the system is largely self documenting and self configuring. The normal maintenance burden can be greatly reduced by using such strategies. After all, a CAE system is being built to support complex engineering design and construction functions, why not apply the exact same principles in supporting the function of designing and building the system itself?

#### 6.4.2 Coding, Implementation, Testing, Production and Hand-over

- **Plan of action and procedures:**

A relatively detailed plan of action must be established along with procedures for writing the software, implementing the system, carrying out the necessary tests, establishing user acceptance procedures, placing the system into production, and final hand-over to the users and the maintenance teams. These are all factors required by real life systems. Simply writing the software is only a small fraction of the total effort. A development team ignoring any of these aspects is placing the system in danger of facing rapid degradation, very high maintenance costs and possible failure.

- **Scope determination:**

One of the difficulties faced with the CEEDS project was the difficulty in determining and limiting the project scope. With each step for which analysis was completed, more steps were identified. The project thus had an expanding scope. This was tolerated to the point where a motion limit was placed on current development and all other identified areas of interest were noted down for further investigation upon completion of the limited scope. This limit was dictated by contractual obligations. The system was designed to enable a certain group of users (namely switchgear, cable and Rack designers) to fulfil their contractual obligations with the minimum of effort. Further expansion will take place in an evolutionary manner, building upon already laid down structures and providing integrated tools for more and more design, construction and financial functions.

- **Cost Justification:**

[SM 15]. A very important and very difficult development activity is the cost justification of all development. All too often benefits are intangible, such as:

- quicker query times;
- constant availability of data;
- the early detection of design errors;
- the ability to do far more up-front design;
- the ability to carry out more iterative refinements;
- the reduction of incorrect coding; etc.

It is difficult to preempt cost savings due to such benefits, but what can be done is to try determine what the lack of such facilities cost on previous projects. This overhead cost should be substantially reduced with the availability of CAE counter measures. For example, a recent study has shown that the amount overpaid on a previous project for which there was no cable invoicing control exceeded the total cost of CEEDS development by a factor of about 1.5. One can then postulate that such overpayment would be minimized with the availability of the system. The system has therefore totally cost justified itself on only one function, namely cable contract invoice control. But this is not a true justification because it can always be argued that the same situation may not occur again. Tangible cost savings can be attributed to things such as reduced paper usage, using cheap computer printouts rather than very much more expensive CAD plots for "schedule" type data, the ability to use cheaper hardware to get the same job done (namely low performance PC's and Terminals, rather than high performance CAD workstations).

#### 6.4.2.1 Generic and project specific standards

The procedures spoken about above are detailed in a set of "Standards" documents. With CEEDS, these were divided into Generic standards and Project Specific standards.

Generic standards are those conventions and procedures that would be applicable to any development project undertaken by a team. Aspects covered by such a document would include:

- Procedures to manage the modification of these standards. The amount of re-work that could result from a standards change must be taken into account;
- Design Procedures:
  - CASE Tool procedures;
  - Configuration management procedures:
    - File Registration;
    - Registration of common utilities (library functions);
    - Version control; check in / check out procedures;
- General requirements for the project Manual filing system;
- Implementation, testing and production procedures:
  - User acceptance procedures;
  - Performance monitoring procedures;
  - Tuning procedures;
- Maintenance procedures:
  - Error reporting and logging procedures;
  - Change/enhancement request procedures;
  - General requirements by maintenance teams for take over of the system;
- General requirements for a project's Manual filing system;
- Software source code standards:
  - Style;
  - Source code documentation;
  - Minimum error handling capability and input validation;
- User Interface guidelines;
- Data structure guidelines (e.g. criteria of when to use third normal form and when to compromise for performance gains);
- Definitions of terms / a Glossary (for example, what exactly is meant by "module", "table", "database", etc.);
  - A development abbreviations dictionary;
- General documentation standards;
- General principles of file naming, table naming and variable/column naming;

- Function key usage guidelines;
- Report layout guidelines;
- Screen layout guidelines;
- Backup and archiving philosophy;
- Training guidelines; and
- Minimum requirements for contingency plans.

Project specific standards are those conventions and procedures that would be applicable to a specific project. They are dependent on the development team, the computer environment, the availability of utilities and tools and the nature of the project. Aspects covered by such a document would include:

- Individual responsibilities and areas of expertise of development team members;
- Detailed index for the project Manual filing system;
- Specific details of file naming, table naming, variable/column naming, index names, etc. (Refer to Appendix A6.1 for examples);
  - A list of Special or reserved files;
- Function key usage standards;
- Detailed operation of the Compiling and linking procedures;
- Description of environments:
  - user names, disks, disk sizes, directories, etc.;
  - database users, database sizes;
  - off-line PC based environments;
  - CAD environments;
  - The split into development, implementation, training and production environments;
- Detailed report layout standards (headings, dates, page numbers, footers, report identification, etc.);
- Detailed screen layout standards (Screen naming, Menu layouts, the display of non-standard function keys, feedback messages, etc.) [SC 13];
- Format of error messages ("For Information", "Warning", and "Error" formats and attributes);
- Detailed backup and archiving procedures (during development and during production);
- Training requirements, methods and plans (for development staff as well as users); and
- Detailed contingency plan.

Both sets of documents must be fully configuration controlled to ensure that they are always up to date and that all team members are working according to the correct versions.

#### 6.4.2.2 Testing

Adequate testing of the system is very important. Users lose confidence in a system and the developers of the system, if too many "bugs" surface after testing. Initial testing by the developers must be extremely thorough. This testing mainly ensures that the system operates properly and is as fault free as possible, from a software point of view. There after, user testing must take place to ensure that the system meets their specifications. (This was a problem with CEEDS seeing that there were no official user requirements. Users only started actually using the system during implementation and the conversion from an old system. Modifications made to the system to cover problems detected during this period had to be dealt with quickly. The established procedures made adequate provision and few technical problems were experienced). Once user testing is complete, final polishing touches can be done and system Hand-over can take place.

Test procedures to manage re-testing after a major change must also be established. This would include a set of known test data. New functions would be carried out on this data. The effect of a new function on the test data must be known in advance. Check lists would then be followed to identify any possible integrity problems.

#### 6.4.2.3 Conversion of old systems to the new system

A very important factor that must be planned for well in advance is the transfer of data from any existing systems into the new system.

Factors to be taken into account include:

- Conversions can take a considerable amount of time;
- It is not always possible, or desirable, to run a new system and an old one in parallel. Especially systems whose major tasks include configuration control;
- Re-structuring of existing data to fit into new data structures can be quite a complex task and is not always possible. Large amounts of "new" data may need to be generated in order to meet the new system's requirements;
- Extreme care must be taken to not lose any data. This may be difficult if data structures and attributes differ too radically;
- The new system may enforce a higher demand for data integrity, which could lead to a massive burden on users if data from the old systems are not very sound;
- There is a shortage of trained users at the stage when they are needed the most; and
- New Reference Tables need to be "filled up" with reference data prior to the system moving into production.

Advanced planning can ensure that data on existing systems is brought as close as possible to that required by the new system. User involvement is imperative, because, even though data transfer would probably be done by the implementers of the new system, the data belongs to the users. Often decisions about specific data must be made based on the data itself. Only the owners of the data can make such decisions.

#### 6.4.2.4 Possible Development Scenarios

The project outlined in this report is a major undertaking probably approaching 20 man years. There are a number of possible approaches, each with their own advantages, disadvantages and characteristics. These include:

- In-House development;
- Privatization of a team within the parent company
- Hiring of consultants in an advisory capacity to support in-house development
- Commissioning of a software-house on a contract basis to carry out complete development
- Joint industry and academic development

##### • In-house development:

The choice of this option depends on the size of the organisation and the local software development capabilities. In a small, dynamic, profit oriented enterprise, this option would probably place too much strain on available resources. Such enterprises require a short term solution and would be willing to pay for it. They cannot afford long lead times to allow for learning curves and research and development work. Large organisations on the other hand often have their own Information Technology (IT) departments. The skills available within such a department should be adequate to manage reasonably large projects. The inertia present in large organisations would provide the tolerance of long development times. It is also possible to establish teams within the engineering disciplines with the dedicated task of establishing specifications and then implementing the system once IT has done the development. If such teams contain sufficient software skills, they could actually carry out prototyping which is handed over to IT to place into production. Maintenance of the systems could also be taken over by special IT maintenance teams. Within the engineering disciplines, teams would be setup for each major development island with one discipline acting in a coordination role to ensure that necessary interfaces between the islands are established and adhered to.

- (1) Within ESKOM, a possible scenario would be the establishment of a team for each major discipline (Electrical, Mechanical, Civil, Chemical and Transmission) and possibly having Systems Engineering acting in a coordination role. IT would act as a service function to all these teams. In this way, software consistency is enhanced because a single development team (IT) is responsible for all production systems. Regular interdisciplinary meetings could be organised to promote cross



- **Privatization of a team within the parent company:**

A modern tendency in large organisations is the privatisation of specialist functions into small spin-off companies. Most large western organisations suffer from bureaucratic inertia with many employees adopting non-productive attitudes. The type of people required to develop and implement high technology systems tend to be pro-active professionals. Such people often find it difficult working in the restrictive bureaucratic environment of large organisations. The formation of a private company with the express task of developing such systems is a possible way of creating the freedom required. There are problems with this approach though. Firstly, the members of such a team have to become very business conscious, thereby causing distractions. Secondly, a very close working relationship is required between the users of the system and the developers. This close relationship could be placed in jeopardy when users realise that former colleagues are now working for themselves. Thirdly, small private companies simply do not have the resources required to carry out large projects. Unless an agreement is reached between the private group and the in-house IT department as to the use of resources and the taking over of systems, private groups will find it difficult to manage large scale projects spread over a wide field of expertise.

- **Hiring of consultants in an advisory capacity to support in-house development:**

If in-house resources are insufficient to get a project going, suitable consultants can be commissioned to aid in the process of establishing the necessary in-house expertise.

---

pollination of ideas and to establish island boundaries and interfaces. ESKOM management is favouring the in-house development strategy

- (1) This option has been listed as a last resort by ESKOM management because of the poor return on investment experienced in the past.

- **Commissioning of a software-house on a contract basis to carry out complete development:**

This option is probably the path that will be taken by small companies that lack in-house capability. It is highly unlikely that large organisations with the necessary skills available would contract out to a software house. If resource problems are experienced, a first option would be an attempt to employ the necessary skills before contracting out. This obviously also depends of the level of skill and specialisation required. It would be easier to contract out for very high skill and specialisation requirements because there may not be long term employment opportunities for such skills within the organisation.

- **Joint industry and academic development:**

Those areas of a project requiring in-depth research could be done in collaboration with academic institutes. Refer to [SM27] for more details of such joint ventures. Industry's view point on this approach tends to be that they enter into such joint ventures more for the sake of helping the academic institutes out of social obligation that actually expecting some return on the investment of time and money. This situation is changing with the introduction of academically backed "Science Park" type institutes that carry out research, but with a business approach.

Detailed plans would need to be drawn up and the resources required to achieve the set time scales would need to be determined. If these resources are not available in-house, then various combinations of the above options could be used to achieve the end goal.

Refer to Appendix A6.2 for plans on future CEEDS development.

#### 6.4.3 Software Configuration Control

[CM 01-09]. Software projects tend to be quite complex and involve a large number of "components". Strict configuration management of these components is necessary to ensure that the system does not degrade too rapidly and to facilitate modifications, enhancements and new developments. As has been mentioned above, the principles according to which the CAE system carries out its configuration management of the model, can be applied to the configuration management of the system itself. [SC 02]. Essentially the configuration management system is inherent in the system, it is not stand alone. In the same way that the system enforces modification rules on the model, it also enforces modification rules to the modification of itself. Inherent advantages are that it is difficult to carry out un-authorised ad-hoc modifications and the system looks after itself to a large extent. In the same way that the model cannot be made too "tight" in order to facilitate iterative design, the system configuration management system cannot be made too tight. It must be possible though to query the system for any exceptions and software integrity problems. The exact detail of the software "model" is not dealt with in this report, but it is similar to the Physical Model already presented. "Objects" are divided into Classes such as Reports, Menus, Screen Forms, Contractual Documents, Files, Users, etc. "Links" are relations such as: a certain Report is allocated to a certain Menu; a certain User is allowed to use a certain Menu Option, etc.

## 6.5 System Documentation

Documentation is one of those aspect of a software development project that is crucial, yet often ignored. An important deliverable of a software development project is its documentation. The process of producing documentation should be the driving force in a project. The software that is implemented at the end of a project is virtually useless without adequate documentation. The methods of producing documentation are briefly discussed. Then the two main categories, namely

- User Documentation, (Including system administration) and
- Technical Documentation, are discussed briefly.

### 6.5.1 Methods of producing Documentation

Traditionally, documentation is supplied as a bulky set of manuals. These manuals often find their way to the back of cupboards, under piles of other similar manuals, stuck behind piles of dusty old computer paraphernalia, etc., and are read by only a very small percentage of users. The documentation methodology advocated here, for future CAE systems, differs quite substantially from the paper based situation just described. Except for a small information manual, describing the system in very broad terms, instructions on how to get onto the system, how to get it running and how to access the on-line documentation, there will be very little official paper based documentation.

The current system being used on CEEDS consists of a PC based Hypertext system. Any PC being used as a terminal has the ability to swap between the mainframe session and a simultaneous PC session. The terminal emulation software is started up, the hypertext system is activated and the user then swaps over to the mainframe session and logs onto CEEDS. He is now in a position to swap between the mainframe session and the CEEDS hypertext based documentation. Moving the documentation onto PC was the only viable solution, considering the severe restrictions on the capability of the mainframe. One advantage is that the PC documentation can be "packaged" and loaded onto the mainframe. All users are then in a position to "receive" the package onto their local PCs and "un-package" the documentation in order to run it. Distribution is therefore done by electronic means. If changes are made to the documentation, a new "package" is loaded and all users are informed by means of a logon message, that new documentation is available. This alleviates many of the problems of paper based documents such as printing, binding, physical distribution, the difficulty of maintaining the documents, distributing change sheets, etc.

There are many shortcomings to this system though mostly as a result of limited computer capabilities. Users who have "dumb" terminals don't have documentation facilities. The hypertext system is limited to the use of ASCII symbols, no graphics can be used because the swap over to the mainframe session cannot take place with the PC in graphics mode. There are currently no formal methods of making hard copies of sections of the documentation. Screen dumps can be made, but this is a limited facility. The hypertext concept itself has some disadvantages. It is designed specifically to be structureless. Herein lies its flexibility and also its tendency to disorientate first time users.

Some of the requirements for documentation accompanying future CAE systems are:

- The documentation must be an integral part of the total system. It is therefore always the latest version, and no distribution problems exist;
- The documentation and help facilities are one and the same thing;
- The documentation must be context sensitive;
- The on-line facilities should be hypertext based;
- There should be a structured index into the hypertext facility. This is to prevent the disorientation mentioned earlier;
- Text and graphics must be able to be intermingled. Animation should be used for complex explanations, if the facilities exist. Ultimately, one could have context sensitive interactive video as part of the documentation system;
- There should be a fully formatted printable version of those parts of the documentation that may be useful in paper form for independent reading. The facilities to print selected portions of this documentation must be a standard part of the system. If the printable documentation is stored separately from the on-line documentation, complete configuration control will have to be exercised to prevent any inconsistencies.

#### 6.5.2 User documentation

The following aspects should be documented and available to a user:

- How to access and exit the system, i.e. a "Getting started" chapter;
- How to use the documentation;
- Everything to do with the detailed operation of the system, Function key usage, how to use menus, how to produce reports, how to do queries etc.;
- The concepts upon which the system is based. Descriptions of the models and facilities are required;
- There should be tutorial material for users wishing to get going on the system by themselves. Formal teaching does not suit all users, especially busy ones;
- Handy lists of codes which are not explicitly stored in a database dictionary, i.e. a Quick Reference Guide. For example: a cable has an "Air/Ground" flag to indicate the predominant environment in which it finds itself. Acceptable codes are "A" - Air, "G" - Ground, "U" - Unknown);
- Conventions of usage. There may be a number of valid ways of doing something, but one way has been agreed upon, and all users should follow the laid down convention;
- Hints and tips of how to get out of jams. There are many small oddities that can take place on a system, especially one distributed across a large network. Experienced users and developers should be able to add to this list;
- Details of the underlying database structures for those advanced users wishing to carry out advanced queries;
- Detailed procedures to carry out certain design functions.

### 6.5.3 Technical documentation

This is the documentation that will be used by system maintainers and future developers. It must contain all the necessary documentation to allow a newcomer to carry out a successful modification or enhancement. Aspects that should be included are:

- Maintenance procedures. This is perhaps the most important part of technical documentation. Very exact, detailed and step-for-step procedures must be laid down about how one goes about doing a modification or an enhancement. The necessary authorisation hierarchy and administrative procedures must be stated in detail. The more software enforceable these procedures are, the safer the system is against un-controlled and/or un-authorised modification.
- CASE Tool documentation. This should provide complete documentation on data structures, data flow diagrams and process decomposition diagrams.
- Source code listings.
- Detailed database and data file structure listings.

Paper based copies of all documentation should be kept in fire proof safes in case of major disasters.

## 6.6 System Administration

The administration of the system is important to keep the system running smoothly. Even though a well written system takes care of a lot of its own administration, there are many activities that still require people to make decisions and carry out certain actions. System administration would initially be done by the developers until nominated users can be trained to take over. Typical activities would include:

- User Administration: Registration of new users, deletion of old users and menu access profile maintenance.
- System performance monitoring: Checking system speeds and reporting to the necessary trained people if, for instance database or operating system tuning is required.
- Error reporting: Users must have someone to which system problems can be addressed. The administrator in turn would contact the necessary people to take care of the problem.
- Database monitoring: The size of log tables needs to be checked periodically and data removed to archive if necessary.
- Data recovery: Data may require recovery from archiving or computer system failure may necessitate disaster recovery when the system is running again.
- Administration of backup facilities:
- Administration of any requested enhancements or modifications to the system.
- Informing all users, via a logon message, of anything effecting the system or themselves as users. For example, if new documentation is available, or the system will not be available for some period of time, etc., users must be informed.
- Carrying out any "behind the scenes / through the back door" fixups that may be required from time to time due to newly discovered "bugs" and/or untimely hardware failures.

The system should contain an Administrators log in which all administration activities are logged for audit purposes. For example, if a mass delete is requested, the person authorising the work must do so in writing and the action must be entered in the Administrators log.

## 7 Conclusions

The creation and implementation of large scale Computer-Aided Engineering systems is not an easy task. There are many players involved and the environments in which such systems are employed are very dynamic. Organisations sometimes change too fast for their information systems to keep track. Often the very introduction of such systems leads to organisational change which in turn forces the system to change. Such systems also tend to be very complex and very interlinked.

The success of CAE systems is very dependent on numerous non-technical issues. Management involvement is imperative and cannot be underestimated. Resistance to change is absolutely normal. Developers must accept the fact that their ideas are going to be rejected simply on the basis that they advocate change. There are techniques that can be employed to overcome these problems. Training is one such technique that plays a vital role. Training is the one method that overcomes many inherent problems of introducing high technology into an organisation. The establishment of adequate procedures is another necessary activity without which failure is inevitable.

A very sound modelling foundation which is based on mathematical theory is necessary. This ensures continued research into faster and more efficient algorithms. The model base chosen, namely hierarchical graphs, is a very powerful and flexible modelling tool. The same techniques can be applied to the modelling of electrical networks, racking networks, structural connection and software configuration.

Developers must attempt to prevent the complexity of the model from exceeding the capability of the surrounding Model Manipulation Functions. There must be sufficient functionality provided to carry out basic model manipulations. Viewing of the contents of the model must be very flexible. The contents and so (2D, 3D or textual) of a display must be fully, easily, consistently and intuitively customisable. Configuration Management principles must be supported and adhered to.

A sound strategy which includes absolute discipline and a methodical approach must be adhered to during development. Well documented and laid down maintenance procedures are required to enable a quick response to changing needs. Without very strictly enforced maintenance procedures, systems would tend to degrade to a point where maintenance and other costs associated with system degradation and unavailability become prohibitively high.

The organisation must be prepared to move with technology. The computer platforms must expand to utilise modern hardware and software. Standards in Graphical User Interfaces, networking, database technology, operating systems and languages must be adhered to. Modern Artificial Intelligence, simulation and modelling techniques have their place and must be investigated.

## **A2.1 CEEDS I flaws**

CEEDS 1 was the first attempt at developing an integrated system to support the Electrical Engineering design process. An unwritten law of software development states that a system is seldom "right" before its third version. There were a number of inherent flaws in the data structure and in the development methodology used, these include:

- The development was not run as a software development project. (Which tend to have their own special requirements). The normal associated consequences resulted.
- Management was not sufficiently involved. New procedures were never adequately investigated or established.
- The data structure was not normalised. This led to data duplication which in turn led to loss of data integrity. The structure was based on Output Documents, i.e. an Auxiliary Power Equipment Database, Switchgear Schedule Databases, a Cable Schedule Database, etc. There was no underlying model. Modification of data was inherently difficult to carry out. Equipment not appearing on these base documents could not be consistently handled.
- The system was not fully specified up-front and had no consistent data model. It thus "evolved" as new ideas and requirements came to the fore. These were then shoehorned into the existing data structure making data manipulation routines increasingly complex and difficult to maintain.
- Interfaces to other systems were not feasible by any means other than flat file transfer.
- The database technology was inadequate. Access times were too slow which forced the use of Fortran Advanced Access directly to the data files. This made the whole system very vulnerable to data structure changes and thus very brittle. It was also essentially a single user system that would lock its files as soon as a user accessed them, thus freezing all other users.
- The mainframe system was old, over utilised and un-reliable resulting in frequent loss of work.
- System maintenance and modification was extremely difficult because of the software environment, the database structure and a lack of documentation.



## A4.1 KKS and BSF Coding Systems

Source: ESKOM KKS and BSF Presentation Material.

### USE OF BREAKDOWN LEVELS IN THE KKS SYSTEM

SERIAL NUMBER OF BREAKDOWN LEVEL	0	1					2				3		
NAME OF BREAKDOWN LEVEL	TOTAL PLANT	SYSTEMS					EQUIPMENT UNIT				COMPONENT		
DESIGNATION OF DATA CHARACTER	G	F0	F1	F2	F3	FN	A1	A2	AN	A3	B1	B2	BN
NATURE OF DATA CHARACTER	A OR N	N	A	A	A	N	N	A	N	N	N	A	N

BREAKDOWN LEVEL 0 = TOTAL PLANT IDENTIFICATION

BREAKDOWN LEVEL 1 = SYSTEM IDENTIFICATION

BREAKDOWN LEVEL 2 = EQUIPMENT UNIT IDENTIFICATION

BREAKDOWN LEVEL 3 = COMPONENT IDENTIFICATION

#### BREAKDOWN LEVEL 1 AND 2

IS MAINLY USED FOR:

PLANNING AND PROJECT DESIGN  
FLOW DIAGRAM CODING  
ERECTION, COMMISSIONING  
OPERATION, SURVEILLANCE

#### BREAKDOWN LEVEL 1 TO 3

IS MAINLY USED FOR:

MAINTENANCE, STATISTICS  
SPARES AND REPLACEMENT PARTS  
PLANT BUDGETING AND COST CONTROL

SERIAL NUMBER OF BREAKDOWN LEVEL	0	1					2				3		
NAME OF BREAKDOWN LEVEL	TOTAL PLANT	FUNCTION					EQUIPMENT UNIT				UNIT COMPONENT		
DESIGNATION OF DATA CHARACTER	G	FO	F1	F2	F3	FN	A1	A2	AN	A3	B1	B2	BN
NATURE OF DATA CHARACTER	N	N	A	A	A	N	N	A	N	N	A	A	N

CLASSIFICATION

IDENTIFICATION

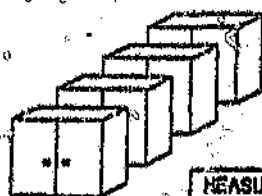
CLASSIFICATION

IDENTIFICATION

### FUNCTION BREAKDOWN LEVEL

CLASSIFICATION

IDENTIFICATION



CFB

MEASUREMENT CABINETS



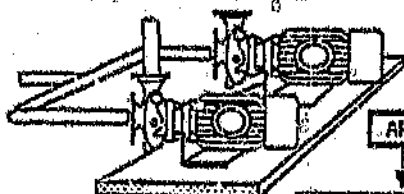
CFB | 04

MEASUREMENT CABINET | 04

### EQUIPMENT BREAKDOWN LEVEL

CLASSIFICATION

IDENTIFICATION



AP

PUMPING UNITS



AP | 002

PUMPING UNIT | 002

SERIAL NUMBER OF BREAKDOWN LEVEL	0	1					2				3				
NAME OF BREAKDOWN LEVEL	TOTAL PLANT	SYSTEMS					EQUIPMENT UNIT				COMPONENT				
DESIGNATION OF DATA CHARACTER	G	F0	F1	F2	F3	FN	A1	A2	AN	A3	B1	B2	BN		
NATURE OF DATA CHARACTER	A OR N	N	A	A	A	N	N	A	A	N	N	A	A	N	N

A NETWORK AND DISTRIBUTION PLANT

B POWER STATION ELECTRICAL POWER SYSTEM

C INSTRUMENTATION AND CONTROL EQUIPMENT

D (INSTRUMENTATION AND CONTROL EQUIPMENT)

E CONVENTIONAL FUEL SUPPLY AND RESIDUE DISPOSAL

F (HANDLING NUCLEAR EQUIPMENT)

G WATER SUPPLY AND DISCHARGE

H CONVENTIONAL HEAT GENERATION

J (NUCLEAR HEAT GENERATION)

K (REACTOR AUXILIARY SYSTEMS)


L STEAM, WATER AND GAS CYCLES

M MAIN MACHINE SETS

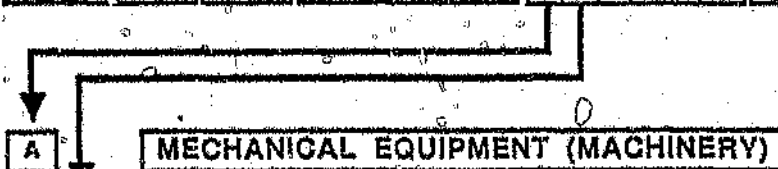
N PROCESS ENERGY PREPARING FOR USERS OUTSIDE

1008/018/AV/381

KP-38/E  
13/1

	ELECTRICITY SUPPLY COMMISSION	REFERENCE OPS 0014	KEY
	KEY PART		
	KKS POWER PLANT CLASSIFICATION SYSTEM	PAGE	OF
			REV.
L	STEAM, WATER, GAS CYCLES		
LA	Feed water system		
LAA	Feed water tank, deaerating		1
	from to		
		deaerating installation or feedwater tank inlet tank outlet incl. heating installation and vapour condenser	
LAB	Feed water piping system (excl. feed water pumping system and feed water heating)		1
	from excl. to excl.		
		feed water tank outlet boiler inlet header	
LAC	Feed water pumping system (excl. interconnecting piping between booster and main pump)		1
	from to		
		pumping system suction nozzle pumping system discharge nozzle	
LAD	Feed water heating system (HP heaters)		1
	from to		
		heater inlet heater outlet incl. desuperheater and drain cooler when integral part of heater	
LAE	High pressure water injection system (superheater attemperator spray system, HP bypass spray system)		1
	from excl. to excl.		
		feed water piping system user (e.g. superheater attemperator)	
LAF	Intermediate pressure water injection system (reheater attemperator spray system)		1
	from excl. from excl. to excl.		
		feed water pump extraction (take-off connection) branch of another system user (e.g. reheater attemperator)	
THE REGISTERED RECIPIENT IS RESPONSIBLE FOR DESTROYING ALL SUPERSEDED DOCUMENTS			

SERIAL NUMBER OF BREAKDOWN LEVEL	0	1					2				3					
NAME OF BREAKDOWN LEVEL	TOTAL PLANT	SYSTEMS					EQUIPMENT UNIT				COMPONENT					
DESIGNATION OF DATA CHARACTER	G	F0	F1	F2	F3	FN	A1	A2	AN	A3	B1	B2	BN			
NATURE OF DATA CHARACTER	A OR N	N	A	A	A	N	N	A	N	N	N	A	A	A	N	N



**A**  
 A A FITTINGS (VALVES, DAMPERS ETC)  
 A C HEAT EXCHANGER UNITS  
 A F ENDLESS CONVEYOR UNITS  
 A N COMPRESSOR, FAN AND BLOWER UNITS  
 A P PUMP UNITS

**B MECHANICAL EQUIPMENT (PLANT)**

B B STORAGE UNITS (VESSELS, TANKS)  
 B G BOILER - HEATING SURFACES  
 B R PIPING, DUCTS AND CHANNELS

**C DIRECT MEASURING CIRCUIT**

C E ELECTRICAL QUANTITIES  
 C F FLOW  
 C L LEVEL  
 C P PRESSURE  
 C T TEMPERATURE

**G ELECTRICAL EQUIPMENT**

G A SUB-DISTRIBUTOR  
 G S SWITCHGEAR EQUIPMENT  
 G T TRANSFORMER EQUIPMENT

SERIAL NUMBER OF BREAKDOWN LEVEL	0	1					2				3						
NAME OF BREAKDOWN LEVEL	TOTAL PLANT	SYSTEMS					EQUIPMENT UNIT				COMPONENT						
DESIGNATION OF DATA CHARACTER	Q	F0	F1	F2	F3	FN	A1	A2	AN	A3	B1	B2	BN				
NATURE OF DATA CHARACTER	A OR N	H	A	A	A	N	N	A	A	N	H	N	A	A	A	N	N

**K** **MECHANICAL UNIT COMPONENTS (PRODUCTION)**

K A VALVES, DAMPERS  
K G HEAT EXCHANGER  
K F ENDLESS CONVEYERS  
K N COMPRESSORS, BLOWERS, VENTILATORS, FANS  
K P PUMPS

**M** **MECHANICAL UNIT EQUIPMENT (AUXILIARY)**

M B BRAKES  
M G GEARBOXES  
M K CLUTCHES AND COUPLINGS  
M R PIPING PARTS

**Q** **INSTRUMENT AND CONTROL UNIT COMPONENTS  
(NOT ELECTRICAL)**

Q N CONTROLLER  
Q P MEASURING INSTRUMENTS, TESTING EQUIPMENT  
Q R IMPULSE PIPING

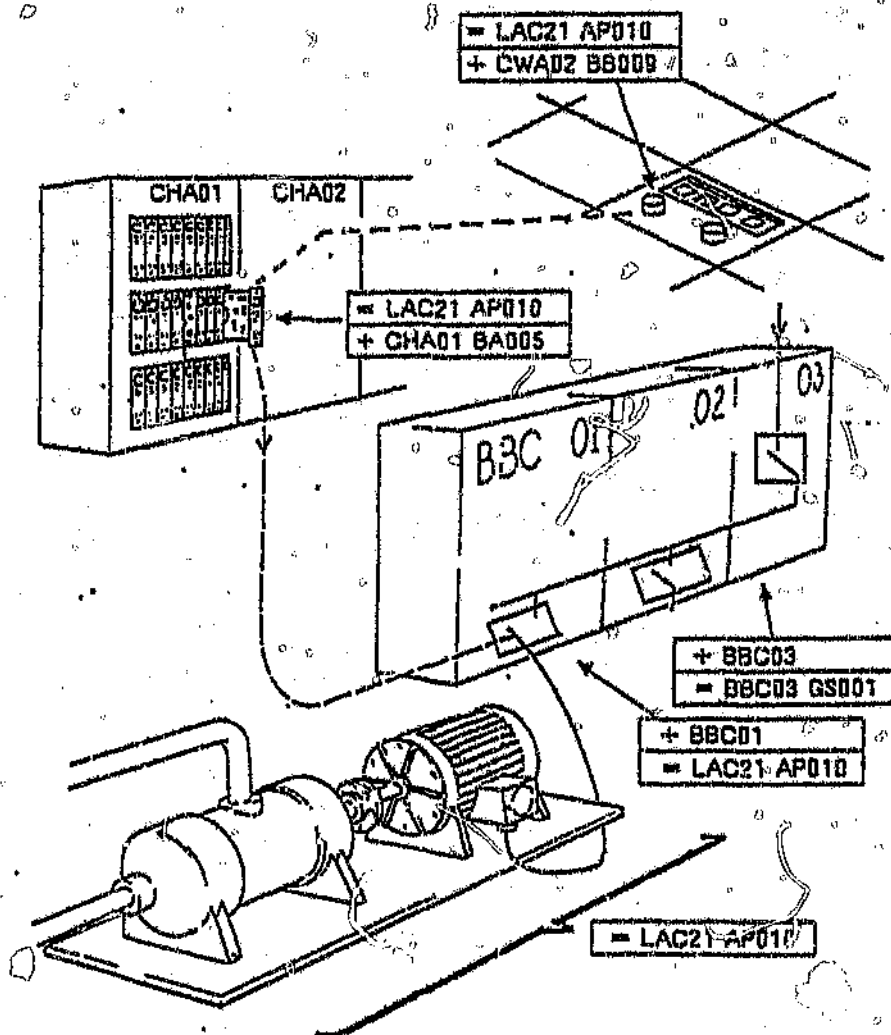
**I** **ELECTRIC UNIT COMPONENTS**

I B TRANSDUCER FROM NON-ELECTRIC TO ELECTRIC QUANTITIES  
I M ELECTRICAL MOTOR  
I S SWITCHES, SELECTORS  
I T TRANSFORMER

## CLASSIFICATION FOR A PUMP UNIT

**[-]** PREFIX FOR PROCESS BASED CLASSIFICATION

**[+]** PREFIX FOR POINT OF INSTALLATION CLASSIFICATION

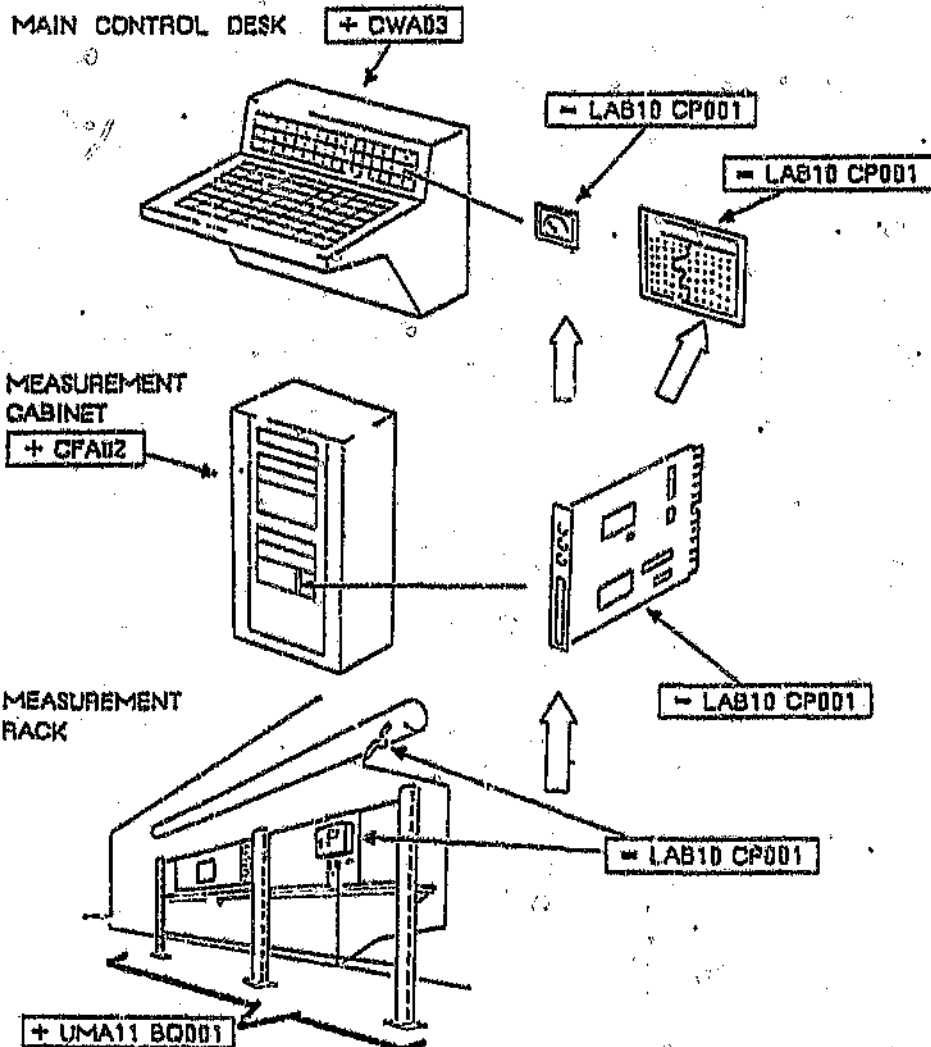




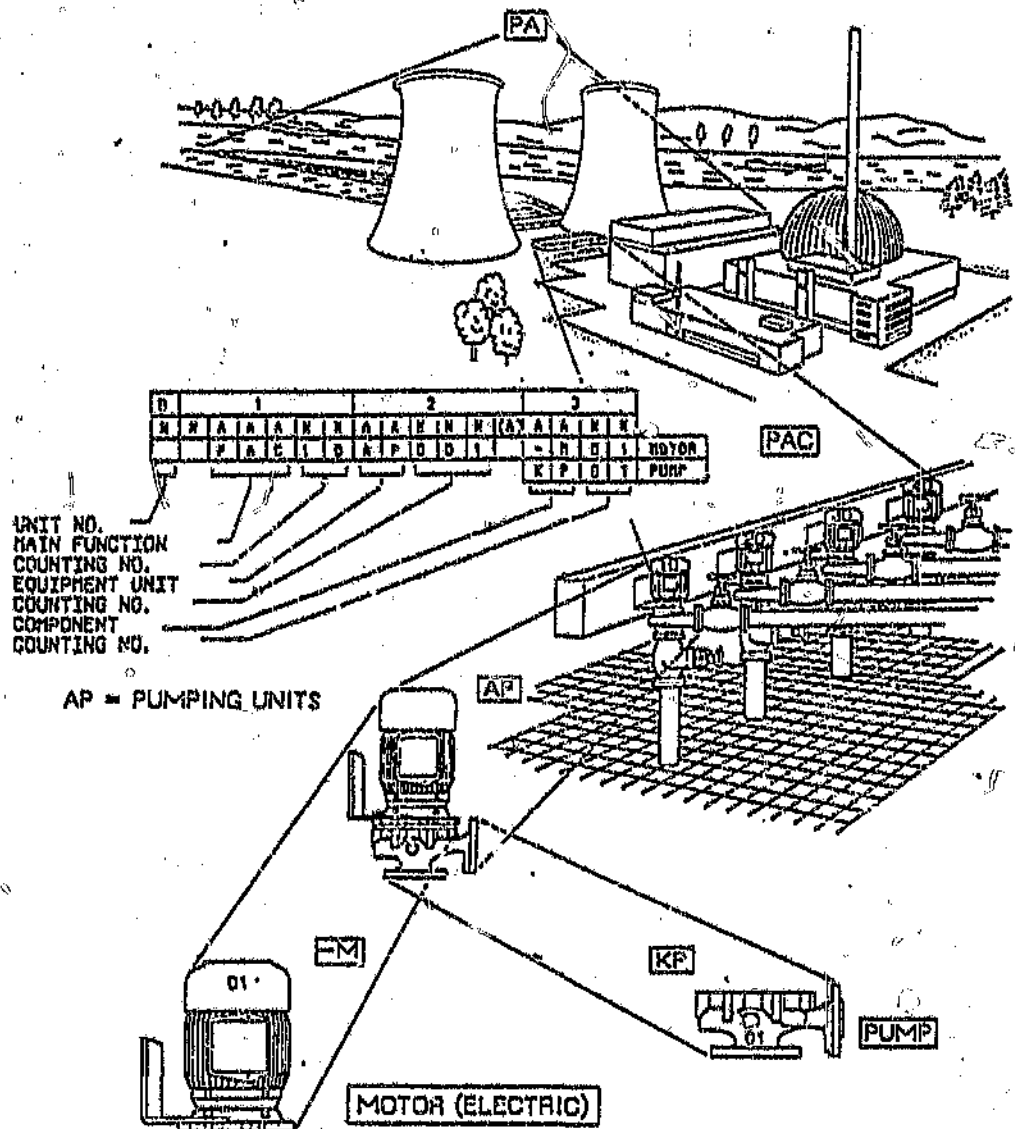


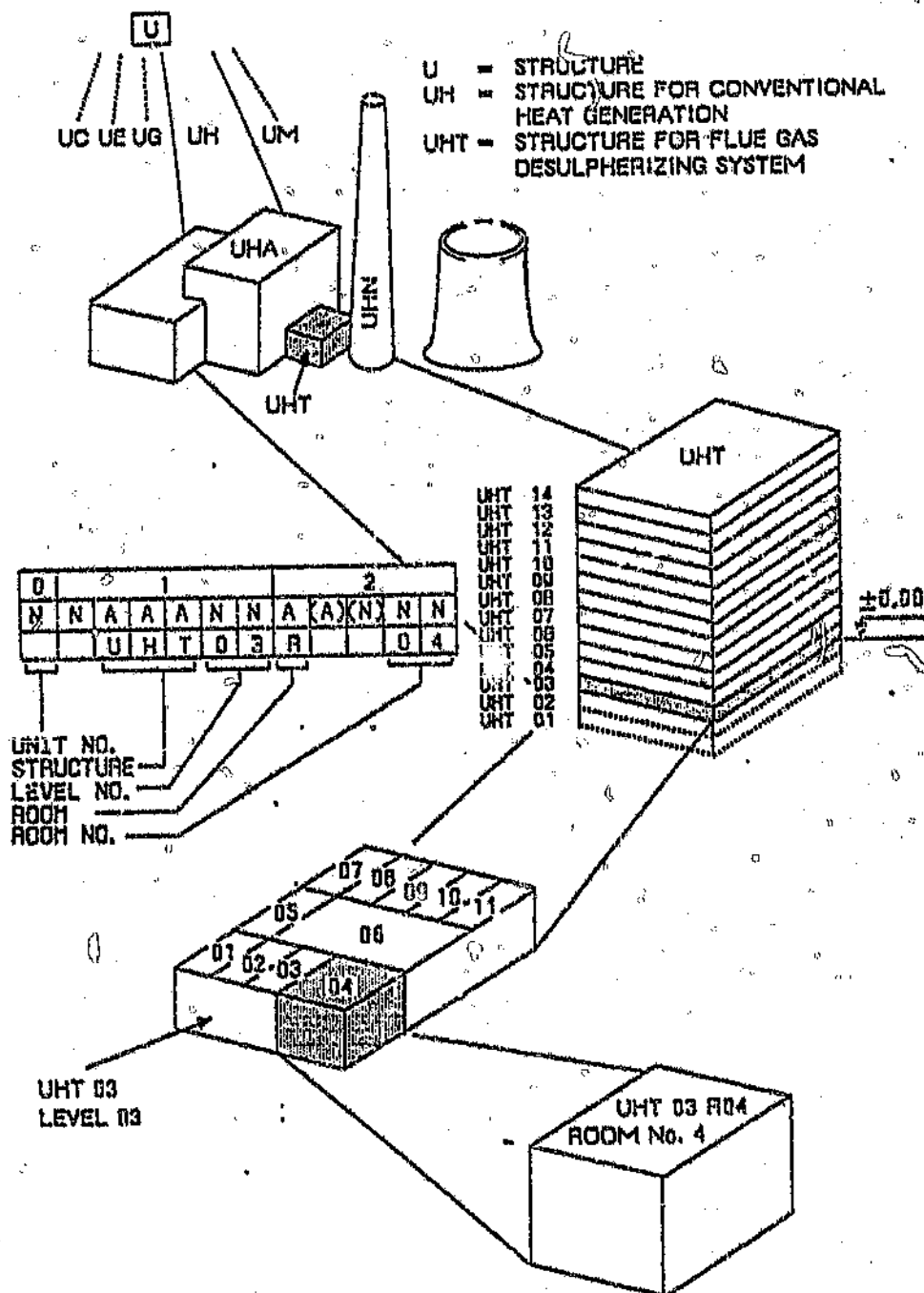
## CLASSIFICATION FOR PRESSURE MEASUREMENT

- |   |   |
|---|---|
| = | PREFIX FOR PROCESS BASED CLASSIFICATION         |
| + | PREFIX FOR POINT OF INSTALLATION CLASSIFICATION |



P = COOLING WATER SYSTEM  
PA = MAIN COOLING WATER SYSTEM  
PAC = MAIN COOLING WATER PUMPING SYSTEM



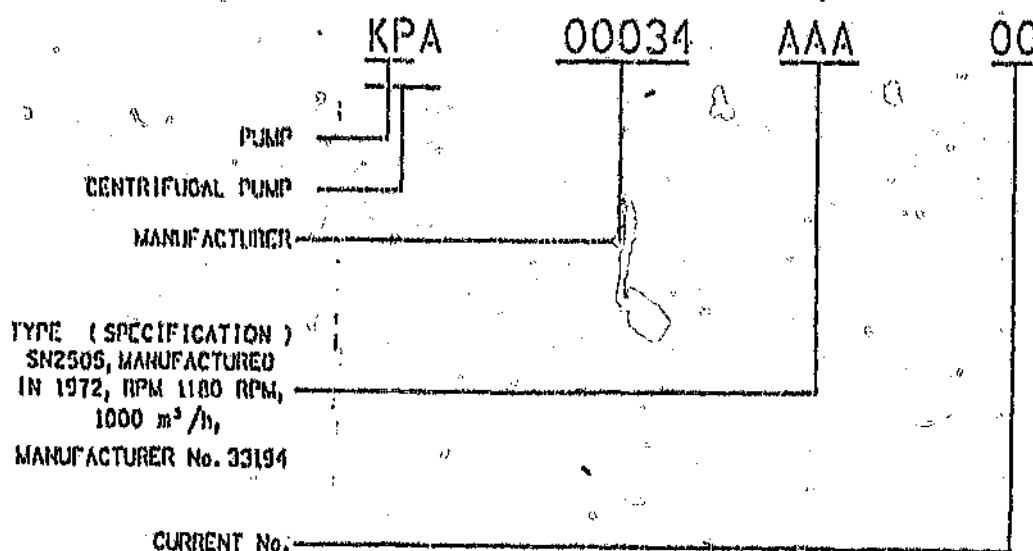


# WHAT IS BFS?

IT IS A COMPONENT TYPE CODE  
DESIGNED TO A USER REQUIREMENT

BREAKDOWN  
LEVEL  
DESIGNATION

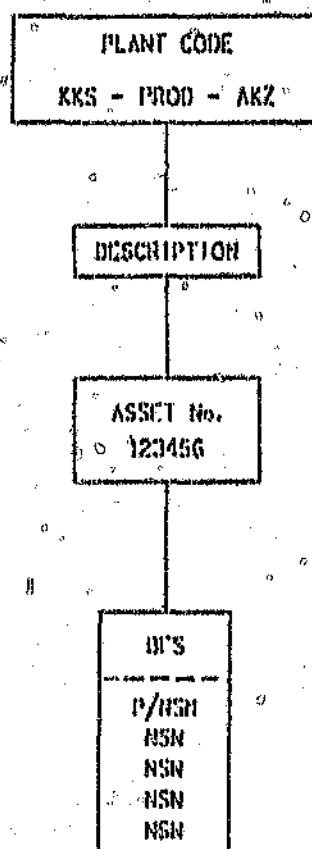
COMPONENT IDENTIFICATION CODE (BFS)			
1	2	3	4
UNIT COMPONENT "KIND"	MANUFACTURER	TYPE	VARIANT



## TYPICAL BFS BREAKDOWN

KM MIXER, STIRRED  
     KMA MIXER  
     KMB STIRRED  
     KMC RAKING MACHINE  
 KN COMPRESSORS, BLOWERS, FANS, VENTILATORS  
     KNA COMPRESSOR  
     KNB BLOWER  
     KNC FAN  
 KP PUMPS  
     KPA CENTRIFUGAL  
     KPB POSITIVE DISPLACEMENT  
 KT PURIFIERS, DRIERS, SEPARATORS, FILTERS  
     KTA PURIFIERS  
     KTB DRIERS  
     KTC SEPARATORS  
     KTD FILTERS  
 KU CONVERTERS  
     NO BREAKDOWN  
 KV BURNERS  
     KVA PULVERIZED FUEL  
     KVB OIL  
     KVC GAS  
     KVD GRATE  
     KVE COMBINED  
     KVF PILOT  
 KW WORKSHOP DEVICES  
     NO BREAKDOWN  
 KX STATIONARY TESTING DEVICES  
     NO BREAKDOWN  
 KZ SPECIAL MECHANICAL DEVICES  
     NO BREAKDOWN  
 M MECHANICAL UNIT COMPONENTS (AUXILIARY)  
 MD BRAKES  
     NO BREAKDOWN  
 MF FOUNDATIONS  
     NO BREAKDOWN

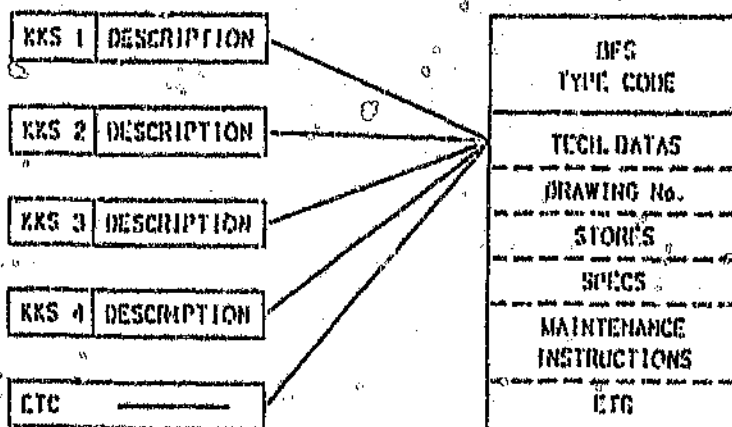
## INTEGRATION OF PLANT CODE &amp; STORES CODE



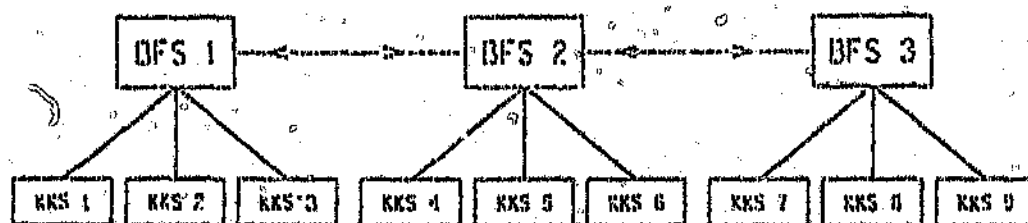
## PRESENT DATA BASES IDENTIFYING SAME COMPONENT

KKS (1)	DESCRIPTION	TECH. DATAS	DRAWING No.	STORES
KKS (2)	DESCRIPTION	TECH. DATAS	DRAWING No.	STORES
KKS (3)	DESCRIPTION	TECH. DATAS	DRAWING No.	STORES
KKS (4)	DESCRIPTION	TECH. DATAS	DRAWING No.	STORES
↓ ETC	DESCRIPTION	TECH. DATAS	DRAWING No.	STORES

## WITH BFS

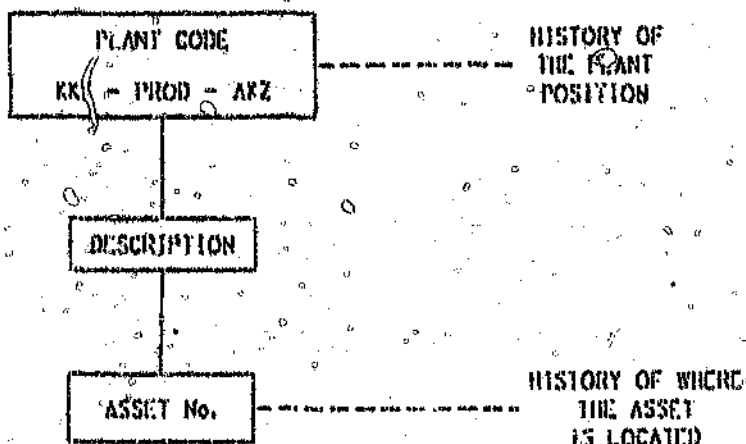


## RATIONALISATION & INTERCHANGABILITY OF STORES ITEMS

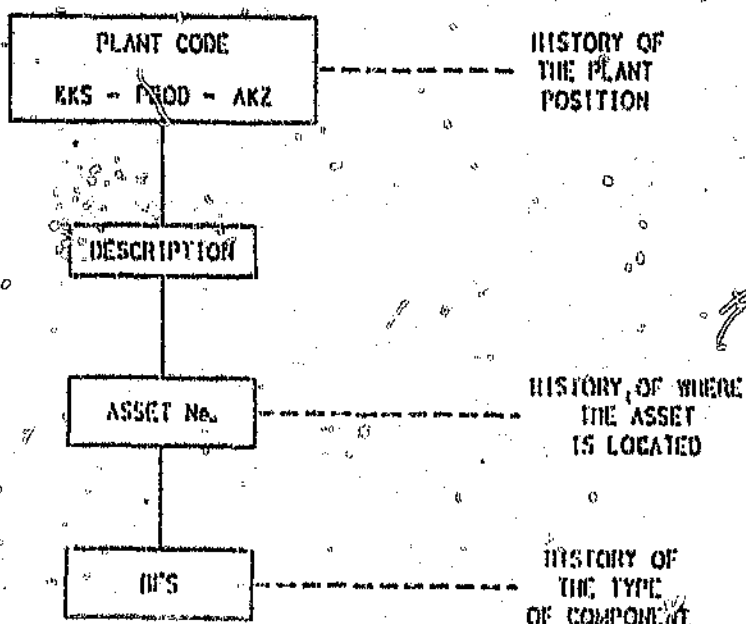




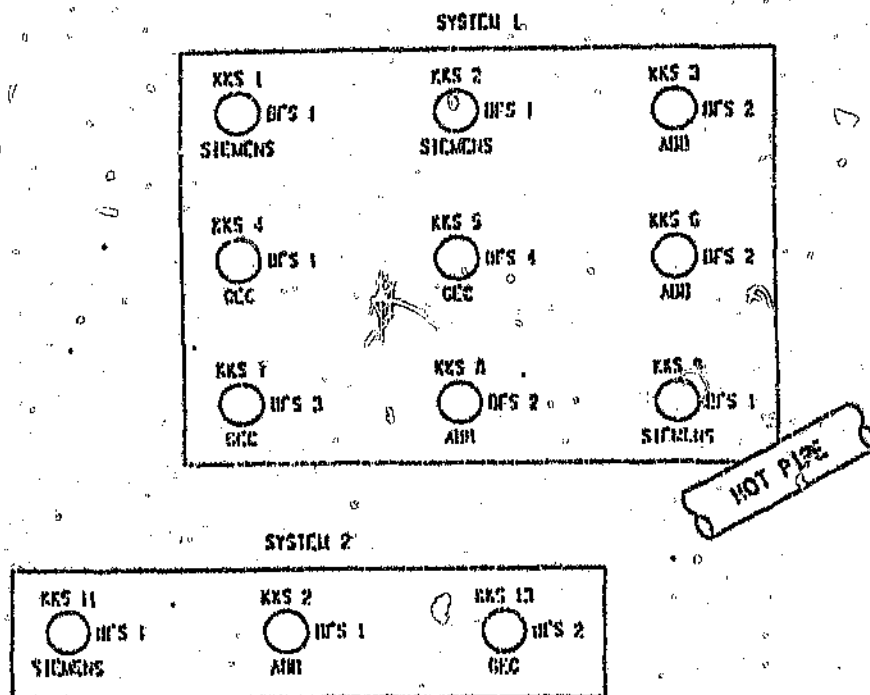
# COMPONENT TYPE HISTORY PRESENT



# COMPONENT HISTORY WITH BFS



## USING HISTORY TO EVALUATE PROBLEMS



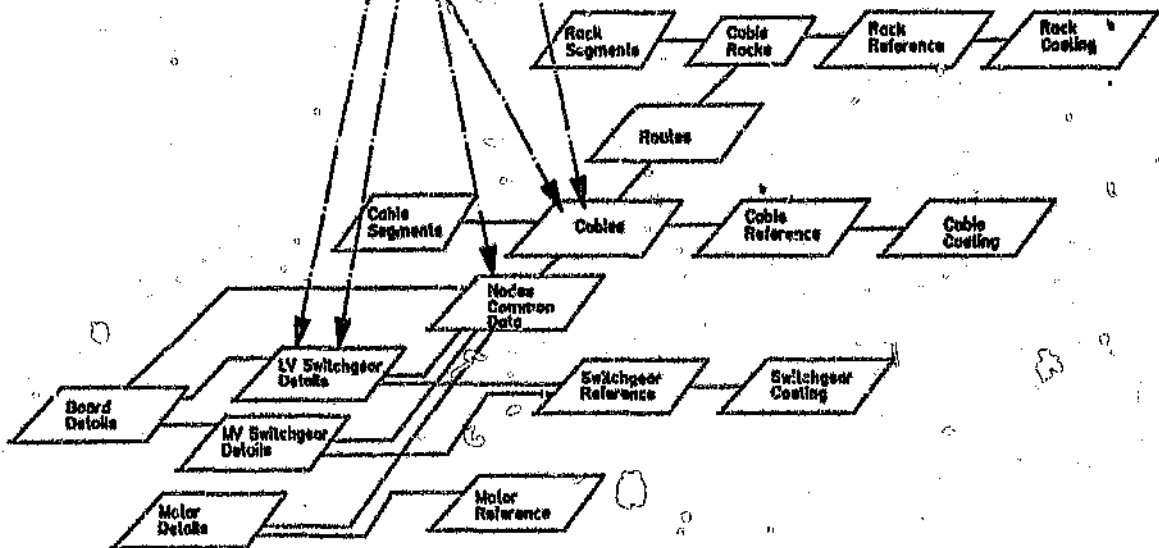
1. POSITION KKS 9  
5 FAILURES OF MOTOR DUE TO LOCATION OF STEAM PIPE  
(SPECIFIC SYSTEM DESIGN PROBLEM)
2. POSITIONS KKS 1, KKS 2, KKS 9, AND KKS 11  
THE MOTORS HAVE FAILED IN THESE POSITIONS  
ONLY WHEN THE SIEMENS MOTOR IS USED  
(TYPE OF EQUIPMENT PROBLEM)

A4.2 Model Structure Reference Data:

A Model Data Structure Dictionary table lies at the heart of all functions responsible for the creation, deletion and renaming of model objects, for example, "Create a Node", "Delete a Cable", etc. It is basically a linkage mechanism between object classes and the relational model used to store the attributes of the instances of the classes. Refer to Figure FA4.2.01.

MODEL STRUCTURE REFERENCE TABLE

OBJ. CLASS		TABLE	KEY FIELD	INS.	DEL.
SWLV	CBLS	SYCODE	D	A	D
SWLV	SWLV	SYCODE	A	D	A
SWLV	SWLV	FEEDCD	D	D	D
SWLV	NOOS	SYCODE	A	A	A
CBLS	CBLS	CBLS	D	A	D



MODEL RELATIONAL REPRESENTATION

CEEDS Model Structure Reference Database

Figure FA4.2.01

The table structure is as follows:

Table Name:	Column Name:	Column Description:
DTADIC	CRTDTE	DATE CREATED
	COMCON	COMPONENT/CONNECTION CLASS CODE
	TBLNME	TABLE NAME
	KEYFLD	PRIMARY KEY FIELD
	KEYFL2	SECONDARY KEY FIELD
	INSTFL	INSERT FLAG (A-ACTIVATED, D-DEACTIVATED)
	DELTFL	DELETE FLAG (A-ACTIVATED, D-DEACTIVATED)

The Table Name is the name of a relational database table that:

- 1) may directly contain attributes of an instance of an object; or
- 2) may refer to other instances of objects.

Provision is made for those objects whose unique identities are made up of two parts (contractual documents for example). Usually only one key field is sufficient.

The "insert flag" indicates to the "Create an object" functions those tables into which records must be inserted when a new instance of an object is registered. If the flag is active ("A"), a record will be created, if it is deactivated ("D"), no record will be created.

The "delete flag" indicates to the "Delete an object" functions those tables from which record(s) must be removed when an instance of an object is deleted. If the flag is active ("A"), record(s) will be deleted, if it is deactivated ("D"), no record(s) will be deleted.

The Model Data Structure Dictionary table will contain a list of ALL places, throughout the relational model, where an object is stored or referred to, even if the record containing a reference to an object is neither created nor deleted by the activation of any of the create or delete routines. This facilitates the renaming of an object because this list essentially tells the "Rename an object" routine all the places where such a class of object must find itself in the relational model.

This method of model "management" has additional benefits, these being:

- 1) Model integrity is managed in a very simple (and therefore speedy) manner. The most important action is deletion because of the rippling implications of an object being deleted in such a manner as to compromise model integrity. Basically if a reference is found to an object in any location where the "delete flag" is deactivated, then that object cannot be deleted before the reference is removed. One problem with such a simple mechanism is that it cannot easily be analyzed as to why a deletion was prevented without building individual checks into the program. This is undesirable as it removes the generic nature of these routines. It is possible though to give the table and column names that were detected as contravening the deletion criteria, but the user would then have to interpret the physical reason.
- 2) The routines that create, change and delete objects are totally generic, i.e. a single "Create an object" routine is able to create any class of nodes, racks and cables.
- 3) The capturing of changes due to deletion and renaming takes place in a totally generic manner as well. (Refer to Appendix 5.1).

The following is an extract from the CEEDS Data dictionary table for LV Switchgear Nodes and for Cables:

OBJECT CLASS	TABLE	KEY FIELD	INSERT	DELETE
CBLS	CBLDDT	CABNUM	D	A
CBLS	CBLRTE	CABNUM	D	D
CBLS	CBLSDT	CABNUM	D	A
CBLS	CBLSDT	CBLPRT	D	A
CBLS	CBLSNT	CABNUM	D	A
SWLV	CBLSDT	SYCODH	D	D
SWLV	CBLSDT	SYCODL	D	D
SWLV	CQDDIC	SYCODE	D	A
SWLV	NODSDT	SYCODE	A	A
SWLV	NODSNT	SYCODE	D	A
SWLV	SWDCDT	FDSCDE	D	D
SWLV	SWLVCM	SYCODE	D	A
SWLV	SWLVDT	FDSCDE	D	D
SWLV	SWLVDT	SYCODE	A	A
SWLV	SWMVDT	FDSCDE	D	D

### A4.3 Working within the Model to handle real world problems

There are a number of real world situations that "stretch" the model beyond its normal capabilities. In these cases compromises need to be made, suitable "work-arounds" established and conventions set up. A number of identified problems are listed below and the conventions adopted to solve these problems are detailed.

#### A4.3.1 Nodes:

- Various Interpretations of a Board Node:

A board can be seen as a "cabinet" containing individual Cubicles in which switchgear are housed or it can be viewed as switchgears being fed off a busbar. For board loading purposes, the second interpretation is used but this poses problems for equipment which are part of the board but are not connected to the busbar. (See next point). Refer to Figure FA4.3.01.

There are also various options for the modelling of Sub-busbars within a board. One is to consider the sub-busbar as a separate board and the other is to consider the sub-busbar as a loadless node and to allocate all of its loads to the main board. Refer to Figure FA4.3.02.

- Coding Inconsistencies Introduced by KKS rules:

Switchgear Labeling rules state that the Switchgear Cubicle has a "Point of Installation" code which identifies a given switchgear in a board. The switchgear cubicle must also display the identity of the node that it is feeding. Normally this is no problem as the node that is being fed is separate from the switchgear cubicle. There are cases though where the node being "fed" is the switchgear cubicle ITSELF but has a different code. In other words, one node is identified by two valid codes. Auxiliary supplies to a board fall in this category because they are part of the board and thus have an identification code indicating this, but they are actually being fed from a source external to the board. The solution adopted is to create a Pseudo node for the "Code" being fed. All the attributes of this "Fed Node" are inherited by the switchgear cubicle. From a board loading point of view, the "Fed Node" draws no load from the board of which it is part, but from the point of view of the board feeding the cubicle with auxiliary power, the cubicle is seen to draw a load.

These situations are depicted in Figure FA4.3.03.

- Single source with Multiple "daisy chained" loads:

A switchgear is modelled in such a way that it feeds a single load. There are cases where a number of loads are "daisy chained". In this case one pseudo node is created with all of the multiple real nodes allocated to it. I.e. the pseudo node is the "Parent" node of all the real sub-nodes. The switchgear then indicates that it feeds the one pseudo node. Refer to Figure FA4.3.04.

- Single node with multiple feeds of different characteristics:

A single node may receive power from multiple sources. The nature of these sources could be different, for example, one node may receive Main AC power and Auxiliary AC and DC power. In this case there is one Real Node which is modelled to contain multiple pseudo nodes. Each pseudo node has the load characteristics of the particular supply it is receiving. Refer to Figure FA4.3.05.

### A4.3.2 Links:

- **Non-Cable connections:**

There are link classes that are not catered for explicitly because of the low number of occurrences. Busbars and direct bolting are examples. "Ghost or Pseudo" Cables are used for this purpose.

- **Cabling to Pseudo nodes:**

Sometimes there is the need to create a Pseudo cable coupled to pseudo nodes. An example of this is the creation of a cable from a switchgear cubicle to the "SPARE". This is for purposes of specifying terminal size on the switchgear.

Real cables must be coupled to real nodes, not pseudo node. In the case depicted in Figure FA4.3.05, the Macro level connections are to the pseudo nodes for board loading purposes, but the cables can only be pulled to the real node because the pseudo nodes don't exist in real life.

- **Split cables:**

One of the fundamental assumptions of the Node-link model is that a link can only be connected to one node at either end. In the case of large multi-core cables such as those used for telephone trunk cables, the cable can be split and terminated to multiple end points. This is modelled by creating a pseudo node at each split and the cable is viewed as multiple links, even though it is actually a single physical cable.

- **Damaged Cables:**

The following is an extract from a CEEDS procedure document detailing Cabling exceptions.

Author: Johan Pienaar.

#### **HANDLING OF CABLING EXCEPTIONS IN CEEDS 2**

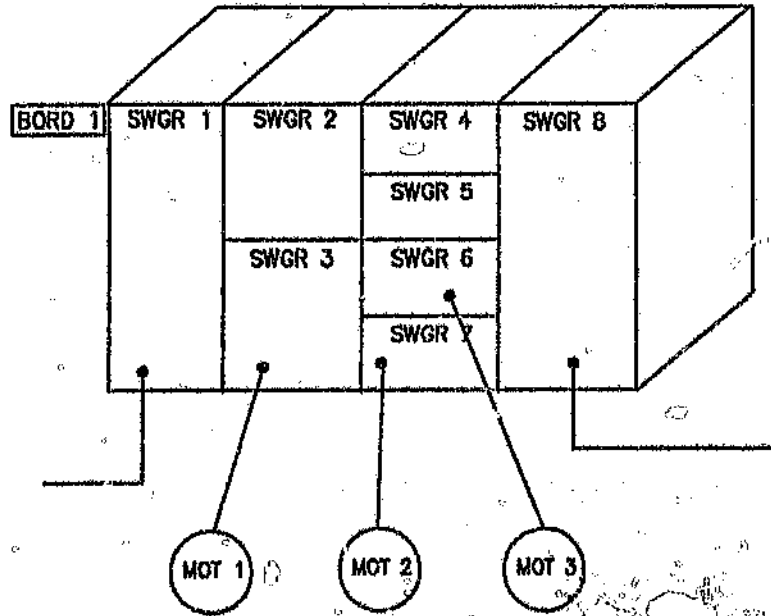
The following exceptional cases concerning cable installation on MAJUBA have been identified and need to be addressed in the context of the CEEDS cable management system.

##### **1. REPLACING PORTIONS / WHOLE CABLES.**

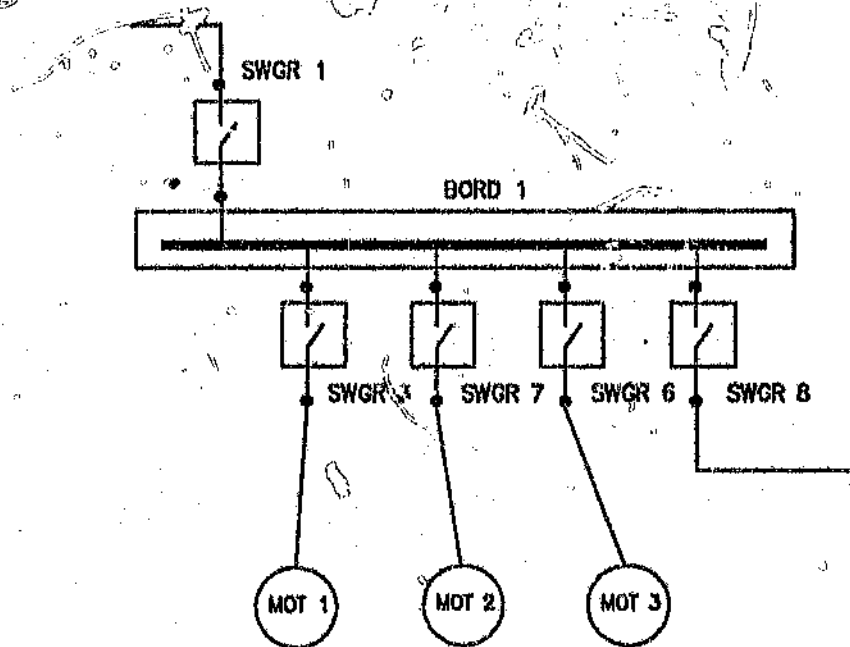
When a part of a cable or a complete cable needs to be replaced the following aspects need to be considered.

1. The original cable's information needs to be kept intact.
2. The instruction to pull the new cable must be issued via a cable pull card giving clear instruction to the contractor on what needs to be done.
3. The contractor must be paid for the new cable.
4. A record must be kept of the new cable.
5. The cable drum management system must be accurately maintained.





Board modeled as a Cabinet  
(point of view from a "Switchgear schedule")

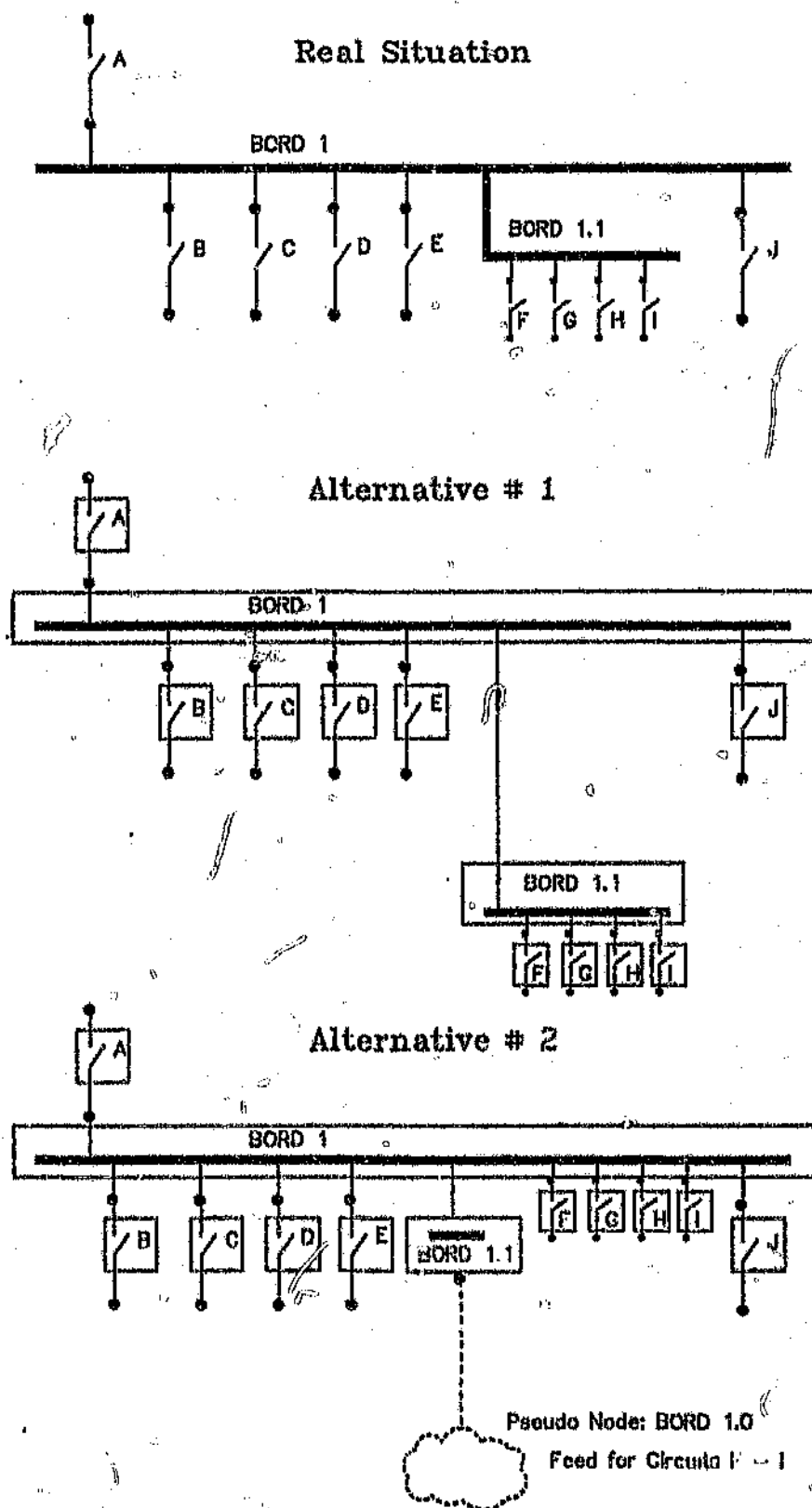


Board modeled as Switchgear attached to a busbar  
(from a "Board Loading" point of view)

A3301

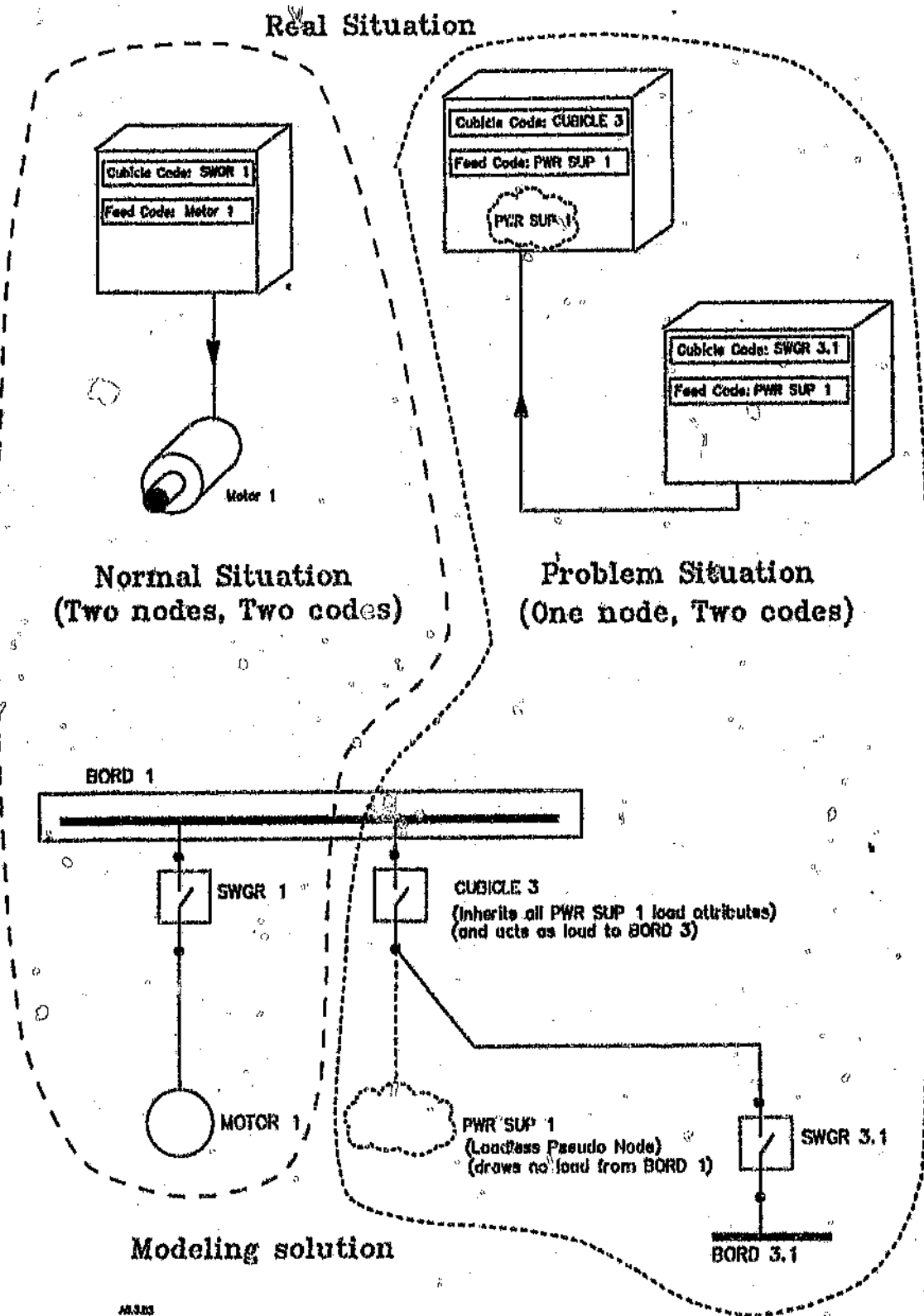
## Modelling of a Distribution Board

Figure FA4.3.01



## Modelling of Sub-Busbars

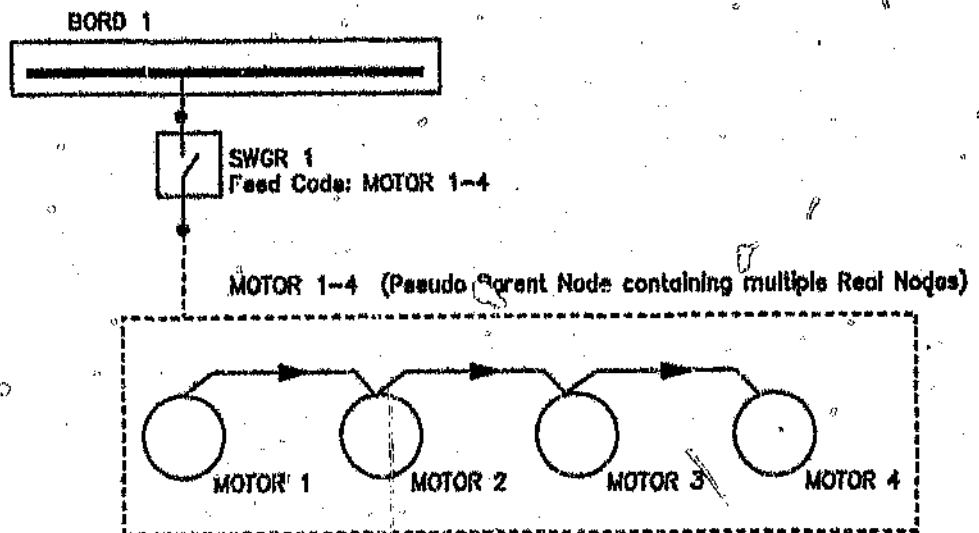
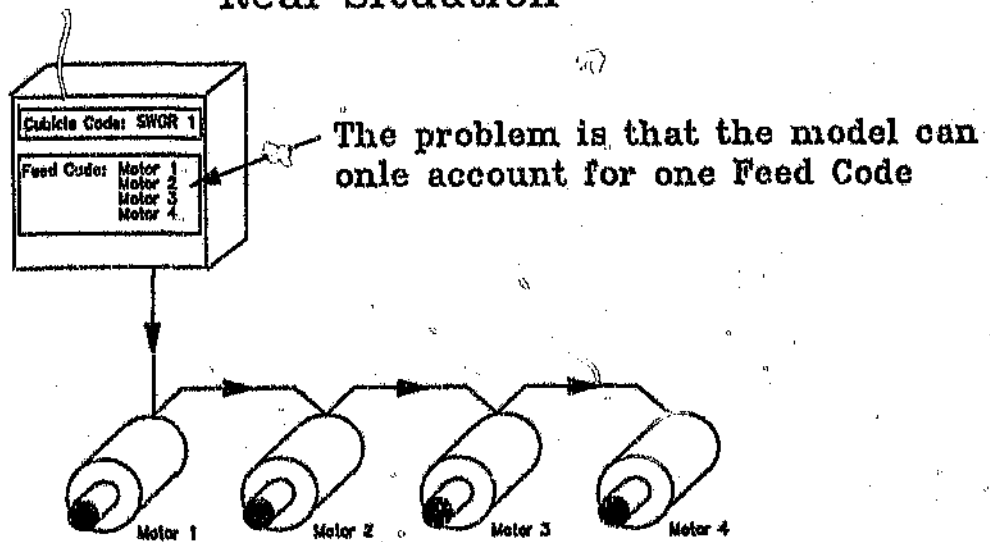
Figure FA4.3.02



Labeling problems due to KKS rules

Figure FA4.3.03

## Real Situation

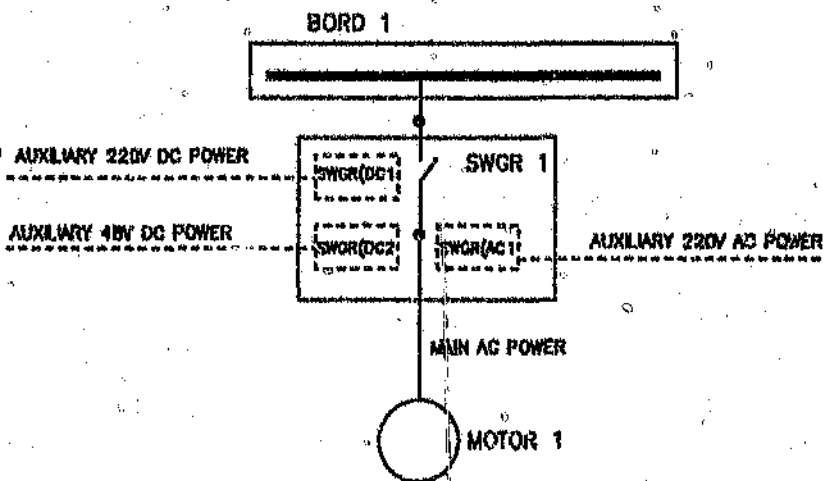
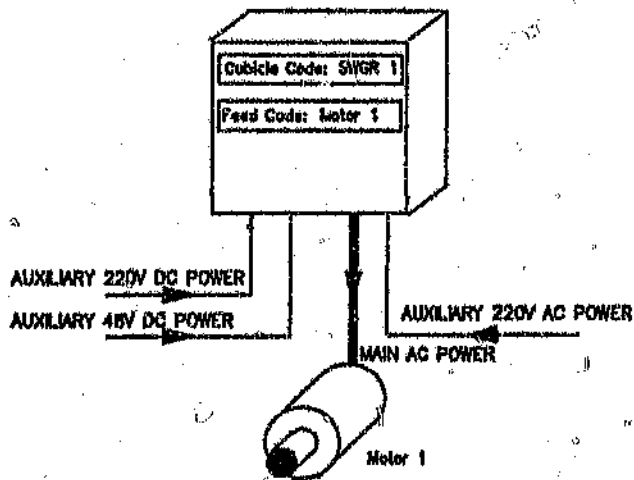


## Model Solution

Switchgear with Daisy Chained Loads

Figure FA4.3.04

# Real Situation (Single Node, Multiple load characteristics)



## Modeling solution (Single Real Parent Node with multiple ) (Pseudo Children Nodes)

Single Node with Multiple Loads

Figure FA4.3.05

**6. A clear link must exist between the original and new cable.**

Looking at the above statement it is therefore clear that a new cable number needs to be generated for the new cable. The issue of a new revision on the original pull card will mean that none of the above conditions can be met,

I therefore propose the following strategy to overcome this problem.

1. An additional field is placed on the cable pull card for notes and specific instruction to the contractor.
2. A new cable number is derived from the original number by adding a numeric value in the place of the last two characters of the cable number e.g. 01BCC1001BFES will get a cable 01BCC1001BF01 if a portion of the cable needs to be replaced. This numbering system will therefore establish the link between the two cables but the new cable will not appear on any cable schedules or drawings.
3. Then new cable will come from and go to a dummy node "JOINT" for example, which will be registered on CEEDS as a node. The actual length of the new piece of cable can now also be given.
4. This cable will not be routed as it will take the route of the cable it replaces.
5. When the new cable is pulled all site feedback for this cable will be completed and the cable drum management system will therefore be kept up to date.
6. The repaired cable will in all respects exist on drawings and on CEEDS as the original cable but a full record will exist of the repairs.
7. The contractor can now invoice for both cables under different cable numbers and no confusion can therefore exist as to what was done on site.

## **2. USING EXISTING CABLES FOR OTHER APPLICATIONS.**

This situation can come about if a previously pulled cable is disconnected and re-connected to a different node. (For example, a temporary cable is swung over to the permanent location)

If a new pull card is printed in this case the contractor may pull an entirely new cable if no clear instructions are included.

The following approach is proposed.

1. If the cable number does not change a new revision of the pull card must be printed using the pulcard notes field to give the contractor clear instruction.
- 2.1 If the cable number does change the original cable must be cancelled first. The original cable must then be registered on a dummy (pseudo) cable drum containing the length of the cable. This drum will be handled as free issue cable.
- 2.2 A new cable must now be generated and a pull card issued for this cable using the notes field for instruction and reference to the original cable number.
- 2.3 With the completion of site feedback the new cable must be registered in the drum management system as coming off the dummy drum.

**NOTES:**

- It must be realized that the cable number is the key to the cable management system and great care should be taken not to re-use cable numbers once a pull card has been printed because there might be hidden contractual implications.
- The contractor must be made fully aware of the fact that issue of a revised cable pull card means that the previous revision must be consulted and he must familiarize himself with the status of the cable. It is not an instruction to just pull a new cable. We must also ensure that we are familiar with the status of the cable before we make changes to the pull card.

## A4.4 Current Reference Facilities

Currently Reference facilities only exist for:

- Cables;
- Cable Racking;
- Switchgear; and
- Transformers.

This is mainly because detailed design/specification of these classes of equipment takes place on the CEEDS system whereas the detailed design/specification of most other equipment takes place elsewhere and the data is then imported into CEEDS for cabling and rack design purposes.

### A4.4.1 Cabling:

There are two Reference Tables describing Cable Data (excluding the Cable Cost Reference Table(s), refer to Appendix A4.10).

- The main table contains for each cable type:
  - Physical properties such as:
    - Mass (kg) per meter;
    - Diameter (mm);
    - Number of Cores;
    - Default length on a drum (m);
    - Whether the cable has calibration marks or not and in what units;
    - Armoured/un-armoured indication;
  - Electrical properties:
    - Voltage Grade;
    - Fault Current;
    - A reference to Core Electrical properties;
  - Contractual properties:
    - Current and alternative manufacturer;
    - A Cost Reference into the cable financial facilities.

The Core Electrical properties were separated out of the main table because there are many different types of cables with essentially the same conductor core properties. The physical properties such as cable diameter, number of cores, voltage grade, armouring, etc. may change for cables with the same core properties. These core properties include:

- Core area;
- AC and DC resistance / Km;



- AC and DC volt drop / Km;
- AC and DC ground and air rated currents;
- Reactance / Km.

Many of these attributes are used by the cable sizing program to decide on a suitable cable.

#### A4.4.2 Cable Racking:

As with cables, Rack reference data is separated into two tables, but for different reasons. There is a main Rack Reference table which describes a rack's physical properties and another, the Rack Loading Reference Table, defines allowable filling criteria depending on the situation in which the racking is being used.

The Main Table would include for each Rack Type:

- a description;
- a Cost Reference to the Rack financial models;
- width and height or diameter;
- Cross sectional area (this could be calculated, but it is explicitly stored in order to speed up the cable routing program);

If a given type of rack is used for process control cables in one situation and for MV cables in another situation, the filling criteria would obviously differ. Refer to Figure FA4.4.01 for an example. The Rack Loading Reference Table would include:

- Rack Type;
- Rack Application (usage situation);
- % filling allowed. This value may only be exceeded by the routing program if permission is granted in each case. In this way a human designer must assess each situation and take responsibility for the decision to over fill a rack. The actual value is determined for each situation in which the rack type is used and is based on experience gained in previous construction projects.
- an indication whether the % filling is decided by cross sectional area or width filling.

Situation: Process control cables      11KV Power Cables



80% Filling by Area



60% Filling by Width

## Rack Filling Criteria.

Figure FA4.4.01

Refer to Figure FA4.4.01.

### A4.4.3 Switchgear

The Power Station Electrical Engineering Department (PSEED) has developed a set of Standard components from which switchgear assemblies are made up. Along with this set of "Standard Circuits", is a set of Design Guides which depict most common situations. The design guides essentially provide a list of all valid combinations of components which go into making up switchgear.

Switchgear components (standard circuits) and valid combinations are stored in one table. A flag indicates whether a record is for a component or a combination. Other attributes include a description, relevant drawing numbers (Standard Circuit, Design Guide, Cabling/Termination diagram, Manufacturers drawings, etc.), kw range (if applicable), connection type and cost references.

All components and combinations are also revision controlled. The reason for this is that a switchgear is specified as a given combination on a given date. There is a cost coupled to this specific combination. At a later stage, a modification may be made to the combination. A switchgear now specified with this combination would possibly have a different cost. Revision control is therefore required. It is anticipated that future switchgear costing would be coupled to the components. The cost of a combination is then the arithmetic sum of the costs of the components making up that combination.

The specification of switchgear in this manner, i.e. specifying a given combination, is not the optimum manner of supporting switchgear design. This is because a switchgear must be completely specified up-front, there are no facilities to allow iterative design of switchgear.

One of the features of CEEDS is that it is able to generate required cables coupled to switchgear. Currently this generation of cables is limited to power cables. The inclusion of a Cable Configuration Reference Table would facilitate the automatic generation of all cables (power, protection and process control) attached to a switchgear. Each switchgear combination is given a Cable Config. Reference Code which defines all the cables connected to a given combination. The reason for separating the cable configurations from the switchgear combinations is that many combinations may share the same cabling configuration.

If a cable type is predefined, that type is inherited by the generated cable. If not, the cable must be sized in the normal way. Usually all non-power cables are able to be predefined.

Refer to Figure FA4.4.02 for an example.

### A4.4.4 Transformers:

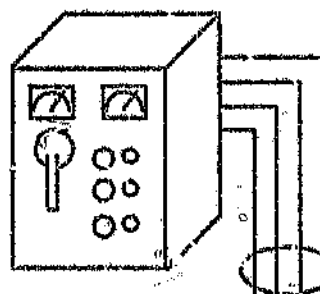
The reference data as it is currently implemented on CEEDS is very specific to the MAJUBA project and is restricted due to decisions taken to limit the types of transformers used. This was done in order to prevent excessive proliferation of spare parts.

The Transformer Reference Data has only one table and includes attributes such as:

- Primary, Secondary and Tertiary Voltage;
- % Impedance, reactance and resistance;
- No-load and full-load losses;
- Cooling type;

# Switchgear Reference

NODE ID: 11BFA01AA001  
CLASS: SWLV  
TYPE: H1/AA/00



Component / Combination	Description	Cable Configuration	
H1/00	HELD IN TYPE # 1	C	
SC10/01	DC FAL RELAY	C	
SC20/03	REMOTE VOLTAGE INDICATION	C	
SC10/01	EMERGENCY TRIP RELAY	C	
SC18/02	EMERGENCY TRIP RELAY	C	
H1/AA/00	H1/00:SC10/01:SC18/02	S	LV01
H1/AB/01	H1/01:SC10/01:SC20/03	S	LV02

## Cable Configurations

Cable Configuration	Cable Function	Cable Type	
LV01	J		
LV01	X	LVG12ACV	
LV01	P	LVG02ACV	
LV01	F	YVG04ACX	

## Cable Functions

Cable Function	Description
J	All Power Cables
X	Process Control Cable
P	Buszone Protection Cable
F	Field Instrument cable

11A02

# Switchgear Standards

Figure FA4.4.02

- Whether numerous protection circuits are required or not. For example, buchholz, oil temperature, low oil, DC fail, fan fault, etc.
- All relevant standard drawing numbers.

It is possible to create a transformer cable configuration reference facility similar to that intended for switchgear above. However, the limited number of transformers is a factor that would probably prevent the development of such facilities.

## A4.5 Current Database Structure

The following is a list of all tables making up the current CEEDS relational Model. Refer to Figure FA4.5.D1 for an indication of the relationships between tables.

Table Name:	Description:
ABRDIC	ABBREVIATIONS DICTIONARY
BORDCC	SW/GR BOARD DETAILS CHANGE CAPTURE
BORDDN	BOARDS DESIGN NOTES
BORDDT	SW/GR BOARD DETAILS
CBLDDT	CABLES DRUM DETAILS
CBLDRM	CABLE DRUM ADMIN DETAILS
CBLROT	CABLES FOR ROUTING TEMPORARY TABLE
CBLRTE	CABLE ROUTES
CBLSCC	CABLE DETAILS CHANGE CAPTURE
CBLSCD	CABLE PULL CARD
CBLSCR	C - SCHEDULE CABLE COSTING
CBLSDH	CABLE DETAILS HISTORY
CBLSDT	CABLE DETAILS
CBL	CABLE SIZING TEMPORARY TABLE
CBLSNT	CABLE NOTES
CBLRE	CABLE TYPE ELECTRICAL REFERENCE
CBLSRM	CABLE TYPE REFERENCE
CBLSTP	CABLE DETAILS TEMPORARY
CNTCNT	CONTRACTS/CONTRACTOR MATRIX
CNTRCT	CONTRACT DICTIONARY
CNTRTR	CONTRACTOR DICTIONARY
CODDCC	PROCESS / P.O.I. CODE DICT. CHANGE CAPTURE
CODDIC	PROCESS / P.O.I. CODE DICTIONARY
COMCON	COMPONENTS/CONNECTIONS DICTIONARY
DOCSCC	CONTRACTUAL DOCUMENTS DETAILS CHANGE CAPTURE
DOCSCL	CONTRACTUAL DOCS CLASSES
DOCSDH	CONTRACTUAL DOCS DETAILS HISTORY
DOCSDT	CONTRACTUAL DOCUMENTS DETAILS
DOCSLG	CONTRACTUAL DOCS PRINT LOG
DOCSNT	CONTRACTUAL DOCS NOTES
DTADIC	COMP./CONN. DATA TABLES DICTIONARY
DTEDCC	CUE DATES DICTIONARY CHANGE CAPTURE
DTEDDH	CUE DATES DICTIONARY HISTORY
DTEDIC	CUE DATES DICTIONARY
EXDTLG	EXTERNAL DATA IMPORT/EXPORT LOG
FILCRF	FILE FUNCTION REFERENCE

FILDIC	FILE CONFIGURATION DICTIONARY
HISDIO	HISTORY TABLE ADMINISTRATION DICTIONARY
INSTDT	INSTRUMENT DETAILS
MNUOPS	A LIST OF ALL POSSIBLE MENU OPTIONS
NODSCC	NODES TABLE CHANGE CAPTURE
NODSDH	NODES DETAILS HISTORY
NODSDT	NODES TABLE
NODSNT	NODES NOTES
NODSPL	NODES PHYSICAL LOCATION
NODSTP	EQUIPMENT (NODES) TEMPORARY TABLE
RACKCC	RACK DETAILS CHANGE CAPTURE
RACKCD	RACK INSTALLATION CARD INFORMATION
RACKDH	RACK DETAILS CHANGE CAPTURE
RACKDT	RACK DETAILS
RACKNT	RACK NOTES
RACKRL	RACK LOADING REFERENCE
RACKRM	RACK TYPE REFERENCE
RACKSG	RACK SEGMENT DETAILS
RACKTP	RACK DETAILS TEMPORARY
RAKROT	RACKS FOR ROUTING TEMPORARY TABLE
REFDIC	COMCON REFERENCE TABLES DICTIONARY
REPSDT	CEEDS GENERAL REPORTS
STRDIC	STRUCTURE CODE DICTIONARY (IMP. FROM PERMAC)
SWDCCG	DC SWITCHGEAR DETAILS CHANGE CAPTURE
SWDCOT	DC SWITCHGEAR DETAILS
SWGRCF	CABLE CLUSTER FUNCTION REFERENCE
SWGRRG	SWITCHGEAR CABLE CLUSTER REFERENCE
SWGRRM	SWITCHGEAR TYPE REFERENCE
SWLVCC	LV SWITCHGEAR DETAILS CHANGE CAPTURE
SWLVDT	LV SWITCHGEAR DETAILS
SWMVCC	MV SWITCHGEAR DETAILS CHANGE CAPTURE
SWMVDT	MV SWITCHGEAR DETAILS
TRFRCC	TRANSFORMER DETAILS CHANGE CAPTURE
TRFRDT	TRANSFORMER DETAILS TABLE
TRFRRM	TRANSFORMER TYPE REFERENCE
USRLOG	A LOG OF ALL ACTIVITIES USED BY A USER
USRPRF	PREDEFINED ACCESS TO MENU OPTIONS BY USERS
USRSEC	ALL PERSONAL DATA OF THE CEEDS USERS

A listing of all columns for each of the tables is available but is not included in due to excess bulk.



**Figure FA4.5.01**

## **A4.6 Equipment Classes being considered for detailed modelling**

Currently the following equipment classes are being modelled in detail:

- Cable Racking;
- Cables;
- Power Distribution Boards;
- Medium Voltage AC Switchgear;
- Low Voltage AC Switchgear;
- DC Switchgear;
- Transformers;
- Junction Boxes;
- Motors; and
- all others are Miscellaneous nodes.

The criteria for determining whether an equipment class needs to be modelled in detail include:

- Are the attributes sufficiently different from existing classes to warrant the effort ?;
- Are the quantities of a specific class sufficient to warrant the effort ?;
- Are there existing stand-alone systems for such equipment ?

Other equipment that can still be modelled in more detail includes:

- Plant Instruments;
- Batteries;
- Battery Chargers;
- Battery Tripping Units;
- Dip Proof Inverters;
- PLC's;
- Measurement, Alarm and Metering Panels;
- Local Control Panels;
- Valves;
- Resistive loads (Heaters);
- Supply points;
- Lighting;
- Protection panels;



## A4.7 Cable Racking Facilities

Refer to Figure FA4.7.01 for an example of Rack Modelling.

The following is an edited extract from the CEEDS hypertext based user documentation. Topic links are left marked in the text by a pair of '#' characters.

Author: Johan Pienaar.

\*\*\*\*\*  
TOPIC rack\_design

ASSOCIATE initial\_rack\_design , updating\_the\_rack\_design ,  
approving\_the\_rack\_design , rack\_site\_feedback

TEXT

The rack design process consist of the following basic steps

1. #INITIAL RACK DESIGN#.
2. #UPDATING THE RACK DESIGN#.
3. #APPROVING THE RACK DESIGN#.
4. #RACK SITE FEEDBACK#.

##

\*\*\*\*\*  
TOPIC initial\_rack\_design

ASSOCIATE volume, rack\_application, dummy\_racks, meterstones

TEXT

INITIAL RACK DESIGN

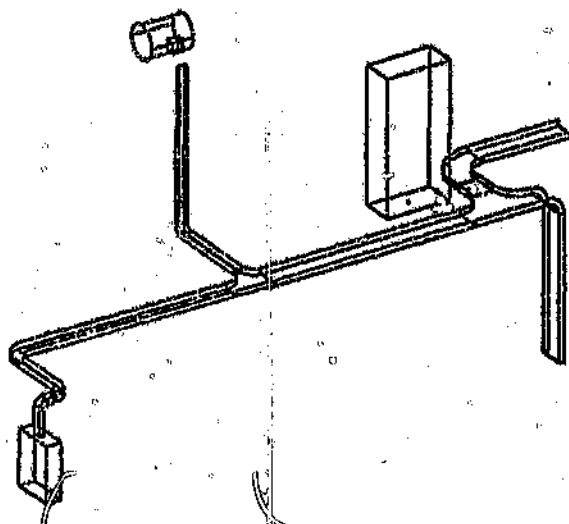
Centre lines of future racks should be digitized in a #volume# representing the area the racks would be in well in advance since the rack sizes need not be accurately determined in order for the initial design to be loaded into CEEDS but the #rack application# must be given.

Only the centre lines of the rack segments are drawn on the Digitizing package and the placing of the lines need therefore not be that precise.

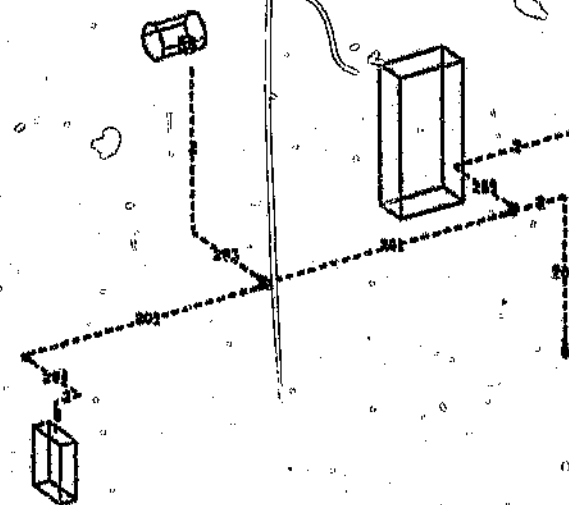
This means that #dummy racks# need not be introduced into the system to cater for changes in the width of a rack in order to keep one of the edges of the rack in line, only the centre lines need to line up.

This will mean that the racks need not be re-digitized if the width of some segments changes, later on in the design process.

## RACK MODEL



## DIGITISED MODEL



## RACK DATABASE

KKS NUMBER	S	X FROM	Y FROM	Z FROM	X TO	Y TO	Z TO
01UEF01B0200K11	1	-1698	19738	70	-1590	19738	98
01UEF01B0200K11	2	-1598	19738	90	-1782	19738	98
01UEF01B0201K11	1	-1727	19689	90	-1745	19708	98
01UEF01B0202K11	1	-1726	19689	90	-1758	19648	98
01UEF01B0203K11	1	-1727	19689	98	-1745	19708	98
01UEF01B0203K11	2	-1745	19708	98	-1745	19708	126
01UEF01B0204K11	1	-1759	19648	98	-1736	19648	98
01UEF01B0204K11	2	-1736	19648	98	-1738	19636	98
01UEF01B0204K11	3	-1738	19636	98	-1738	19636	92
01UEF01B0205K11	1	-1782	19738	98	-1718	19748	98
01UEF01B0205K11	2	-1718	19748	98	-1786	19768	98

## CABLE RACKING MODEL

Figure FA4.7.01

The only dummy racks that will be allowed into CEEDS are those that are required to bridge two racks and thereby creating a link between otherwise separate racks for routing purposes.

Once the initial design for an area is complete the design for that area can be exported from CAD (Micro Station) and loaded onto CEEDS. At this stage the technical status will be IDF and these racks are therefore now available to CEEDS to aid the cable design process.

When the racking design is imported into CEEDS the segment lengths and #meterstones# for the beginning and end of each segment are calculated and updated in the Segment and rack details tables. For this process it must be ensured that the rack segments numbers follow each other consecutively.

Delays in the initial racking design will therefore delay cable design since routing and sizing can not be performed.

##

\*\*\*\*\*  
TOPIC updating\_the\_rack\_design  
ASSOCIATE rack\_filling

#### TEXT

If it is seen that some racks are getting full or some racks have been left out the rack design can be changed on CAD and re-loaded into CEEDS. Changes to the rack design will however influence design data already existing on CEEDS and must therefore be handled very carefully. The two aspect that will be affected is Cable routes and #Rack filling#.

Upon uploading CEEDS will perform the following functions.

1. If a segment has been left unchanged the existing data on CEEDS will be left unchanged.
2. If the width of a segment has changed the rack type will be updated and the rack filling of the rack will be recalculated. If the filling now exceeds 100% a warning will be issued that some cable will have to be unrouted first and the process is terminated.
3. If a segment has moved in any way (coordinates of the start or end of the segment have changed) and there are cables routed on the segment a warning will be issued to unrout the cables first. If there are no cables on that segment the segment length and coordinates will be updated on CEEDS.
4. New segments will be added.

5. Segments that have been left out of the Updated design will be deleted from CEEDS if there are no cables routed on that segment. A warning will again be issued to unroute cables routed on this segment first.

##

\*\*\*\*\*  
 TOPIC approving\_the\_rack\_design  
 ASSOCIATE rack\_installation\_cards

TEXT

#### APPROVING THE RACK DESIGN

Once most cables have been routed and the required rack widths more accurately determined the final rack design is again imported into CEEDS as an approved design.

The rack design status becomes "Full Design Approved" (FDA) and the same checks will be performed as before.

Once the rack design has passed through this process the #rack installation cards# can be approved (Status becomes "Final Fully Approved" (FFA)) and printed on site.

##

\*\*\*\*\*  
 TOPIC rack\_site\_feedback  
 ASSOCIATE

TEXT

#### RACK SITE FEEDBACK

At various stages in the rack construction cycle the progress must be filled in on the rack installation card and entered into CEEDS using the rack site feedback facilities.

The site feedback will serve as an indication of the site progress and will indicate any deviations from the racking design.

Once the site feedback on the rack design indicates that the rack has been built the cable pull cards of cable routed on those racks can be printed.

No racking invoices will be processed unless the site feedback is complete.

##

\*\*\*\*\*  
TOPIC meterstones  
ASSOCIATE

TEXT

#### RACK METERSTONES

A meterstone is defined as the distance from the start of a rack to a certain point on the rack traveling along the rack segments.

The start of segment 1 of a rack will therefore have a 0 meterstone and the end will have a meterstone of 5 if the segment length is 5m.

The meterstones are used to indicate where the cable enters and leaves the rack and in what direction it will travel when printing the route of the cable on the cable pull card.

##

\*\*\*\*\*  
TOPIC volume

TEXT

#### 3-D VOLUME

This is the volume in a 3-D space representing a area of the Power Station that is identified by a structure code.

The location of the area is measured from the centre point of the station and all equipment and racks in that area are placed in the volume using the digitizing process on CAD.

##

\*\*\*\*\*  
TOPIC dummy\_racks

TEXT

Dummy racks are imaginary racks that present a possible route for cables but it does not physically exist on site. Dummy racks are identified as having a 2 as the last character of the rack type.

Ghost racks (short L shaped racks used on CEEDS1 to force a cable onto a certain rack) will not be allowed on CEEDS2 since the routing program will take care of these situations by proper rack selection during the routing process. In this case the Data in CEEDS should model the physical situation as closely as possible.

Ghost racks are also unacceptable from a routing point of view since it increases the number of nodes in the routing network and thereby taking up valuable memory space and slowing down the routing process. (See Cable Routing)

##

\*\*\*\*\*  
 TOPIC rack\_filling

TEXT

PROCEDURE TO CALCULATE AND UPDATE THE FILLING OF RACK SEGMENTS.

For a given rack type and application get the following:

Area/width flag (RACKRL.ARWDFL)

% filling allowed (RACKRL.PERVOC)

If the segment is filled by area

Add cable width\*2 to RACKSG.AERFIL

calculate % filling (RACKSG.AERFIL/RACKRM.RAKAER \* 100)

else

if core area <= 16mm sq (CBLSR.COREAR)

add 0.5 \* width to RACKSG.WIDFIL

else

add width to filling RACKSG.WIDFIL

calculate % filling (RACKSG.WIDFIL/RACKRM.RAKWTH \* 100)

Add weight of cable (CBLSRM.CABWGT) to total on rack (field must be added)

if % filling exceeds % filling allowed print warning

if % filling exceeds 100% print error and rollback.

##

\*\*\*\*\*  
 TOPIC rack\_application

TEXT

Racks are all dedicated to carry certain types of cable that should not be mixed.

The rack application codes are as follows.

P - Process control.

L - LV multicore cables

S - LV trefoil groups.

M - MV multicore cables.

T - MV trefoil groups.

##

\*\*\*\*\*  
TOPIC rack\_installation\_cards.

TEXT

#### RACK INSTALLATION CARDS.

Rack installation cards are the only contractual documents whereby the contractor can build the racks and invoice Eskom for the construction.

Each racking card printed on site will be given a unique sequence number and strict control will be exercised on the handling of these cards.

The data printed on the cards are derived directly from the rack digitizing process used in the design of cable racks.

The rack installation cards should be accompanied with the relevant 2-D and 3-D racking design drawings to facilitate construction.

##

## A4.8 Current DP model for Cable Design

CEEDS does not currently have a stand alone DP model as described above. A sub-section of the cable design DP model has been designed on paper and is hard coded into relevant Physical Model Manipulation Functions. For example: the selection function for submitting cables to the Sizing Routine "Knows" that only cables whose current status is "IDF" can be submitted for sizing. The sizing routine then updates the status to "SZI" when it has successfully sized a cable. These functions know what their prerequisites and following states are because these conditions are hard coded into the programs. This leads to the very undesirable maintenance problem of having to modify many programs if there is any change in design philosophy that affects the states and dependencies between states.

Refer to Figure FA4.8.01 for an indication of the currently implemented model. Below is an extract from CEEDS documentation that describes the Cable Design Philosophy.

### CABLE DESIGN PROCEDURE.

Author: Johan Piensar.

This document outlines a proposed cable design procedure for use with CEEDS 2.

#### 1. IDENTIFYING A CABLE.

In order for any cable to logically exist on CEEDS at least one end of the cable needs to be connected to a Node.

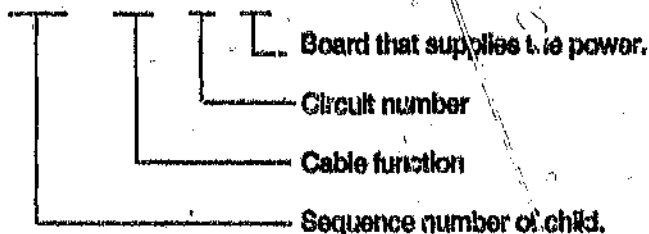
At this stage the cable will get an internal CEEDS cable number that will identify the cable to CEEDS for the rest of its existence in the system.

A power cable can be made known to CEEDS at a very early stage by allocating a CEEDS cable number to the cable.

It was decided to use an internal number because of the fact that the actual cable number can only be allocated once the KIC codes of the nodes on both sides of the cable has been approved.

This internal CEEDS number is made up as follows.

NN AAA NNN AAA NN



A child cable is defined as a cable running in parallel with another cable and performing the same function.



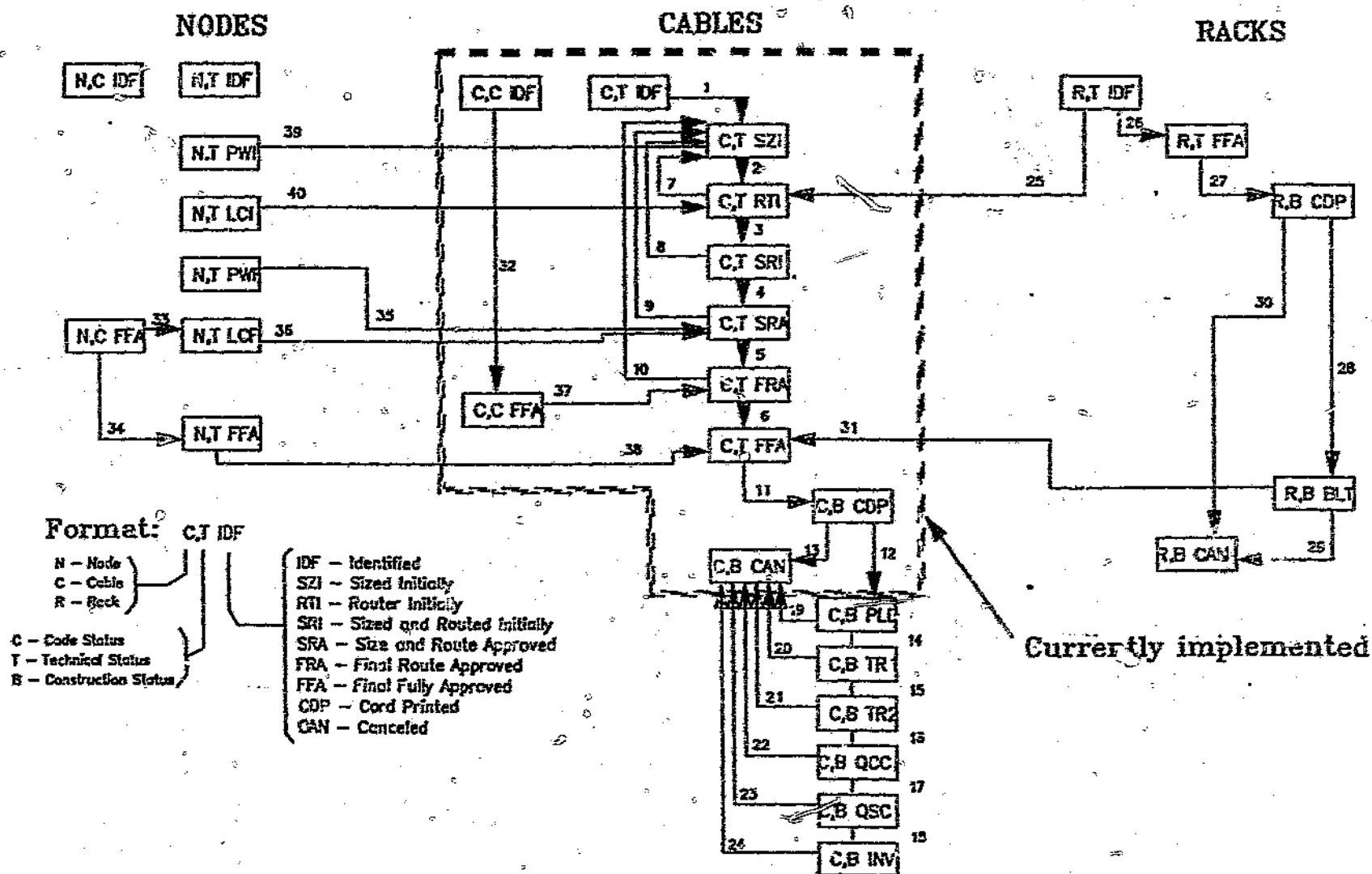


Figure FA4.8.01

CEEDS Design Process Model:

448.01

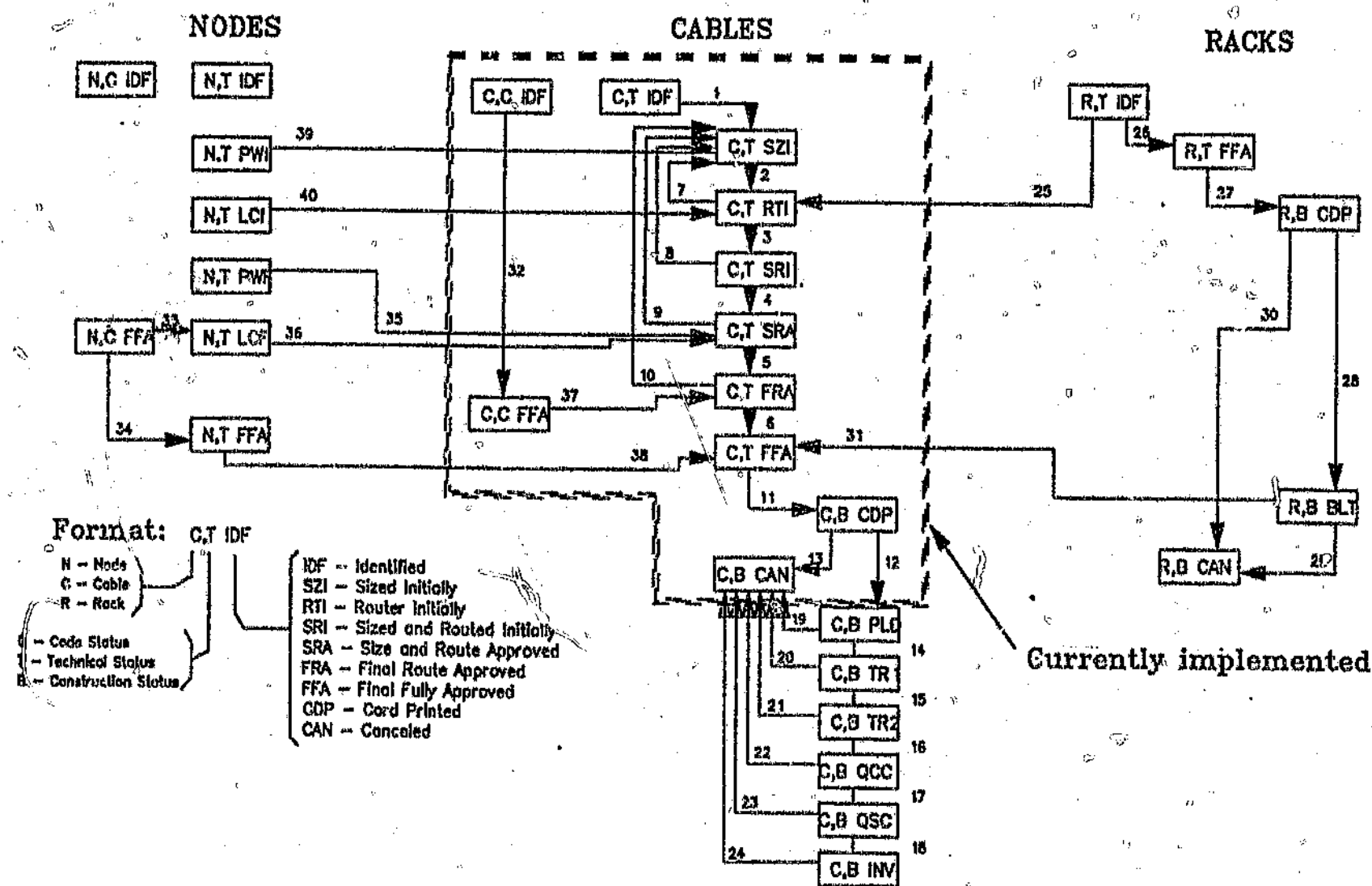


Figure FA4.8.01

CEEDS Design Process Model:

454.01

The concept of a parent cable with associated children cables will form a vital aspect in the handling of cables within CEEDS in that if cables are used in parallel the cable group will always be identified by the parent cable and all the other cables making up the group will be identified as children of the parent cable.

One of the attributes of any child cable is the cable number of its parent. Parent cables will therefore have no parent and the parent cable field will have a "NONE" value.

Most processes will only be performed on the parent cable and the children cables will automatically acquire the attributes of the parent.

The new cable is inserted into the Cable Details Table through the Inset cable screen form found under MAEAI000.

Updates to the cable details are then done through the Update Cable Details screen form (MAEAF000).

## 2. INITIAL SIZING OF POWER CABLES.

If both sides of the cable are known, the power cables can be sized. The initial sizing is done on the default length of 92m unless a more accurate estimate of the cable length is given before hand.

The Cable sizing program may generate additional (children) cables.

The results of the sizing program can be analyzed by studying the two reports produced by the program. These reports will reflect both the electrical and financial considerations given to selecting a certain cable type. The program will use the most economical solution but it can be overruled by the cable designer by updating the data in the sizing temporary table.

The current ratings (amps) given in the cable reference table are given for 3 cables touching. When updating this table care must be taken to insure that the same convention is adhered to. The sizing program will adjust the derating factor if additional cables are generated and/or when the default rating is changed in the sizing reference file.

The sizing program allows for the adjustment of a number of parameters such as temperature, depth in ground, spacing, etc. by changing the default values in the sizing parameter file.

Once the cable has been sized the Technical status of the cable will change to SZI and the cable can now be routed.

## 3. ROUTING.

Before a cable can be routed both nodes that it is connected to must first be digitized and the initial rack design for the areas the cable must pass through must be loaded onto CEEDS.

In order to run the routing program the cables that are to be routed must be selected from the Cable Details Table (CBLSDT) into the temporary sizing table (CBLROT). All the Racks that the cable can be routed on must be selected from the Rack Segments Table (RACKSG) using the facilities on CEEDS. This will only select parent cables since the children cables will inherit the route from the parent.

The routing program will find the shortest route for the cable on the available racks and will place the parent and the children on the rack segments by inserting the cables in the Cables Routes Table (CBLRTE) and updating the % filling of the segments in RACKSG. Refer to Appendix A5.4 for more details.

The routing report must be consulted and the routes checked to ensure that they are correct. Cables must be un-routed if the selected route is not correct and racking design must be checked to find the cause of the routing error.

If a route is found for the cable/group the technical status will be updated to RTE and the cable can then be re sized.

#### **4. RE SIZING.**

Once the design length of the cable has been more accurately obtained, the cable is sized again to ensure that the sizing and economic parameters are met for the new design length.

If the number of cables changes, the cable is unrouted and the status reset to SZI.

If the cable size is changed by the sizing program the rack filling will be updated. If the new cable now no longer fits on the selected route, this cable will be automatically un-routed and the status reversed to SZI. The design length will be retained so that the next iteration of the routing and sizing process will be more accurate.

If the re-sizing program is successful, the status will be set to SRI.

#### **5. ALLOCATION OF ESKOM CABLE NUMBERS**

When the design team is satisfied that most of the power cables for a particular board have been allocated, the program that allocates ESKOM cable numbers can be run. This program will generate cable numbers for the power cables in such a way that children cables will get consecutive numbers to the parent and cables will be numbered from circuit 1 upwards using the cable numbering rules defined in KKS documents.

If this program is run too soon, subsequent reruns of the program may not give entirely satisfactory results as gaps may appear in the numbering sequence because there may be more children cables of previously numbered parent cables and cables may have been deleted since the last run.

#### **6. APPROVING CABLE INFORMATION.**

Once all codes associated with a cable have been approved and the racks the cable is routed on have been built on site (as confirmed via Site feedback), then the cable can be approved for the printing of a cable pull card.

This is done through a screen form that shows all information that will be printed on the pull card and the responsible cable designer must then check this information and if correct approve the cable design.

This puts the cable in the final approved status (FFA) after which no further changes can be made to the cable design.

Pull cards for all cables in the FFA technical status can now be printed on site if and when required.

#### **7. PRINTING CABLE PULL CARDS**

A screen form list of all cables that are ready to be printed is presented from which the cables are selected for the printing of cable pull cards.

The completion of this function will print the pull cards to the selected printer and each pull card will be given a unique pull card number which are allocated sequentially. A pull card may be reprinted if it is damaged in the printer but this will be marked as a reprint and full history is kept of all printing operations.

Once a pull card has been printed this cable can no longer be deleted from the system and if the cable is cancelled it will be marked as cancelled and a cable cancellation note will be issued to inform the site representative and the contractor that the cable has been cancelled.

#### **8. SITE FEEDBACK**

When the pull card was been issued to the contractor the feedback must be completed and fed back into CEEDS as soon and often as possible. This feedback will not only update the cable drum management system but will also serve as indication of the progress on site to the cable design team.

#### **9. CABLE DRUM MANAGEMENT.**

As soon as a cable drum is delivered to site the drum must be allocated an Eskom Cable Drum Number and then registered on the drum management system.

During site feedback of cable information, the cable lengths and drum numbers that the cable comes from are recorded. The drum management system will check that the cable types correspond and it will update the cable drum lengths.

Record can therefore be kept of every single length of cable taken off a drum.



## A4.9 Current Project Plan Link In CEEDS

The problems with the current implementation of CEEDS and the Project Planning Package are:

- CUE, the project planning package, is an off-the-shelf product. There are no easy ways of accessing its "database", other than via provided facilities which limit the data extraction capabilities to a manually triggered extraction of data in a flat file format. CEEDS cannot therefore directly query CUE data. CEEDS, in turn, is based on the ORACLE Relational Database Management system, which is not directly accessible by CUE.
- CUE and CEEDS are running on different mainframes with different operating systems, so inter-communications are limited.

The solution chosen was to duplicate limited CUE data within the CEEDS environment. A "Work Package" reference table was created. Each record represents a high level Work Package and contains its CUE identification, a description, two activities and one date for each activity. Refer to Figure FA4.9.01 for the CUE representation and the corresponding CEEDS record structure.

A "Work Package" ID is attached to each object in the Physical Model. The choice of which two detailed activities to choose out of a Work Package, and then which date to choose for that activity (e.g. early start, start, early finish, finish etc.) are left to the discretion of the designers and planners. A series of extraction "macros" are written for CUE to extract the desired data into a flat file with a format suitable for importing into CEEDS. The extraction from CUE to CEEDS can be a regular activity, or a once off, followed by duplicate updating on both systems. The data duplicated within CEEDS is only a fraction of that contained within CUE as it is limited to key events and a key date within an event. Any attempt to expand these facilities further would be counter productive because of duplication of effort.

From within CEEDS, it would be possible to get lists of equipment that can be grouped by a Date criteria. If any further details are required about a work package and its activities, CUE would have to be consulted directly.

There are obvious limitations to this non-integrable system. These include:

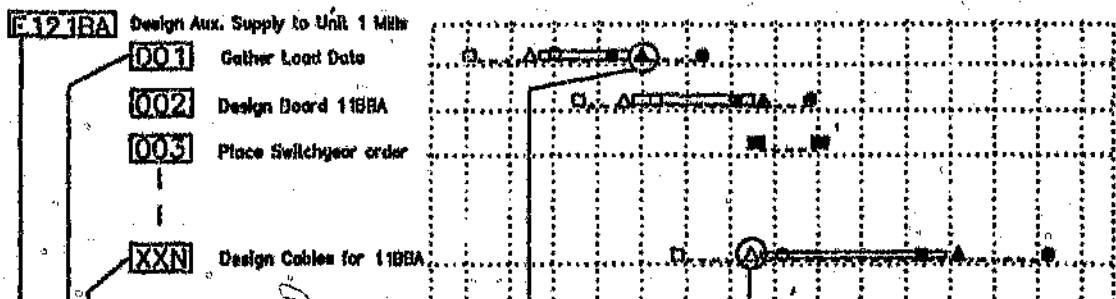
- The limitation of a one-to-one mapping between equipment and work packages;
- Limited date based query capabilities within CEEDS;
- Danger of CEEDS duplicate data falling out of synchronization with actual CUE data. It is not possible to ensure constant data integrity. Reports coming out of one system may be contradictory to reports coming out of the other system;
- Possible need for regular manually triggered updates; and
- Planners need to be trained on two different systems.

# PROJECT PLAN WORK BREAKDOWN STRUCTURE:

EARLY START DATE □  
 START DATE △  
 LATE START DATE ○  
 EARLY FINISH DATE ■  
 FINISH DATE ▲  
 LATE FINISH DATE ●

HIGH LEVEL LOW LEVEL

WEEK NO.



WORK ID.	DESCRIPTION	ACTIVITY ONE	DATE ONE	ACTIVITY TWO	DATE TWO
E 12 1BA	Design Aux. Supply to Unit 1 MVA	001	10-JUL-91	XXN	29-NOV-91

CEEDS "WORK PACKAGE" DATABASE:



PHYSICAL MODEL:

CUE Representation of a WBS and its record structure within the CEEDS Database.

Figure FA4.9.01

#### **A4.10 Current and proposed Financial Models**

CEEDS, being an electrical design facility, would only attempt to do costing on those classes of equipment actually involved with Electrical contracts. Equipment covered by other contracts (e.g. the boiler mechanical contract), would be managed by other systems.

CEEDS currently has facilities for determining costing for Cabling. Shortly Cable Racking will also be implemented. Both of these facilities are driven by having various cost rates attached to a Type of Cable or Rack. These rates combined with physical attributes such as length and number of joints would determine the cost of a cable or rack. Racking is slightly complicated by variations in the support structures. The same rack type could cost more in a given situation due to heavier or more complex support structures.

It is intended that Switchgear costing will also take place on CEEDS. The costing philosophy for switchgear is very different to that of Racking and Cabling. This is because a Board is made up of a number of Cubicles which in turn are made up of a number of components. Costs are attached to components. The specification of a switchgear identifies all of these components and the cost can thus be calculated.

It is unlikely that transformer costing would be incorporated into the system because of the relatively small number. These could be managed manually more effectively.

Currently there are no plans under way to include costing for C & I equipment because of the turnkey nature of C & I contracts.



## A4.11 Proposed Attribute Confidence Factors

Design is a series of processes, many of which are dependent on the output from previous stages. The "goodness" of the outcome from one design process is very dependent on how well previous processes were done. On CEEDS, the "start" of one of the main design processes is the information gathering on equipment requiring power. The more accurately these power requirements are known, the more accurate subsequent design processes can be. For example if a power rating is only a rough estimate, then the cable size can also only be a rough estimate.

A system using "Confidence Factors" attached to selected attributes is proposed as a method of measuring the "goodness" of supplied data. From these factors, an indication can be obtained as to how good subsequent design attributes would be. At the end of the process, an indication can be obtained of the probability and extent of re-work on a system if it is built with data of a given quality. Costs can be attached to this re-work and the relevant decision makers can either accept such possible costs or attempt to improve the initial data and try to bring down the probability and extent of re-work.

Attributes playing a major role in design processes must be identified and supplied with facilities for attaching a Confidence Factor to each such attribute. For the various design processes, e.g. cable sizing, it must be decided what attributes are required to carry out the job of determining new attributes. These are called Contributory Attributes and Dependent Attributes respectively.

For example, the determining of a cable type depends on the power rating of the load, its starting current, its full load current, whether it's a two, three or four wire system, its distance away from the source of supply, etc. An algorithm is used to carry out this process.

Each contributory attribute does not have the same effect on the dependent attributes. For example Starting current may have a less important effect on cable sizing than Power Rating. Those weighting factors for each of the contributory attributes must be determined. The weighting factors along with the confidence factors of contributory attributes are used to determine the confidence factor of the dependent attribute.

For example: Power Rating, length and number of cores are very important for cable type determination while full load current and starting current play a less important role.

The confidence factor of Cable Type may be calculated as follows:

$$D_{Acf} = 1(CAwt_1 * CAcf_1, CAwt_2 * CAcf_2, \dots, CAwt_n * CAcf_n)$$

Where:

$D_{Acf}$  is the Dependent Attribute's confidence factor,

$CAwt_i$  is the i-th Contributory Attributes's Weighting factor,

$CAcf_i$  is the i-th Contributory Attributes's Confidence Factor.

A network of such dependencies can be established. In each case a suitable function and appropriate weighting factors would need to be determined to give end confidence factor values that are meaningful and reflect reality. Case studies, control exercises and weighting factor tuning would need to be done and data gathered for quite some time before these confidence factors can be used with any reliability.

Such a scheme would be implemented along similar lines to all other models discussed in the main report. In other words, all attributes and relations between attributes are explicitly stored in a database and self querying programs written to use the data. In this way, dependencies can be changed and tuning done without having to modify software all the time.

## A5.1 Change Capture Facilities

Ideally data change capture should take place totally transparently and should capture any change, irrespective of how the change was made. Unfortunately the database management system being used on CEEDS does not provide sufficient functionality at the database level to create such facilities. The approach taken on CEEDS is a compromise. All changes taking place through screen forms and by pre-written data processing programs are captured. Any changes made directly to the database via the Structured Query Language (SQL) interface are not captured, although such changes should seldom take place. It was also decided that change capturing would only be done on data that could have contractual implications if an audit trail was not available. Change capture facilities are not attached to reference data because such data should not change too often and if it does, the changes would be detected by integrity checks (Refer to Appendix A5.5).

### A5.1.1 Change Capture Tables

The method used on CEEDS is that each screen form through which data on selected tables can be changed has a "Ghost" table associated with it. This ghost (change capture) table has the same structure as its "host" table with the addition of three fields. These being User ID, Change Date and Change/Delete flag. Every time a record is changed or deleted, the pre-change data is written to the change capture table. On a regular basis, these change capture tables are cleared out and the contents compressed into their corresponding permanent "history" tables.

### A5.1.2 History Tables:

The permanent history tables have the following attributes:

- Object ID code;
- Table in which change took place;
- Column in which change took place;
- Previous value;
- Date and time of change; and
- User ID of person who made the change.

On CEEDS, history tables exist for Nodes, Cables, Racks, Contractual Documents and Work Packages.

### A5.1.3 History Tables Dictionary:

A History Tables Dictionary describes the relations between Tables, their Change Capture Tables and their History Tables. There may be a number of change capture tables that are cleared out into a single History table. For each change capture table, a flag indicates whether the change capturing is active or not. If the flag indicates that the facilities are "switched off", the screen form doesn't capture pre-change records into the change capture table.

#### **A5.1.4 Historic Data Compression program:**

Refer to Figure FA5.1.01. A Historic Data Compression program clears all Change Capture tables into their corresponding History tables. This program reads the History Tables Dictionary to decide which change capture tables to clear into which history tables. Because the program queries a dictionary which describes the configuration of change capture and history tables, the data structure of the physical model can be expanded without modification of the compression program. Maintenance is thus reduced because all that is needed is the insertion of extra records in the History Tables Dictionary.

The compression routine firstly scans through the change capture table to find any records marked as deleted. It then records these deletions in the history table and makes the previous value field "DELETED". Any other record then found in the change capture table that has the same key as a deleted record is itself deleted. This is because the changes that took place to a record before it was deleted are not of any interest and are thus discarded.

It then compares the remaining records in the change capture table with the details table on a column for column basis to determine exactly what has changed. Detected changes are recorded into the corresponding history tables. Once all records in a change capture table have been processed, the table is cleared to begin the next round of change capturing.

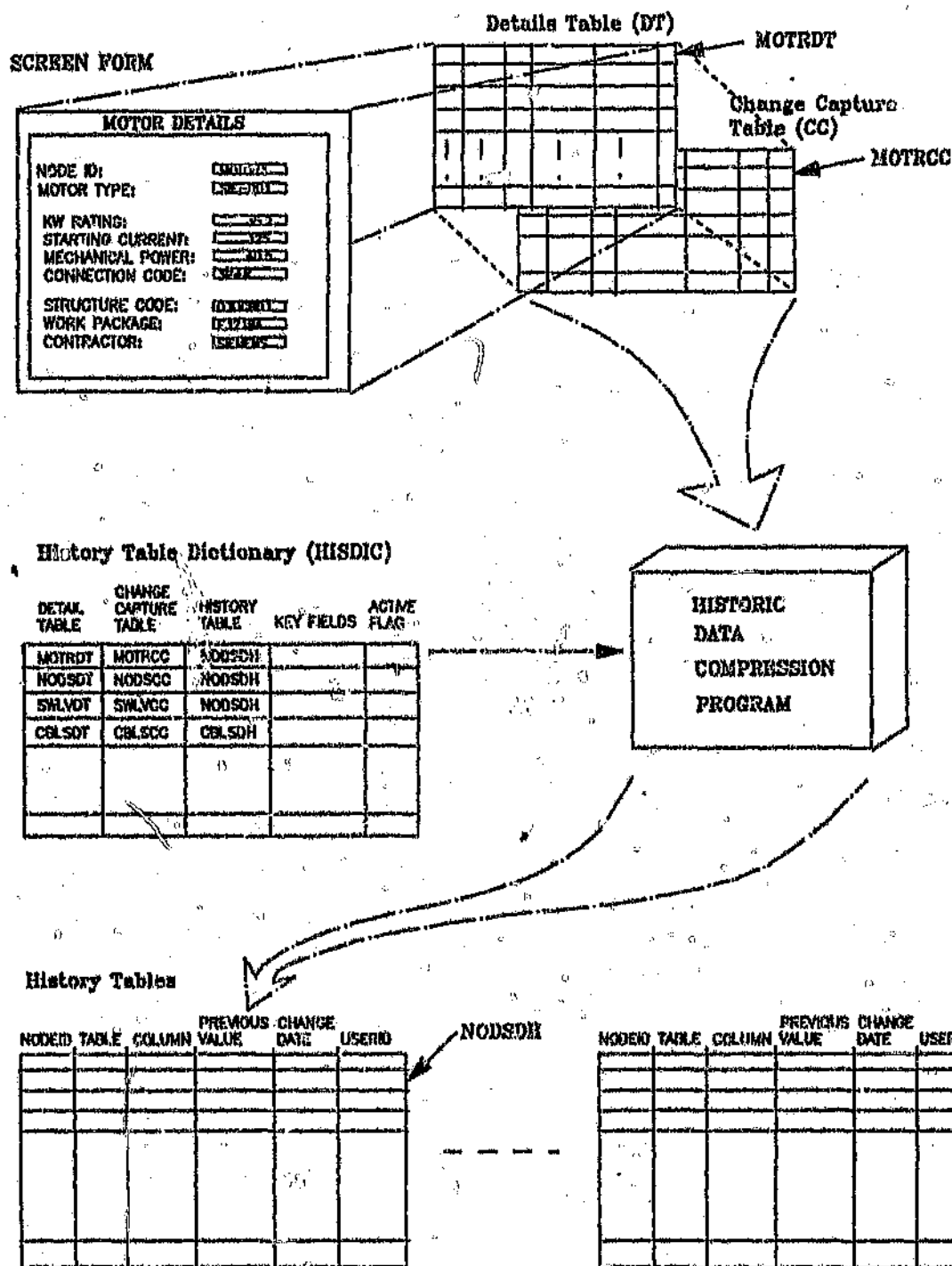
This activity must take place outside of normal use because to ensure that no data is lost, no changes must be captured while the compression routine is being run. It could be coupled to a time activated trigger to execute on a regular basis during night time hours.

#### **A5.1.5 Change capture for Data Processing programs:**

A number of Model Manipulation Functions carry out data modification. These changes do not take place via screen forms as mentioned above. There are two mechanisms available to such programs, either write the complete record to the change capture (CC) table or write just the changes directly to the appropriate History (DH) table. There is a choice as to whether the "User" is the person who activated the routine, or the name of the routine itself. The current philosophy is to use the routine name because there is no guarantee that the routine was triggered by a person, it could have been triggered by another process. The most important of these functions are data Import routines and the Object ID Change and Delete routines.

#### **A5.1.6 Uses of History Data**

- Historic data is kept as an Audit trail for:
- The production of Change Summary Reports for contractual documents. These documents contain all changes that have taken place to the information depicted in a document between one revision and the next;
- The correction of accidentally or maliciously changed data;
- To detect who made certain changes;
- and any other audit activities.



## CEEDS Change Capture Facilities

Figure FA5.1.01

## A5.2 CEEDS Contractual Documentation Production Facilities

The following is an edited extract from the CEEDS hypertext based user documentation. Topic links are left marked in the text by a pair of '#' characters.

\*\*\*\*\*  
TOPIC Std\_Reports  
TITLE " How to generate Standard Reports "

CEEDS can produce three type of documents:

- a) #General Reports#
- b) Standard Reports and
- c) #Contractual Reports#

In most menus, there is an option "P - Reports...". Reports accessible from this facility are regarded as Standard reports. They are pre-written, fixed format reports. Standard reports are written in such a way as to be of general use to many people. If a user requires a specific report, then the #General Reporting Facilities# should be used.

How to use the General Reports Facility:

- On selecting the "Reports..." option in a menu, you are taken into the "General Reporting System". A List of Reports accessible from a given menu is presented. This list is given in a standard #Oracle Screen Form# in which all standard query facilities are available.
- First of all, you need to select the desired report. This is done by using Next Record, Previous Record and/or Next Set of Records Function Keys to get the cursor onto a desired report. NOTE:- Arrow keys cannot be used. See Tips below.
- Once the cursor is on a wanted report, press function key PF15 (or PC equivalent) to see any Instructions that may be supplied with the report. These instructions are sometimes necessary because some reports prompt for certain values and the instructions will give you prior warning of such prompts.
- To create the report, press PF16 (or PC equivalent). This activates the program attached to the report, which creates a file containing the report. (See #Report List#). The #General View/Print Facility# is then activated to give you the opportunity to view and/or print the report.

Tips:

If there is a long list of Reports, and you want one near the bottom, it will be quite slow to use Next Record all the way down. It is sometimes faster to go into query mode and give a search criteria on the report title that will only return the desired report. You can then run it directly.

There are facilities for passing parameters to reports. For more details

on how Reports are Registered and allocated to Menus, see "Reports List".

```
*****
TOPIC Std_Reports_List
TITLE " Standard Reports List "
```

Any #Standard Report# must be registered in CEEDS in the Report List. A report is identified by its Title. Attached to a report is a Report Program Name (consisting of a File Name and File Type). This Report Program File is the program which creates the report. Also attached to a report are its instructions.

When a report is registered, it is allocated a #menu address#. This is the menu address from which the report will be accessible. For example: If a report is given the menu address of MAAA0000 then the "P" option in menu MAAA0000 will contain this report.

If a report is to be accessible from multiple menus, the report is simply registered any number of times, each time being given a different Menu Address.

There are view facilities on this Reports List.  
(See #System Query Facilities#).

A list of parameters can also be supplied for a given report if needed, although this facility will seldom be required. Also attached to a report are two flags:- one indicating whether #User Activity Logging# is to take place and the other indicating whether "Document Print Logging" is to take place. Currently both these logs are de-activated.

```
*****
TOPIC General_Reports
TITLE " General Reporting Facilities "
```

CEEDS can produce three type of documents:

- a) General Reports
- b) #Standard Reports# and
- c) #Contractual Reports#

General reports can be generated using SQL Plus or SQL QMX, both ORACLE products.

For beginners, QMX is far easier to use than SQL Plus, and for experienced

users, QMX can be used for quick simple queries. You only need to fall back on SQL Plus for more complex queries.

In general, facilities exist for ANY users to create their OWN QMX or Plus reports, save them to their OWN private storage area and run them at ANY time.

To learn either QMX or Plus, refer to the relevant manuals, or attend the relevant courses.

\*\*\*\*\*  
TOPIC Contractual\_Documentation  
TITLE " Contractual Documentation "

CEEDS can produce three type of documents:

- a) #General Reports#
- b) #Standard Reports# and
- c) Contractual Reports

The term "Contractual" is a generic term used to indicate any document produced by the CEEDS "Contractual Documentation" facilities.

Characteristics of these documents include:

- having #a Pre-Specified Fixed Format#,
- #Revision Control#,
- Limited access (via #Menu Access Profiles#),
- Print logging,
- #Document Archiving and Re-Print of any previous revision.#

The documents don't specifically need to be used for contractual purposes, they could for instance be used as construction aids.

Each time a "Contractual Document" is issued or re-printed, a log entry is made of the Document Number, its revision, date and person requesting the printout.

Other Related Topics: #Documentation Classes#, #List of Documents#,

\*\*\*\*\*  
TOPIC Document\_Classes  
TITLE " Document Classes "

Any #Contractual Document# made known to CEEDS must be given a "Document Class". The registration of contractual documents within CEEDS is covered in the #Document Lists# topic.



The Document Class is a two character code, the first character giving an indication of the Subject Matter (e.g. L for LV Switchgear, C for cables, R for a Room, E for general Equipment etc.), and the second character giving an indication of format (e.g. S for schedule, B for Block Diagram, C is a Cabling/Termination Diagram etc.).

For Example: The Document Class LS will be an LV Switchgear Schedule,  
 CB is a Cable Block Diagram,  
 CS is a Cable Schedule,  
 VL is a room(volume) layout drawing.

Note: There are some Contractual Documents that CEEDS produces and some that CEEDS refers to (known as Internal (I) and External (E) documents respectively). Every Document Class must be classified as either an internal or an external document.

Attached to each Internal Class is the name of the software program that generates that document. This is known as the Report File, and it defines the Format of a given document. If it is an external document, this attribute has the value "No Files". Also attached to each class is the name of the "Banner" and "General Notes" Files. See #Report Layout# for details of these files, and of the general format.

Note: There is an exception to the well defined Document Class presented above. In the case of Rack Layout Drawings, a given drawing may include multiple "volumes" or a "volume" may be split across multiple documents. To allow for this problem, Rack Layout Drawing go from Class R0 - R9 allowing for 10 various combinations. A Rack itself has an attribute indicating a number from 0 - 9, which when combined with "R" will indicate the Rack layout Class of the Document in which it will appear.

\*\*\*\*\*  
 TOPIC Document List  
 TITLE " List of Documents "

Every Document / Diagram / Drawing REFERENCED by CEEDS (1), as well as every #Contractual Document# PRODUCED by CEEDS must be registered within CEEDS. All such documents are registered in the Document List.

(1) - This applies to documents whose "numbers" are not explicitly stored elsewhere within CEEDS.

A document is identified to CEEDS by a combination of "Document Class" and a "Search Criteria". This combination must be unique. The search criteria is used to identify documents within a given class.

Examples:

The LV Switchgear Schedule for Board 11BFA will have a document number of:

LS 11BFA



Search Criteria = Board Code  
Document Class = LV Switchgear Schedule

The Cable Schedule for the same board will be identified as CS 11BFA.

A Room/Volume Layout drawing for the volume identified by Structure Code 01UCB01 will have a document identity of VL 01UCB01.

Other attributes attached to a document are:

- Date of registration.
- Title: A description of what the document is.
- Two Drawing Numbers (DIS and KKS): For reference to other documentation registration systems within the organisation, namely Drawing Information System (DIS) and the Majuba Documentation System (KKS system).  
Note: If these numbers are provided, and are asked for on a document, then they will appear on that document. BUT if they are not entered in the Document List against a given Document, they obviously cannot appear on the document.
- The current working revision number (See "Revision Control").
- The Date at which the current revision was set, in other word the date at which the last revision was issued.

From a software point of view the following attributes are also defined:

- A parameter list; which is sent to the Report Program defined as part of the Document Class (Normally only the search criteria is given as a parameter)
- A "Data File" Name. The File which is created by the Report Program is given this name. The Data File Type is generated by the archiving system as part of Revision Control.  
(See #Document Classes#, and #Revision Control# for more details).

This Data File Name follows the following conventions:

- \* The name must be 8 characters in length.
- \* The first two characters are the Document Class.
- \* The Next 6 characters are the first 6 characters of the Search Criteria. If the Search Criteria is less than 6 characters, pad with 'X's.
- \* The last character is used as a unique identifier to resolve possible duplications that may occur within the first 7 characters. If no resolution is necessary, use an 'X'.

Examples:

For document LS 11BFA, the Data File Name would be LS11BFA  
For document VL 01UCB01, the Data File Name may be VL01UCB1 and  
for document VL 01UCB02, the Data File Name may be VL01UCB2 etc.

NOTE: A File name of 8 characters could create quite a severe restriction in trying to resolve duplications as using the full alphabet and all digits, only 36 conflicts can be resolved for any given conflict. BUT if the search criteria is used correctly, it should seldom have to exceed 5 characters. This is only a restriction in the Mainframe environment.

#### Search Criteria for some common documents:

Switchgear Schedules: 5 Character Board Code  
 Cable Schedules: 5 Character Board Code  
 (data grouped by Contractor within a document)  
 Equipment Lists: 2 Character Contract Area Code.  
 Rack Layouts: 5 or 7 Character Structure Code (defining a volume).

The Contractual Documentation "Revision Change" and "Re-Print" facilities are based on this Document List. In other words, when selecting a document for issue, it is selected from this list.

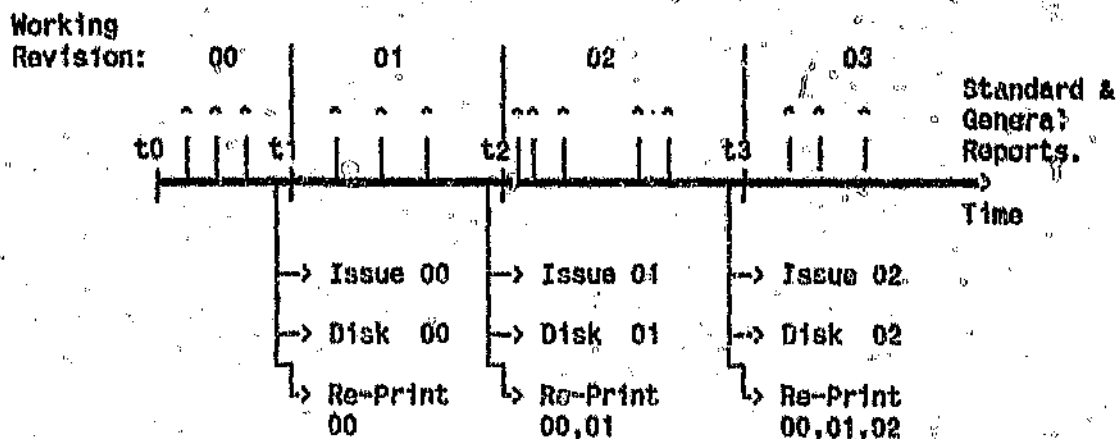
Note: The Revision Change and Re-Print Facilities naturally only work on documents whose Document Class indicates that they are Internal documents.

See #Re-Printing of Contractual Documents#.

\*\*\*\*\*

TOPIC Revision Control  
 TITLE " Revision Control "

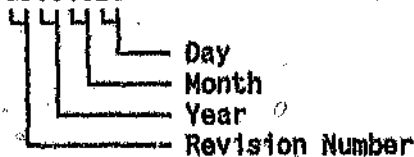
The Revision Control Philosophy used in CEEDS for #Contractual Documents# differs slightly from conventional Drawing Office Procedures. Consider the following time representation:



### General Procedure:

- At time t0, data begins to be entered into the system.
- #General# and #Standard reports# can be produced at any time thereafter.
- Once it is decided by the relevant responsible people that a "Contractual Document" needs to be created, then that document is selected from the "Contractual Documents List" and it is "Issued". This takes place at time t1.
- The action of "Issuing" a Contractual Document from CEEDS involves the following steps:
  - \* The program that creates the document is activated and a document file is created, (See "Document Classes")
  - \* This file can then be printed to obtain a hard copy,
  - \* The file is then copied to a CEEDS archive disk. This file is now safe and is available for #re-printing#.
  - (This file is given the File Name provided with the document details when a Document is registered. The File Type is generated and indicates the Document Revision and the date it is archived.

The File Type format is: 00901023



Note: A list of Archived Document Files is available from the Contractual Documents Menu.

- \* An entry is made into the Contractual Document Print Log indicating which document was issued, its revision, the date of issue/print and the person requesting the issuing of the document.
- \* The current working revision number for this document is incremented by 1.
- This whole cycle then repeats itself for subsequent revisions.
- For each CEEDS produced Contractual Document, a corresponding "Content Change Report" can be produced.

### General Notes:

- General and Standard Reports can be produced at any time. There are no restrictions placed on these reports.
- As soon as a contractual document is "identified", i.e. registered in the system, it has a WORKING Revision Number of 00 by default. This can be reset to take into account any existing revisions that were produced by other means (such as CEEDS 1).

For more details, see: #Document Classes# and #Document Registration#

\*\*\*\*\*

TOPIC Re\_Prints

TITLE "Re-Printing Documents "

All Contractual Documents are stored on a CEEDS archive disk, (See #Revision Control#). From this area, any revision of any document already produced can be re-printed using the contractual document re-print facility. All one needs to do is select the desired document and provide the desired revision number. (Note that this revision number can only go as high as Working Revision - 1)

The Re-Print Facility is available in the Contractual Documentation Menu.

It may happen that the Contractual Document File that is being requested for Re-printing has been removed from the CEEDS Archive Disk and has been backed up onto Tape. If this happens, you will be informed that the request will take a while (approximately half an hour) because the Mainframe operators must recover the File from tape and re-instate it onto the disk. Hereafter the procedure carries on as per normal. (This may happen in the event of trying to re-print a very old document that was removed from the Archive Area to conserve disk space)

\*\*\*\*\*

TOPIC Report\_Layouts

TITLE " Document Layouts "

The format of one #Class# of contractual document may differ substantially from that of another. The format of any Class is specifically customised to meet the requirements of the users of that document. There are some basic guidelines though.

All CEEDS produced Classes of Documents will have:

- 1) Banner/Administration Page:

This is the first page of a report and has an enlarged Title, an Administration Block (for signatures etc) and possibly a header or foot which is common throughout the complete document.

Other than the Header or Footer, the contents of the Banner Page are stored in a Flat file (the name of which is attached to the Class, and is the same as its corresponding General Notes File). This file has a File Type of "BANNER".

- General Notes Page(s):

These page(s) contain explanation and contractual notes which are applicable to ALL specific documents within a Class. I.e. they are notes explaining the contents of the Class of Document.

For example the notes in this section will be applicable to ALL LV Switchgear Schedules irrespective of which board they are for.

This page may also contain a Header or Footer in common with the rest of the document.

Other than the Header or Footer, the contents of the General Notes Page are stored in a flat file (the name of which is attached to the Class, and is the same as its corresponding Banner File). This file has a File Type of "GENNOTS".

- Specific Notes Page(s): (Optional)

Within the CEEDS #Model#, there is the facility to attach "Notes" to each object. Notes attached to object(s) within the scope of a given document can be extracted and printed here.

- Revision Notes Page(s):

Along with each new revision of a document, there must be a set of notes describing the changes made to the contents of the document since the previous revision.

CEEDS has the facilities to attach "Notes" to any Document identified in the #Document List#. Revision notes are entered (using these facilities) against a specific document. The compilation of revision notes can be greatly aided by a #Change Report#.

(See #Revision Control# for more details).

- Summary Information Page(s): (Optional)

Often a document may contain a large quantity of detailed information and sometimes it is quite useful to have summary information of some sort. (But this is determined by the function of the document).

- Report Body Page(s):

This is simply the bulk of the document containing the main information of the report. The format of this section is specifically tailored to suit the purposes for which the document was designed.

### A5.3 Cable Sizing Algorithm

One of the interesting features of the CEEDS algorithm is that cables are optimised on cost and not just technical criteria. The core area required to carry the desired load is determined based on technical requirements (Refer to the "Cable Sizing Program" below). The old manual selection method employed a "cable selection graph" to determine a suitably sized cable. Normally, the first cable to meet the requirements was chosen, but it is possible to run two smaller parallel cables rather than one large cable. CEEDS explores various possible combinations of cable sizes and by querying the Cables Financial model, the capital expenditure cost of the cables is determined (including labour, terminations, etc). When a minimum cost is found, that combination is recommended.

Investigations are planned to determine optimum cable sizes based on life time costs. i.e. capital expenditure is weighed up against electricity losses in the cable over the life time of the station. It is possible that a larger cable and therefore more expensive, could end up being cheaper in the long run due to lower losses. There is also a trade off between having a limited range of cables to choose from to prevent stock proliferation and being able to select the optimum cable from the widest possible range.

The following documentation was written to accompany a PC based product which was developed in conjunction with the actual CEEDS sizing facilities. The two programs are very similar except that CEEDS does cost optimising whereas the PC package selects purely on technical attributes. It is adequate though to demonstrate the technical selection process.

#### CABLE SIZING PROGRAM.

Author: Johan Plenaar.

#### INTRODUCTION.

The PC based cable sizing program was developed for use within PSEED as an aid for the sizing of power cables. The program was developed specifically for Majuba in the process of developing the CEEDS sizing program. If there is a need for the program to cater for other stations it can however be modified without difficulty by creating the cable reference file for that station.

The purpose of the sizing program is to match the cable type selected from the available types to the load that must be supplied in such a way that the minimum specified criteria are met with the most economic cable type for that purpose.

The cable sizing program is based on the same concepts as the CEEDS 1 program written by Andre Kotze. The major refinements to the program is the calculation of a derating factor based on inputs of temperature, depth in ground, and number of parallel current carrying cables.

The sizing program will only select cables specified in the sizing reference file (at this stage only for Majuba - MAJUBA.REF)

#### REFERENCE FILES.

The maximum current for every cable is given for 3 cables running in parallel in the cable reference file, these values are extracted from ESKOM Standard MBIM 401, and are maintained by the cable specialist.

All default values can be adjusted to ensure that the derating factor is calculated more accurately if the conditions deviate from the defaults given. There should however be very few exceptions.

## **SIZING PHILOSOPHY.**

The cable sizing is done in a similar fashion as would be adopted with the manual sizing of cables using the volt drop curves by evaluating the following variables.

- Air temperature (Default 30 C);
- Ground temperature (Default 25 C) if the cable is run in ground (indicated by the air/ground flag);
- Ground resistivity correction factor for cables run in ground;
- Additional derating factor that can be used to manually override the calculated derating factors if required;
- The nominal voltage dictates the type of cable that will be used;
- The design length that is used to calculate Volt drop;
- Full load current to calculate volt drop and for minimum cable size selection;
- Starting current to calculate starting volt drop;
- Air/Ground flag for selection of unarmoured/armoured cables and the correct current tables;
- Connection type to select the number of cores and voltage type;
- The % volt drop allowed;

The derating factor is calculated and is then applied to the maximum currents as specified in the current tables.

The program starts off by selecting the correct set of maximum current tables for the specified voltage grade and application combination.

It then starts with the smallest cable and compares this maximum current with the full load current, if a suitable cable is found the percentage volt drop is calculated and if it exceeds the maximum allowable % volt drop the next bigger cable is selected and tested until one is found.

Now the starting volt drop is calculated and if it exceeds 15% of the nominal voltage a bigger cable is selected until this criteria is also satisfied.

If no solution is found an additional cable is added in parallel, the derating factor is recalculated and the process is repeated.

The program will return the number of cable groups that can be used as well as the maximum length the cable can be extended to without exceeding the volt drop limits.

It will also give the volt drop and % volt drops on which the selection was based as well as the calculated derating factor.

The results can be printed on a line printer and the number of parallel groups can be increased. (A cable group can be seen as a set of cables that logically belong together as a single entity, normally a multicore cable is considered a group but if three single core cables are run in a close trefoil formation this also forms a single group).

The sizing program used in CEEDS2 will actually select the most economical combination of parallel cables that satisfies the minimum criteria.



## CONCLUSION.

The sizing program should size the majority of AC power cables correctly according to the criteria stated above. For some DC loads, other exceptional cases and deviations from the given criteria, the cable specialist must be consulted.

Costing is not done in the PC based version because of the amount of maintenance required to the data to keep it current, especially if other stations are included later on.

The program was written by J. Plenaar in PSEED (x3982) and can be used by any person who feels that they can use the software.

Copies of the software can be obtained from the author and any suggestions will be welcomed. Improved versions and reference files for other stations may be made available from time to time.

## A5.4 Cable Routing Facilities

The routing facility consists of a routing program that communicates with CEEDS via temporary database tables. A "preparation" activity is undertaken in which desired Racks and Cables are selected from the CEEDS tables into these temporary tables. The routing program then routes the selected cables on the selected Racks. The main tables are then updated in one process. This method allows for the most flexible use of the facilities. The following extract from CEEDS documentation describes the Routing Philosophy in more detail.

### ROUTING PHILOSOPHY.

Author: Johan Plenaar.

The routing program is divided into 5 large areas that perform different functions.

1. Initialize the network matrix.
2. Find the segment closest to the end of the cable.
3. Find up to 6 segments closest to the start of the cable.
4. Find the shortest route through the matrix from the start segment to the end segment.
5. Calculate the filling of the segments the cable is routed on and reinstate the matrix to its original state.

Steps 2 to 5 are repeated for every cable and steps 4 and 5 are repeated if one of the segments in the optimal route is full before the matrix is reinstated.

For routing purposes a node is defined as a point in space where two rack segments meet, therefore segments can be seen as connecting two nodes.

The heart of the program is the two dimensional adjacency matrix ( $adj[i][j]$ ) that defines connectivity between node  $i$  and  $j$  in the network. If  $adj[i][j] = 0$  it means that a segment exists that connects node  $i$  to  $j$  and the value of  $adj[i][j]$  is the record number of array racks[] where this segment is defined.

Racks[0] contains a segment with infinitely long length and a function called `segle()` is used to extract length of the segment from racks in the function that finds the shortest route.

Another array called `node[]` is used to store the coordinates of every node in the network.

Refer to Figure FA5.4.01 for details on the Matrix layouts and relationships.

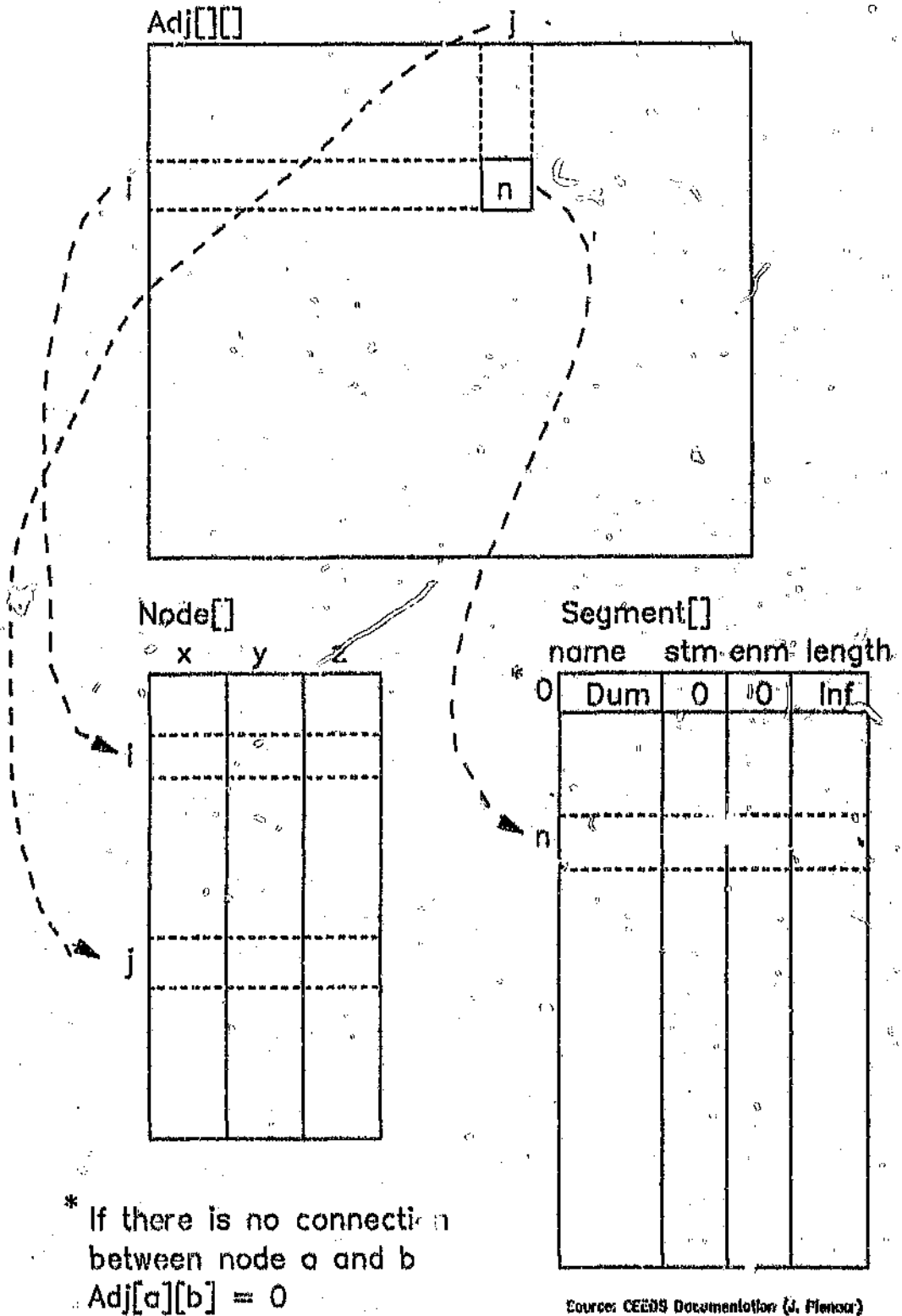
A segment always starts at  $i$  and goes to  $j$  in the convention used in the routing program, if the segment goes in the other direction the start and end meterstones is reversed by the program to ensure that the meterstones are correctly calculated for output to the routes table (CBLRTE). Refer to Appendix A5.7 for details on Meterstones

#### 1. INITIALIZE NETWORK MATRIX.

Initially every  $adj[i][j]$  is set to 0 to ensure that no nodes are connected.

Every segment that must be considered for routing is now read from the temporary Rack segments table.

If the start node for the segment already exist (the x y and z coordinates of the start of the segment already occurs in `node[]`) this node is taken as the start node of the segment, otherwise a new node is created.



**Source:** CEEDS Documentation (J. Flenoy)

ALDI<sup>®</sup>

## Cable Routing Program Matrix Structures

**Figure FA5.4.01**

This process is repeated for the end node.

This new segment is now added to racks[] and adj[i][j] and adj[j][i] is set to the index of the new segment. A check is performed to ensure that the directional conventions are adhered to.

Record is kept of the total number of nodes and segments in the initial matrix.

## 2. FIND THE END NODE.

(Node in this context refers to Rack Nodes, not Equipment Nodes).

The end rack is found first because there is a greater chance of not finding the end rack in the network because of lower rack densities at the plant side of the racking design. If an end rack is not found the routing process is stopped and the next cable is attempted.

To look for the closest rack every node is visited.

If the end of the cable coincides with the position of a node that node is used as the start node of the cable.

Otherwise the program creates a new node.

It then checks if there is a connection between two nodes i and j and if so a point  $x_i, y_i, z_i$  is found such that this point is the intersection of the extended centre line of the segment and the shortest line from the cable end point to this extended line.

If point  $x_i, y_i, z_i$  falls on the segment (between nodes i and j) the slack at the end point is calculated from the new node to  $x_i, y_i, z_i$ . Otherwise the slack is calculated from the new node to the end of the rack where the cable will leave the rack.

The slack is calculated in the same way a cable would be installed on site. The cable moves vertically down until it is horizontal with the position where it will enter the segment and then is bent through 90 degrees and moves horizontally directly towards the segment.

$$L = \text{abs}(z_1 - z_2) + \text{sqrt}((x_1 - x_2)^2 + (y_1 - y_2)^2)$$

$x_1, y_1, z_1$  - position of cable start point.  $x_2, y_2, z_2$  - position where cable enters the segment.

If the shortest distance towards any segment in the network exceeds 5m an error message is given that no segment is closer than 5m from the cable end point and this cable is not routed.

When the closest segment has been found a new segment is created that extends from the cable end point to the position where the cable will leave the rack network. If this point falls on a segment between two nodes the original segment is divided and a new node is included in the network matrix.

## 3. FIND THE START NODE.

Finding the start node is very similar to finding the end node except in this case the 6 closest segments are considered and included into the network.

This is done by first finding the closest segment, replacing it with two new segments to the new node and repeating the process to find the next closest segment. Because the previous segments are now excluded from the network they will not be found again during the next iteration. Segments to the new node are also not considered by only examining the original extent of the matrix.

To find the start segments this function only considers segments that pass within 1 m in the x and y (horizontal) direction of the x and y coordinate of the cable. The vertical height is not considered although the cable slack may not exceed 20m.

#### 4. FIND THE SHORTEST ROUTE.

This program is based on the Nicholson Method [MDL 26] to find the shortest route between two nodes in a network.

The Nicholson algorithm is very efficient for this type of routing problem in that it searches from both sides (the start and the end node) and it terminates as soon as the conditions for the shortest route are met thereby reducing the number of iterations considerably.

#### 5. UPDATE RACK FILLING.

If a route has been found for the cable the % filling of every segment is updated in the Rack Segments Table (RACKSG). If the filling of the segment exceeds the allowable filling (as listed in the Rack Loading reference Table), a warning is given but the cable will still be placed on that route. It is up to the racking and cabling design team to decide if this cable should be unrouted or not.

If the filling of the segment exceeds 100% when the cable is placed on the segment that segment is excluded from the matrix and another attempt is made at routing the cable.

#### 6 FUTURE ENHANCEMENTS.

If it is found that the size of the matrix is still limited due to memory constraints, methods could be investigated to change the addressing of the matrix to exploit the triangular structure of the matrix.

The methods currently used should give an improvement on the limit that was encountered on CEEDS 1 but further improvements would mean a reduction in speed.

Another modification that will be investigated is the specification of intermediate nodes in the network that the cable must pass through.

## A5.5 CEEDS Batch Integrity Checks

There are a number of integrity and design validation checks that are available and a couple that are still in the pipeline. These checks fall into the following categories:

- **Reference Table Integrity checks.** Reference information for a given class can often consist of a number of related tables. Link integrity between all these tables must be checked. These checks do not directly involve the model data and will be used by the specialists responsible for reference data.
- **System Software Integrity checks.** There are a number of tables used to describe the software system. These contain information such as the standard reports list, menu options, user information, file configuration information, model structure information, etc. An un-coordinated modification to some software could place the system integrity at risk. Problems due to the breach of integrity may not be discovered for some time and it may be too late by then. These checks don't involve the model data and will be used by the system developers and maintainers.
- **Model Based Checks:**
  - **Documentation checks:** Documentation is often dictated by the model. For example there should be a Switchgear Schedule and a Cable Schedule for each Board in the system.
  - **KKS Code verification checks:** Detecting codes not adhering to KKS rules.
  - **Dictionary Checks:** Checking the validity of "dictionary" codes in the model. For example, are all structure codes used in the model valid according to the Structure Code Dictionary?
  - **Object Integrity:** This is a very important check. If an object is not stored correctly, severe errors can occur.
  - **Model Statistics Report:** The number of Nodes, Links, Racks, Boards, Switchgear, Motors, etc. can be shown. Other statistics such as the number of Structure codes, abbreviations, contract areas, documents, etc. are also useful. Percentage completion reports are also very valuable, for example, for cables, the % sized, % routed, % pulled on site, etc. are used for direct indication of progress.
  - **Connectivity Integrity checks:** These checks will be used to ensure that the Macro level connectivity and the cable level (Mid level) are compatible and will also indicate those nodes not yet cabled up.
  - **Detailed Design Integrity checks:** Separate very detailed analysis can be carried out on various design aspects such as "Switchgear" design, "Cabling" design, "Transformer" design, etc. These reports would contain quite a lot of intelligence in that they effectively determine possible "faults" in a design.

All of these reports should be available in "Summary" format and "Detailed" format. Summary reports should only contain statistics of integrity violations and detailed reports should contain lists of the "guilty parties".

## A6.1 CEEDS Naming Conventions

This appendix consists of edited extracts from the CEEDS development standards.

The main components in CEEDS are Operating System Files and ORACLE Database Tables.

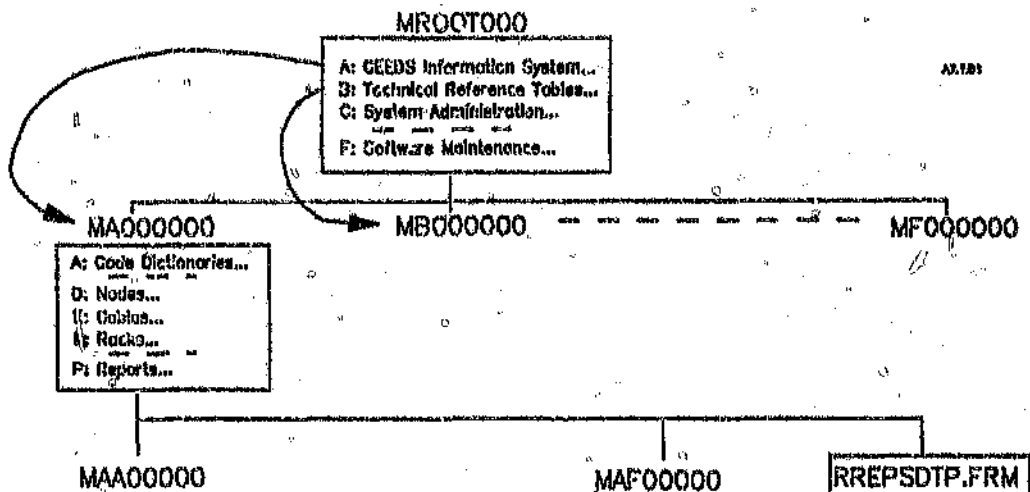
### A6.1.1 File Naming

Files have a File Name (maximum 8 alpha-numeric characters), and a File Type (maximum 8 alpha-numeric characters). The IBM Virtual Machine Operating System (VM) requires that a File Name start with an Alpha Character, and must contain no blanks. File Names follow a CEEDS convention, whereas file Types are usually dictated by the software using them. For example FORTRAN source files have the File Type of FORTRAN and operating system executable "script" files have the type of EXEC, ORACLE Screen Form files are typed INP for source and FRM for compiled.

- Any file name must be 8 characters long. If, by following all other rules, a name is not 8 characters long, it will be padded with "X's" or a number sequence.
- The first letter of a name indicates the function of the file. For example, M - menu files, R - reports, S - screen forms, X - data transfer routines, P - data processing routines etc.
- The remaining seven characters depend on the category of file, i.e. whether they are Menus, Screen forms, etc.

#### A6.1.1.1 Files defining the menu structure

- The first letter is an 'M'.
- The file name of the root menu is MROOT000



CEEDS Menu Structure Naming Convention.

Figure FA6.1.01

- The menu called up by option 'A' in MROOT000 is MA000000, and the one called up by option 'A' in MA000000 is MAA00000; etc. Refer to Figure FA6.1.01 for an example.
- 'Q' is used to pad the name to eight characters long.
- Menu options were restricted from 'A' to 'P' due to screen space. 'Q' always being "Return to previous menu" and 'R' being "Return to ROOT menu. Refer to Figure 7.1.05 for an example of a CEEDS menu.
- Option 'P' is normally reserved for the "Reports..." option.
- The number of levels was restricted to 7 as a result of the file length. (This was not found to be a practical restriction).

#### A6.1.1.2 Files defining Screen Forms

- The first letter is an 'S'.
- The next 6 letters are those of the main table displayed in the form. (Data could be extracted from other tables for display purposes, but usually, a screen form is based on only one table).
- The last letter gives an indication of the actions that can be carried out on the form.

For example:

V - view only,

I - Insert only,

C - Capture (Insert and update inserted record only),

D - Delete only,

U - Update only,

M - Maintain (full access).

Refer to Figure FA6.1.02.

#### Privilege

- I - Insert only
- V - View only
- U - Update existing data
- D - Delete records only
- M - Maintain (full privileges)
- etc.

Table Name  
(Refer to Figure FA7.1.04)

"S" = Screen Form

Screen form name:

S						
---	--	--	--	--	--	--

Examples: S B O R D N T M = Maintain Board Notes

S C B L S D T V = View Cable Details

AF.1.02

## CEEDS Screen Form Naming Conventions.

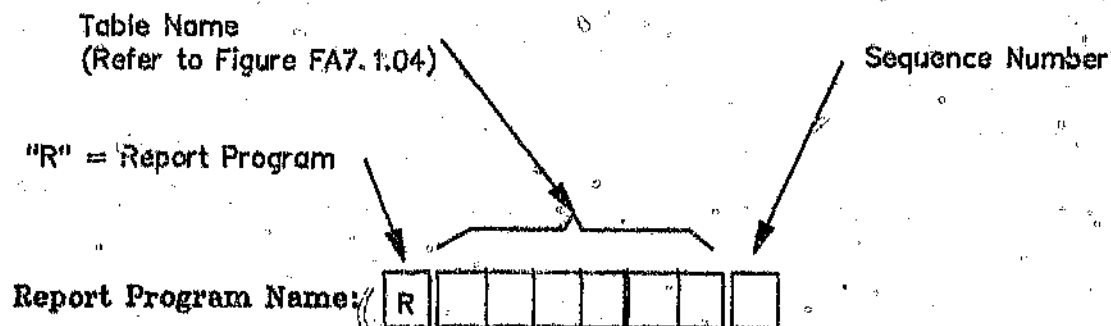
Figure FA6.1.02



- See "Table naming" below for how the table name is derived.
- A Screen forms name is displayed along with the form to firstly act as a reference in the case of errors, secondly to indicate the table name being worked on and thirdly, to give the indication of privilege level.

#### A6.1.1.3 Report Files

- The first letter is an 'R',
- For non-contractual reports, the convention is the same as for screen forms, except that the last letter is a sequence number.



Examples: R B O R D N T 1 = Board Notes format # 1  
 R C B L S D T 3 = Cable Details format # 3

### CEEDS Report Naming Conventions.

Figure FA6.1.03

- For contractual documents, the second and third letters indicate Document Class, the fourth to seventh indicate equipment class and the last is a sequence number.

See Figure FA6.1.03.

#### A6.1.1.4 Other files

Naming for other files is less restricted.

- The first letter indicated function,
- the next 6 indicate the table being worked on (if possible), otherwise a meaningful compromise is made,
- the last letter is usually a sequence number.

#### A6.1.2 Table and Column Naming

Table Names and the Column Names within these tables were set at 6 characters, to comply with a FORTRAN limit on variable names, and to match in with the file naming conventions above.

#### A6.1.2.1 Table names

- The first four characters indicated the equipment class being modeled. For example: CBLS - Cables, RACK - Racks, SWDC - DC Switchgear, MOTR - Motor, etc. These equipment classes are exactly the same as those used internally within the model. Common Nodes data is classified under NODS.
- The fifth and sixth characters indicated the aspect being modeled. For example, DT - Details, NT - Notes, TP - Temporary, RM - Reference Main (Type), CR - Cost Reference, etc.

Refer to Figure FA6.1.04

#### A6.1.2.2 Column names

#### Equipment Class

Boards  
DC Switchgear  
LV Switchgear  
MV Switchgear  
Cables  
Transformers  
Racks  
etc.

BORD  
SWDC  
SWLV  
SWMV  
CBLS  
TRFR  
RACK

#### Table Function

DT Details  
NT Notes  
DH Details History  
CO Change Capture  
RC Cost Reference  
RT Type Reference  
etc.

Table Name:

--	--	--	--	--	--

Examples: B O R D D T = Board Details

A6.1.04 B O R D N T = Board Notes

## CEEDS Table Naming Conventions

Figure FA6.1.04

Column names are made as meaningful and consistent as possible within the constraints of 6 characters. Standard abbreviations are used, for example a cable is CB, a CBL or CBLs (not a CAB or CABL), depending on the number of characters available.

CEEDS 2

MROOT00  
A3931911-MAR-91  
14:54:20

## MAIN ROOT MENU

- \* A = CEEDS Information System...
- \* B = Technical Reference Tables...
- \* C = System Administration...
- \* D = General Data Queries...
- \* F = Software Maintenance...
- \* G = Information About CEEDS...

Q - Quit to EXIT

R - Return to EXIT

PF15 - GENERAL QUERIES PF16 - MENU QUERIES

Select a menu option : \_

\*- Block 1/1

Page 1

Count: \*0

CEEDS 2

MA000000  
A3931911-MAR-91  
14:55:00

## CEEDS Information System...

- \* A = Code Dictionaries...
- \* C = Terminations / Signals...
- \* D = Components (Nodes)...
- \* E = Cables & Network...
- \* F = Racking & Routes...
- \* I = Design Activities...
- \* O = Site Facilities...

Q - Quit to Previous Menu

R - Return to Root Menu

PF15 = GENERAL QUERIES PF16 = MENU QUERIES

Select a menu option : Q

\*- Block 1/1

Page 1

Count: \*0

SYS\_AVAIL MY\_JOB

## CEEDS Menu Layout Examples.

Figure FA6.1.05

## A6.2 Future Plans for CEEDS Development

The following is an extract from a memo sent to Power Station Electrical Engineering Management in October 1990.

### FUTURE PLANS FOR CEEDS DEVELOPMENT

#### Synopsis:

As it is now approaching the time to begin implementing CEEDS 2 within the division. It is felt that the time is right to take a look into the future of CEEDS. There is still massive potential to be gained from the proper application of Information Technology within the division, and in a wider context, within all parties with which the division interacts in carrying out its business.

The purpose of this document is to:

- lay down some thoughts as to possible future scenarios for CEEDS,
- to provide some of the technical details,
- to make other people aware of where we plan to take CEEDS (or at least, where we would like to see CEEDS going),
- to attempt to get management approval for continued development along the lines outlined below,
- to highlight the importance of various groups taking responsibility/ownership of various CEEDS facilities,
- to emphasize the need for communication between the various disciplines in Engineering and lastly, and most importantly
- to try establish an excitement within the division (not a false, misinformed expectancy), so that people realise that CEEDS is their tool for improving the performance of the division. It is not a political lever, an empire building exercise, a job destroyer, etc.

Various relevant aspects will be detailed, including:

- What the CEEDS objectives are and why these objectives need strong support by all parties involved,
- Possible future development phases for the CEEDS product,
- Resources required for future development (People, Knowledge and Technology), and
- some general observations regarding current shortcomings and possible future applications and marketing potential.

#### Development Objectives and End Goals:

In order to carry out a development with the magnitude (and possible importance) of CEEDS, very clear objectives and goals need to be established quite far in advance.

The end goal of CEEDS is to provide an Integrated set of Computer Based Tools to aid the division in carrying out its every day activities of designing and building electrical and control systems for Power Stations and other plant.

This end goal is obviously broken down into further detailed sub-goals, and the work required to achieve these sub-goals must be determined. This will not be elaborated upon here.

The point that must be emphasized though, is that these objectives need to be strongly and continually maintained and supported so as not to lose momentum and motivation in our striving towards the end goal. When engineering sets out to design and build a power station, the operating Power Station is the end goal and everyone should have a very clear mental picture of where they are heading in order to reach the goal within given constraints. Without this conceptualisation, chaos would be the result of any large scale effort that has no fixed end goal. It is no different in the designing and implementation of large scale software systems. It is for this reason that details are given below of how it is envisioned to continue with the evolution of CEEDS. Management approval of these (or other negotiated) objectives are essential for the continued development of CEEDS.

To plan, budget, attract qualified personal and to keep a qualified team together, requires the approval and commitment from management and the users of the team's products. Without this approval and commitment, there can be no motivation or dedication within a development team and the capabilities of the team will degrade to only be sufficient to keep current systems functional. Incentive and creative drive will be lost.

#### **CEEDS Development:**

##### **- CEEDS 2:**

CEEDS 2 is a VM mainframe based system with textual model manipulation and with little built-in "intelligence". (Design Procedures that are catered for, are hard coded into "Physical Model Manipulation Programs"). The introduction of the Model Concept is not entirely suitable for text based mainframe systems, but it is going to pave the way for the next phase, namely CEEDS 3.

With training programs scheduled over the next couple of months, many people will be introduced and trained on CEEDS 2, so details of what the systems' current capabilities are will not be elaborated upon. What is outlined are the planned activities on the current system over the next couple of months.

- The system needs to be tested and approved by its various "owners" (such as the design team, the specialists, the drawing office, systems engineering, projects, etc.). Once minor alterations are made to bring the system into accordance with user requirements, and representative users have signed off acceptance certificates, the complete system will be placed in production. Only then will the system be handed over to IT for maintenance.
- The system needs to be documented. Documentation that is planned includes:
  - 1) CEEDS concepts and philosophies
    - modeling aspects,
    - configuration management,
    - using the system,
  - 2) Divisional procedures to carry out activities that make intensive use of CEEDS (such as switchgear design, cable design and administration)
  - 3) Technical documentation - hard copies of source code,
    - maintenance philosophy and procedures,
    - testing procedures,
    - IEW documentation.
  - 4) Registration of all source code into the new IT source code management system (CCC). This will be coordinated with the placing of the system into production.
- Users need to be trained on the system. This is going to require some dedication from users, especially those people who have a busy schedule. It will also be an on-going task for which certain people will have to be specifically trained.
- The Financial Model needs to be analyzed and expanded. Firstly the Racking Cost Reference Database needs to be created and included in the system. This will then at least provide the ability to do accurate Cable and Rack Costing. Facilities for:
  - the inclusion of budgetary figures and comparison to actual figures;
  - forecasting cash flow and invoicing rates;
  - invoice control and validation;
 still need to be analyzed and incorporated into the current system.

A Financial Model for Switchgear Costing can also be considered.

- The system must be experimented with to improve and optimize its performance. Many concepts currently incorporated into CEEDS have never been tried and tested in a production environment. The real world has an abrasive nature that tends to dull the crystalline structure of theoretical models.
- The scope of the equipment covered must be expanded to cover a wider field. Areas that still need attention include:
  - \* DC equipment (batteries, charges, UPS's, etc.);
  - \* Protection equipment;
  - \* C&I equipment (instruments, PLC's, Control Panels, etc).
- Now that the Physical Model is being introduced and fine tuned, attention can be turned to improving analysis facilities. This would include providing tight coupling of data between CEEDS and packages such as PSSU/PSSC (electrical system analysis packages). Another aspect that has received some attention, but could be greatly improved is the area of data integrity checking and data "availability" analysis.

Functions could be developed that automatically detect possible problems in the coding and describing of objects.

- PC Based data capturing facilities for in-house use and more specifically for contractors have been developed. Once these have been tested and modified to meet users requirements, the personnel responsible for contract management will need to introduce the software to their respective contractors and encourage them to use the system. Currently the capturing of Nodes and Cables have been catered for. Once the PC based software is in use by contractors, and they understand the larger concept of CEEDS, then contractors' requirements can specifically be investigated to make the ESKOM/Contractor information interchange smoother.
- There is also the possibility that contractors can make direct use of CEEDS using Eskom equipment. Comparison facilities for comparing CEEDS data to data contained in other systems maintained by large contractors (such as Siemens) could be provided.
- The Design Procedure Model needs in-depth analysis and must be formalized. Rack design is currently aided by software developed as part of the CEEDS project (CREG - Cable Racking and Equipment Digitising), but is not considered to be an inherent part of CEEDS. CEEDS imports a Rack Design from the CAD system and considers it to be correct and approved.



What still needs to be done is in-depth research into suitable representational schemes for the design procedure model and an analysis of the larger scope of design undertaken by the division.

The benefits of such a study are that the whole operation of the division can be analyzed from an information flow and information sharing point of view. The design process can be formalized (and yet still be flexible). This should reduce the divisions dependency on a few key staff members that hold everything together. Another benefit from the formalisation of a Design Process is that it can be incorporated into the underlying information system (namely CEEDS) and therefore provide additional functionality and "intelligence" to the design tools.

#### - CEEDS 3:

The CEEDS 2 system will migrate over to a networked workstation and PC based system using the mainframe only for mass storage and country wide access. This stage involves removing the barriers between CEEDS and CAD so that they become one and the same. The model will have a graphical front-end and will allow direct graphical model manipulation. The "Design Procedure Model" will be a separate entity and will control "Physical Model Manipulation Programs" that would in turn work on the "Physical Model". The separation of the Design Procedure model as a self standing entity will be to pave the way for the next phase (CEEDS 4). (This separation of the Design Model may even be implemented on CEEDS 2 for software maintenance reasons). Included below are some points relevant to the proposed further development of CEEDS to a next generation system:

- The hardware environment will be different in that the complete CEEDS system will be spread over various systems. (See Hardware resources below). It is proposed that initially, the design functionality of CEEDS will migrate to a UNIX based workstation platform. This environment will allow for the integration of CEEDS functionality with CAD functionality in a stable homogeneous environment (Not possible on the current mainframe system). Following this, more and more CEEDS functionality will migrate to the graphical environment and eventually the mainframe will only act as a "model database server" and wide area communications medium.

This migration will need to take place in a number of well defined stages that are dependent on numerous outside factors (many of which are dependent on the IT department). The stages would probably follow a scenario as outlined below:

- 1) Development of basic graphical design facilities (See next point below) on a single stand alone UNIX CAD Workstation. Many new concepts and facilities will need to be developed and learned to achieve this stage. This will place a high learning curve burden on all development staff involved.
- 2) Integration of the latest CEEDS 2 design within this new graphical functionality. (Still on a single workstation). This may require the rewriting of large sets of CEEDS 2 procedures as these were tailored for use on the mainframe and were written in FORTRAN.
- 3) The simultaneous investigation of the use of medium and high range PC's as CEEDS terminals in the new environment.
- 4) The linking of two workstations via an available network to develop and test the facilities required for multi-user capability. This will incorporate a whole host of new concepts and software such as distributed database servers, file sharing, communication protocols, security etc. The network will also need to be evaluated for speed and functionality.
- 5) The introduction of cheaper PC's into the network to expand the user base and to develop and test the facilities required for distributed processing (whereby a program executes on a workstation, but uses the PC as its front-end hardware). This allows the use of a sophisticated system on a hardware platform consisting of a few expensive workstations and a horde of very much cheaper PC's.

During this development CEEDS 2 will continue with production un-disturbed.

- 6) The final stage will be to incorporate the mainframe as a network node. This will allow the mainframe to act as database server. The result will be a combined CEEDS 3 graphical system and a CEEDS 2 textual system.

After this stage the environment will be established for a full scale migration of functionality off the Mainframe as described above.

- The major development activity on CEEDS 3 (from a software point of view) will be to provide a means of interfacing to the various models in a graphical fashion. Specifically the physical model.
- Another development will be to provide a consistent, standard, point & click interface operating within a standard windowing environment. This will have the benefit of reduced training requirements because any extensions to the system

must be provided with this standard interface and therefor requires training only on the concepts, not on the detailed operation.

(The Standard referred to here is the emerging de-facto standard of X-Windows with the Open Systems Foundation Windows manager (a Microsoft Windows/Presentation manager look-a-like, feel-a-like)).

- If it proves too difficult to implement the design process model on CEEDS 2 in the mainframe environment, then it will be done as part of this project.
- Finally, the software development approach can totally move over to Object-Orientation in which all attributes (and certain functionality) of an object (entity) reside with the object.

#### - CEEDS 4:

Part of the on-going evolution of a system is that it grows to incorporate leading edge technology as that technology becomes commercially available and viable. AI techniques are beginning to emerge from research laboratories and are being applied to the solving of real-world problems. Part of the CEEDS 4 "project" will be the investigation of potential application areas of this technology within CEEDS. Large learning curves will again be major obstacles and for this reason it is proposed that full time study in this particular area be undertaken.

The full potential of AI technology has not yet been investigated in any depth but some preliminary thoughts include the following:

The CEEDS 3 system will gain intelligence in the form of

- Background "Truth Maintenance" systems that ensure data integrity and exception handling.
- Design Rule verifiers, conformance to contract verifiers etc. that are tied to the Design Process Model(s).
- Front end Expert systems aiding in a variety of daily activities from Design activities to invoice control. Design activities could include aids for Switchgear selection, motor and transformer sizing, selection of appropriate protection equipment and strategies, cable selection and design, intelligent cable routing and equipment placement, network stability & reliability optimization and many more.
- Diagnosis of many aspects of the model can be carried out by AI techniques that could not be attempted by conventional, algorithmic methods.
- Intelligent documentation production facilities could be

developed that would be used to generate quite sophisticated reports and documents (especially those of a graphical nature). For example, system block diagrams could be entirely generated by computer for any given set of search criteria.

- Interaction with other systems:

An activity which can take place independently of any other developments will be to investigate the feasibility of integrating CEEDS with other systems. These systems could include:

- similar systems for other disciplines (eg. Mechanical & Civil),
- systems that provide services to CEEDS (eg. Project Management Packages (CUE), Documentation management (DIS and KKS Documentation systems), etc.) and
- systems to which CEEDS provides a service (eg. Plant Maintenance Systems (PERMAC), Engineering Simulators (PSSU/E), etc.),
- other CAD systems such as Geographic Information Systems (GIS) used by transmission and distribution.

Currently CEEDS is dependent on data fed in from CUE. The DIS and Majuba documentation systems can be made available to CEEDS users. CEEDS will export design data into PERMAC for plant maintenance purposes.

Some stand alone PC databases have been absorbed into CEEDS and are made available to all CEEDS users as a centralized function. Some include the Official Structure Code Dictionary, the Official Abbreviations dictionary and a list of contrast areas. Some of these "Dictionaries" will need to be separated from CEEDS and made available on a centralised basis for use by other applications. These dictionaries include Structure Codes, Abbreviations, the KKS Key part dictionary and later on, the BFS Component Listing. These will need to be administered by section(s) representing all users of these centralised dictionaries. For example, a centralised KKS Key part dictionary will be used by CEEDS, PERMAC and the Majuba documentation system, amongst others, and should be administered by Systems Engineering.

The PSSU/E interface is to be investigated shortly. If requirements arise for CEEDS facilities to be used to manage the design of an electrical system spread over a large geographical area (eg. township reticulation), then interaction with a GIS will be investigated.

- Use of International Coding Standards:

The CEEDS physical model has specifically been designed to be KKS code independent. Code verification facilities will be provided to carry out automatic verification in batch mode to provide an indication of possible errors in coding. Separate verification facilities can be provided for whatever coding system is being used on a given project. (This was to provide CEEDS with the capability of carrying out Lifex type work on old stations not using the KKS Coding System. It also makes it possible to use meaningful preliminary codes (non-KKS) during initial design.)

CEEDS is fully compatible with the KKS Coding system being used on Majuba.

If the BFS system is adopted within ESKOM, in-house coding will need to be translated to comply with international standards. If this comes about, there will be an even stronger link between PERMAC and CEEDS as a plant item can be identified on CEEDS as a PERMAC stock item during design, and maintenance planning can actually take place during early stages of design (and indeed, could influence the design by taking maintainability into account).

#### Development Resources:

All of the above ideas are not worth the paper they are written on if we are not able to execute them due to lack of adequate resources. From a development point of view, the most important resources are people and facilities.

#### - People:

- One of the major problems facing ESKOM during the development of reasonably sophisticated software systems is the shortage of people with adequate skills. Unfortunately this is a world-wide shortage and no easy solution exists (especially not in South Africa). It has been managed on CEEDS 2 because the level of development has been fairly fundamental. There are a couple of reasonably sophisticated facilities but nothing approaching what is still envisaged.

The software systems that would be used on CEEDS 3 are very sophisticated packages and this fact must not be overlooked. Quite complex facilities would need to be written to provide the graphical functionality demanded by a system such as CEEDS. At this stage, it is felt that further development would be hampered without intensive training of all people who would be intimately involved with the graphical development. This would most probably involve attending

- o training course offered by the software vendor. The more sophisticated the systems being developed, the more prominent this problem is going to become. There is therefore the need for continuous on-going training and education. All team members are encouraged to participate in further study schemes in order to equip themselves with the necessary background to carry out their jobs. If very skilled individuals are identified, these people should be looked after and encouraged to remain in the technical field.

- One avenue that could give us access to high quality research and development personnel is by making use of our universities. (A recent project proposal by the University of the Witwatersrand, involving the development of a system for urban reticulation, is of great interest to us. It could be of great mutual benefit for ESKOM to become involved in such a project).
- The Information Technology (IT) department will always be involved in any further development and the Engineering Department will always be somewhat dependent on the quality of service provided. This is a good thing, as the IT department is quite a good source of the high quality programmers and computer scientists that would be needed for consultation and development of sophisticated systems. These services come at a cost though.
- Other departments involved in similar work can also be tapped for expertise and advice. Transmission Engineering is such a department.
- Another major resource constraint is the availability and time required of key line personnel who would be used to provide the necessary technical and procedural user requirements for the development team to design against. There is no easy solution for this problem because key personnel are needed, as they are usually the people who know what is required and at the same time they are usually loaded the heaviest as a result of their being able to get the job done.
- General user involvement is also required to facilitate training and the acceptance of changes that may come about with the introduction of a new technology.
- Hardware and Software facilities:

In order to gain potential benefits from some of the ideas mentioned above, a rich software environment is required. It will have to be realised that a system is limited by the software and hardware technology within which it is developed and

implemented. For example, the mainframe has very limited graphical capabilities and if graphics are a requirement alternative platforms will need to be obtained. On the other hand, the purchasing of facilities is very expensive and the purchasing of any equipment and software must be cost justified against potential benefits (tangible and in-tangible).

Software is often application specific, but if software can be limited to standard products (as approved by the ESKOM IT Department), all the benefits of doing so will be available. Benefits such as bulk order discounts, cheaper version upgrades, hot-line support, in-house expertise etc.

With Hardware, it is even more important to stick to ESKOM standards. The IT department has a whole infrastructure supporting a vast amount of equipment. Facilities such as quick turn around maintenance and fault control are available at much cheaper rates than are usually available from the supplier. Another benefit is that equipment can be thoroughly evaluated by teams of knowledgeable people before we even consider purchasing such equipment.

Other factors are things like network compatibility. The IT department will not just allow any random equipment to be connected to any current or future networks. IT also has the task of moving with technology and they have the resources to do the job.

A point that must be taken into account with hardware is that Engineering cannot go it alone unless they are willing to pay for that privilege. A support infrastructure is required.

#### Discussion of Strengths and Weaknesses:

- The Design Process Automation (DPA) Team has been involved in the development of CEEDS for some time now and hopefully will continue providing a systems development service into the future. One aspect that has been overlooked in the past and requires attention for any future work is the need to cost justify activities. This is currently a weakness in the team as no-one has experience in this area. It is also a difficult metric to quantify as many benefits are intangible and other benefits are the provision of facilities to prevent mistakes made on previous projects, or to provide better control over invoicing, for example.
- CEEDS is a large system and covers many activities within the

department. It is also an expanding system. A time will be reached when the development team will be unable to carry the training, implementation and administration loads as well as to continue with further development. Help in implementing the systems, and in training future users will be required from the users. Users/owners of the system will need to become very much more involved if CEEDS is to reach its full potential.

- Organizational planning and need for inter-departmental communication regarding the development of systems similar to CEEDS will be required. This is to prevent duplication, enhance interconnectivity and prevent the black-hole phenomenon (empire building)).

The current CEEDS system is not easily marketable. Firstly because it has been designed to meet the needs of Eskom in building Majuba. Massive customisation would be required to suite the product for other environments. Secondly, the mainframe platform it has been developed on puts it out of the league of many potential smaller customers.

CEEDS does have marketing potential though;

- The current system could be used in a bureau fashion with only minor additions.
- A sub-system of CEEDS could be implemented on a PC for small projects.
- The full, or a sub-system could be implemented on a stand-alone workstation system,
- know-how and experience gained during the development of CEEDS which is applicable to other projects can now be made available (at a cost).

If CEEDS 3 is developed, it would be far more marketable as it would run on cheaper platforms, would be far more flexible and would fit in line with world trends in graphical interfaces and software technology.

- If Eskom Engineering competes in the open market against other engineering contractors, CEEDS could provide a vital competitive edge.
- CEEDS principles could be applied to other applications such as urban reticulation, transmission, building projects (in Eskom), other plants such as Sasol, Yacor, Mosgas, etc. CEEDS could also be used in fields other than power station construction such as ships, vehicles, etc.



---

PS: (6 March 1991)

The CEEDS 2 system is in production and many of the activities described under CEEDS 2 development above have already been completed.

An Interpro 6000 workstation has been purchased for the purpose of developing CEEDS 3. Work has begun in this field.

A project to investigate the creation of an Integrated Engineering Information System (IEIS) spanning all engineering disciplines and plant operation and maintenance has been initiated.

## R. References and further reading material

References and other reading matter are grouped into four sections namely:

- Non-Technical Issues
- CAE and Modeling
- Computer Environment
- Design Methodologies and Software Engineering

Articles and books referred to in the main report are marked with an asterisk (\*).

### R1. Non-Technical Issues

#### MGT 01 \*

Malchrzak, Ann and Salzman Harold.

"Introduction to the Special Issue: Social and Organizational dimensions of Computer-Aided Design".

*IEEE Transactions on Engineering Management*, Vol. 36, Number 3, Aug. 1989, pp. 174-179.

#### MGT 02 \*

Liker, Jeffray K. and Fiescher, Mitchell.

"Implementing Computer-Aided Design: The Transition of Nonusers".

*IEEE Transactions on Engineering Management*, Vol. 36, Number 3, Aug. 1989, pp. 180-190.

#### MGT 03 \*

Forslin, Jan; Thulestedt, Britt-Marie and Andersson, Sven.

"Computer-Aided Design: A Case of Strategy in Implementing a New Technology".

*IEEE Transactions on Engineering Management*, Vol. 36, Number 3, Aug. 1989, pp. 191-201.

#### MGT 04

Adler, Paul S.

"CAD/CAM: Managerial Challenges and Research Issues".

*IEEE Transactions on Engineering Management*, Vol. 36, Number 3, Aug. 1989, pp. 202-215.

**MGT 05**

Badham, Richard.

"Computer-Aided Design, Work Organization, and the Integrated Factory".  
*IEEE Transactions on Engineering Management*, Vol. 36, Number 3, Aug. 1989,  
pp. 216-226.

**MGT 06**

Lee, Gloria L.

"Managing Change with CAD and CAD/CAM".  
*IEEE Transactions on Engineering Management*, Vol. 36, Number 3, Aug. 1989,  
pp. 227-233.

**MGT 07**

Kocaoglu, Dunder F. (Editor-in-Chief).

"Editorial: Recent Developments in Engineering Management Research".  
*IEEE Transactions on Engineering Management*, Vol. 36, Number 4, Nov. 1989,  
pp. 249-251.

**MGT 08 \***

Salzman, Harold.

"Computer-Aided Design: Limitations in Automatic Design and Drafting".  
*IEEE Transactions on Engineering Management*, Vol. 36, Number 4, Nov. 1989,  
pp. 252-261.

**MGT 09**

Sinclair, M.A.; Siemieniuch, C.E. and John, P.A.

"A User-Centered Approach to Define High-Level Requirements for Next  
-Generation CAD Systems for Mechanical Engineering".  
*IEEE Transactions on Engineering Management*, Vol. 36, Number 4, Nov. 1989,  
pp. 262-270.

**MGT 10 \***

Brooks, Laurence S. and Wells, Christopher S.

"Role Conflict in Design Supervision".  
*IEEE Transactions on Engineering Management*, Vol. 36, Number 4, Nov. 1989,  
pp. 271-281.

**MGT 11**

Mansko, Fred and Wolf, Harald.

"Design Work in Change: Social Conditions and Results of CAD Use in  
Mechanical Engineering".  
*IEEE Transactions on Engineering Management*, Vol. 36, Number 4, Nov. 1989,  
pp. 282-292.

**MGT 12**

Plonski, Guilherme Ary.

**"The Need to Reconceptualize CAD: The Case of Brazilian Engineering Consultancy Firms".**

*IEEE Transactions on Engineering Management*, Vol. 36, Number 4, Nov. 1989, pp. 293-297.

**MGT 13 \***

Simmonds, Paul and Senker, Peter.

**"Managing CAD in heavy electrical engineering".**

*Computer-Aided Engineering Journal*, Oct. 1989, pp. 181-184.

**MGT 15**

Ahituv, Niv and Neumann, Seev.

**"Principles of Information Systems for Management".**

Wm. C. Brown Publishers, United States, 1985.

**MGT 16 \***

Eason, K.D.

**"User Centred Design for Information Technology Systems".**

*Phys. Technol.*, Vol. 14, 1983, pp. 219-224.

**MGT 17**

Mainiero, Lisa A. and DeMichieli, Robert L.

**"Minimizing Employee Resistance to Technological Change".**

*Personnel*, Vol. 63, Number 7, July 1986, pp. 32-37.

**MGT 18**

Sullivan, Cornelius H., Jr. and Smart, John R.

**"Planning for Information Networks".**

*Sloan Management Review*, Vol. 28, Part 2, 1987, pp. 33-44.

**MGT 19**

Lucas, Henry C., Jr.

**"Utilizing Information Technology: Guidelines for Managers".**

*Sloan Management Review*, Vol. 28, Part 1, 1986.

**MGT 20**

Smith, Jill.

**"Beyond user friendly - towards the assimilation of multifunctional-workstation capabilities".**

*Behaviour and Information Technology*, Vol. 3, Number 3, 1984, pp. 205-220.

**MGT 21**

Buchanan, D.A. and Boddy, D.  
**"Organisations in the Computer Age".**  
 Gower, 1983.

**MGT 22 \***

Ellis, Peter.  
**"Office planning and design: the impact of organisational change due to advanced information technology".**  
*Behaviour and Information Technology*, Vol. 3, Number 3, 1984, pp. 221-233.

**MGT 23**

Long, J.; Hammond, N.; Barnard, P.; Morton, J. and Clark, I.  
**"Introducing the interactive computer at work: The users' views".**  
*Behaviour and Information Technology*, Vol. 2, Number 1, 1983, pp. 39-106.

**MGT 24**

Salvendy, Gavriel.  
**"Review and reappraisal of human aspects in planning robotic systems".**  
*Behaviour and Information Technology*, Vol. 2, Number 3, 1983, pp. 263-287.

**MGT**

Parker, Edwin B.  
**"Social implications of computer/telecoms systems".**  
*Telecommunication Policy*, Vol. 1, No. 1, 1976, pp. 3-20.

**MGT 26**

Rosenbaum, Bernard L.  
**"Critical Skills for Successful Technical Professionals".**  
*Personnel*, Vol. 63, No. 10, Oct 1986, pp. 56-60.

**MGT 27** Finkelstein, James A. and Hatch, Christopher H.

**"Job Evaluation: New Technology, New Role for HR Managers".**  
*Personnel*, Vol. 64, No. 1, Jan 1987, pp. 5-10.

**MGT 28**

Hoos, Ida Russakoff.  
**"When the Computer Takes Over the Office".**  
*Harvard Business Review*, Vol. 38, No. 4, 1960, pp. 102-112.

**MGT 29**

Heilbroner, Robert L.  
**"Automation and Technical Change",**  
 Prentice Hall, Englewood Cliffs, New Jersey, 1962.

## R2. CAE and Modelling

### MDL 02 \*

Kelkar, S.; Moussa, W.; Wunderlich, R. and Hitchcock, L.

**"Computer-Aided Electrical Design of Power Processing Systems".**

*APEC '88: Third Annual IEEE Applied Power Electronics Conference and Exposition. Conference Proceedings 1988*, pp. 72-81.

### MDL 03

Kanga, Darius.

**"Application of 'Intelligent' Computer-Aided Design Techniques to Power Plant Design and Operation".**

*IEEE Transactions on Energy Conversion*, Vol. EC-2, No. 4, Dec 1987, pp. 592-597.

### MDL 04B

Erthyimladis, A. E.

**"Interactive Power System Analysis".**

*IEE Colloquium on Computer Aided Engineering in Design, Digest No. 41, April 1983*, pp. 2/1-2/7.

### MDL 04C

Cockshoot, D. W.

**"Computer Aided Electrical Design".**

*IEE Colloquium on Computer Aided Engineering in Design, Digest No. 41, April 1983*, pp. 3/1-3/21.

### MDL 05

Bunce, K. M.

**"Computers and the electrical contractor"**

*Electrotechnology*, Jul 1985, pp.101-103.

### MDL 06

Editor

**"Computer programs for the electrical contractor".**

*Electrotechnology*, Jul 1988, pp.81-87.

### MDL 07

Phillips, Barry W.

**"Framework Frontier Opens at DAC"**

(Design Automation Conference). *ESD: THE Electronic System Design Magazine*, Jun 1989, pp. 55-58.

## MDL 08

Puttgen, Hans B. and Jansen, John F.

"An Expert System for the Design of a Power Plant Electrical Auxiliary System".  
*IEEE Transactions on Power Systems*, Vol. 3, No. 1, Feb 1988, pp. 254-261.

## MDL 10

Editor

"INGECAD: Extensive Application to Nuclear Projects of CAE".  
*Framatome Newsletter*, No. 31. pp. 6-9.

## MDL 11

Editor

"Cradle to grave computer-aided engineering for process plant builders".  
*Achievement*, May 1989.

## MDL 12

Konstantinow, George.

"Emerging Standards for Design Management Systems".  
*Proceedings of the IEEE Computer Standards Conference 1988 - Computer Standards Evolution: Impact and Imperatives*, pp. 16-21.

## MDL 13 Hashemi, Nasrollah; Love, Daniel J. and Tajaddini, Fred Y.

"A Relational Database Approach to Power Plant Design and Operating Plant Services".

*IEEE Power Engineering Reviews*, September 1988, pp. 31-32.

## MDL 14 \*

Ohsuga, Setsuo.

"Towards intelligent CAD systems".

*Computer-Aided Design*, Vol. 21, No. 5, June 1989, pp. 315-337.

## MDL 15

Lang-Lendorff, G. and Untorburg, J.

"Changes in understanding of CAD/CAM: a database-oriented approach".  
*Computer-Aided Design*, Vol. 21, No. 5, June 1989, pp. 309-314.

## MDL 16 \*

Roy, U. and Liu, C. R.

"Establishment of functional relationships between product components in assembly database".

*Computer-Aided Design*, Vol. 20, No. 10, Dec 1988, pp. 570-580.

**MDL 17 \***

Shou, Phillip C-Y.

"VLSI design with object-oriented knowledge bases".

*Computer-Aided Design*, Vol. 20, No. 5, June 1988, pp. 272-280.

**MDL 18 \***

Durr, M.; Huck, M.; Kemper, A.; Mohrholz, P. and Wallrath, M.

"Using conventional and nested relational database systems for modelling CIM data".

*Computer-Aided Design*, Vol. 21, No. 6, July 1989, pp. 379-392.

**MDL 19**

Zhang, Z. Z.; Hope, G. S. and Malik, O. P.

"Expert Systems in Electrical Power Systems - A Bibliographical Survey".

*IEEE Transactions on Power Systems*, Vol. 4, No. 4, Oct 1989, pp. 1355-1361.

**MDL 25 \***

West, Steve.

"CAE: changing the face of business and electronic product development".

*Electronics News*, 26 April 1990, pp. 18 - 19.

**MDL 26 \***

Nicholson, T. A. J.

"Finding the shortest route between two points in a network".

*Computer Journal*, Vol. 9, No. 3, 1966, pp. 275-280.

**MDL 27**

Beiter, Stephen E.

"Computer-Aided Routing of Printed Circuit Boards".

*Byte*, June 1987, pp. 190-208.

**MDL 31 \***

Ashfield, Alan.

"Cable-izing computer programs".

*Wiring-Installations-Supplies*, Issue 22, 1984, pp. 21-23.

**MDL 32 \***

Gallagher, Sean.

"Network management in electrical distribution".

*Computer-Aided Engineering Journal*, Dec. 1989, pp. 202-206.



## MDL 33 \*

Vanpelt, H. E. and Ashe, K. L.  
"Radiation exposure reduced with computer-aided maintenance".  
*Power Engineering*, April 1989, pp. 40-42.

## MDL 34 \*

Chan, Sherman.  
"Interactive Graphics Interface for Power Systems Network Analysis".  
*IEEE Computer Applications in Power*, 1990, pp. 34-38.

## MDL 36

Huhns, Michael N. and Acosta, Ramon D.  
"ARGO: A system for Design by Analogy".  
*IEEE Expert*, Fall 1988, pp. 53-68.

## MDL 37 \*

Beaudouin-Lafon, Michel and Karsenty, Solange.  
"Prototyping User Interfaces for Applications Depicted by Graphs".  
*IEEE 1988, TH0212-1/88/0000/0436\$01.00*, pp. 436-445.

## MDL 38

Alvarado, F. L.; Lasseter, R. H. and Liu, Y.  
"An Integrated Engineering Simulation Environment".  
*IEEE Transactions on Power Systems*, Vol. 3, No. 1, Feb. 1988, pp. 245-253.

## MDL 39 \*

Harris, George; Jacobs, Lori J. and Devito, Denise M.  
"Automatic Generation of Tables and Rawports for Electrical Schematics".  
*1990 International IGUG*, Monograph No. 90-031.

## MDL 40 \*

Ramanathan, Jay and Hartung, Roland L.  
"A Generic Iconic Tool for Viewing Databases".  
*IEEE Software*, 1989, pp. 50-57.

## MDL 41 \*

Larson, Orland.  
"Strategic Importance of Relational Databases".  
*Proceedings 1987 North American Conference of Hewlett-Packard Business Computer Users*, Las Vegas, Nevada, Sept. 1987.

MDL 42 \*

Codd, E. F.

"A Relational Model of Data for Large Shared Data Banks",  
*CACM*, Vol. 13, No. 6, June 1970, pp. 377-387.

MDL 43/9 \*

Malda, Anthony S. and Shapiro, Stuart C.

"Intensional Concepts in Propositional Semantic Networks",  
*Cognitive Science*, 6(4), 1982, pp. 291-330.

MDL 43/10 \*

Brachman, Ronald J.

"On the Epistemological Status of Semantic Networks",  
*Associative Networks: Representation and Use of Knowledge by Computers*, edited  
by N. V. Findler, New York: Academic Press, 1979, pp. 3-50.

MDL 43/11 \*

Woods, William A.

"What's in a Link: Foundations for Semantic Networks",  
*Representation and understanding: Studies in Cognitive Science*, edited by D.G.  
Bobrow and A.M. Collins, New York: Academic Press, 1975, pp. 35-82.

MDL 43/12 \*

Minsky, Marvin.

"A Framework for Representing Knowledge",  
*Mind Design*, edited by J. Haugeland, Cambridge, MA: The MIT Press, 1981,  
pp. 95-128.

MDL 43/13 \*

Bobrow, Daniel G. and Winograd, Terry.

"An Overview of KRL, a Knowledge Representation Language",  
*Cognitive Science*, 1(1), 1977, pp. 3-46.

MDL 43/14 \*

Hayes, Patrick J.

"The logic of Frames",  
*Frame Concepts and Text Understanding*, edited by D. Metzger, Berlin: Walter de  
Gruyter and Co., 1979, pp. 46-61.

**MDL 44 \***

Greenstein, Irwin.  
**"Is AI/DBMS Misd Big Catalyst for Software Shift?",**  
*MIS Week*, Vol. 8, No. 16, April 1987, pp. 1,40,43.

**MDL 45 \***

Aleksander, I. and Morton, H.  
**"Artificial Intelligence: an engineering perspective",**  
*IEE Proceedings*, Vol. 134, Pt. D, No. 4, July 1987, pp. 218-223.

**MDL 46 \***

MacLeod, I. and Lun, V.  
**"Expert systems in engineering",**  
*Electron*, Nov/Dec 1986, p. 4-6.

**MDL 47 \***

Roman, Harry T. and Marian, Frank A.  
**"The Year 2000 Power Plant",**  
*IEEE Computer Applications in Power*, April 1989, pp. 19-22.

**MDL 48**

Kirschen Daniel S.; et al.  
**"Controlling Power Systems During Emergencies: The Role of Expert Systems",**  
*IEEE Computer Applications in Power*, April 1989, pp. 41-45.

**MDL 49**

Smith, Douglas J.  
**"Intelligent computer systems enhance power plant operations",**  
*Power Engineering*, Dec 1989, pp. 21-26.

### **R3. Computer Environments**

**SC 01**

Bramer, Brian.

"Problems of software portability with particular reference to engineering CAE/CAD systems",

*Computer-Aided Engineering Journal*, Dec 1988, pp. 233-240.

**SC 02**

Brown, Alan W.

"The relationship between CAD and software development support - a database perspective",

*Computer-Aided Engineering Journal*, Dec 1988, pp. 226-232.

**SC 03 \***

Dahlgren, Kent.

"Putting Graphical Interfaces into PERSPECTIVE",

*Dr. Dobbs's Journal*, Nov. 1988, pp. 32-41.

**SC 04 \***

Flynn, Brian.

"A Word on Windows",

*ACCESS*, May/June 1989, pp. 29-31.

**SC 05 \***

Greco, Frank D.

"Windowing Systems Overview",

*Programmer's Journal*, Vol. 6, No. 4, 1988, pp. 18-20.

**SC 06 \***

Jones, Sara.

"Graphical interfaces for knowledge engineering: an overview of relevant literature",

*The Knowledge Engineering Review*, pp. 221-247.

**SC 07 \***

Jordan, Mike and Reach, Beverly.

"Navigating Information resources with DD Access",

*AEC Review*, Vol. 1, Third Quarter 1989, pp. 4-5.

SC 08 \*

McPertlin, John P.  
"A Graphical Look at Interfaces",  
*Information WEEK*, Oct. 1989, pp. 36-42.

SC 09 \*

Pretzold, Charles.  
"Introducing the OS/2 Presentation Manager",  
*PC Magazine*, July 1988, pp.379-394.

SC 10

Rui, A.; Weston, R. H.; Gascolgne, J. D.; Hodgson, A. and C. M. Sumpter.  
"Automating Information transfer in manufacturing systems",  
*Computer-Aided Engineering Journal*, June 1988, pp. 113-121.

SC 11 \*

Steven Baker, M.  
"X Windows - A View from DOS",  
*Programmer's Journal*, Vol. 6, No. 4, 1988, pp. 32-39.

SC 12

Wilson, Andrew C.  
"A Picture's worth a thousand lines of code",  
*ESD: The Electronic System Design Magazine*, July 1989, pp. 57-60.

SC 13 \*

Ballinger, J.  
"IBM's Common User Access (CUA) Standard and its Impact on User Interface Design",  
*SAIEE Proceedings of the Symposium: Software Issues in User Interface Design*,  
January 1991, pp. 15-22.

SC 14 \*

Milton, Deon and Van Zijl, Lynette.  
"The Development of a Large Graphical Interface using X and the X Toolkit",  
*SAIEE Proceedings of the Symposium: Software Issues in User Interface Design*,  
January 1991, pp. 23-32.

**SC 15 \***

**Microstation Manuals**

**Intergraph Corporation, One Madison Industrial Park, Huntsville, Alabama 35807-4201**

**SC 16 \***

**ORACLE Manuals**

**Oracle Corporation, Belmont, California, USA.**

#### **R4. Design Methodologies and Software Engineering**

##### **Software Engineering:**

**SM 01**

Akima, Noboru and Ooi, Fusatake.

"Industrializing Software Development: A Japanese Approach",  
*IEEE Software*, March 1989, pp. 13-21.

**SM 02**

Allman, Eric and Stonebraker, Michael.

"Observations on the Evolution of a Software System",  
*Computer*, June 1982, pp. 27-31.

**SM 03**

Ampt, Gilles J.

"On the Integration of software design Information",  
*Information and Software Technology*, Vol. 29, No. 6, July 1987, pp. 321-329.

**SM 04 \***

Ashworth, Caroline M.

"Structured system analysis and design method (SSADM)",  
*Information and Software Technology*, Vol. 30, No. 3, April 1988, pp. 153-163.

**SM 05**

Belanger, David G.; David Bergland, G. and Wish, Mike.

"Some Research Directions for Large-Scale Software Development",  
*AT&T Technical Journal*, July/Aug 1988, pp. 77-92.

**SM 06 \***

Boehm, Barry W.

"Software Engineering".

*IEEE Transactions on Computers*, Vol. C-25, No. 12, Dec 1976, pp. 1226 - 1241.

**SM 07 \***

Brooks, F. G.

"The Mythical Man Month",  
Addison Wesley, 1975.

SM 08 \*

Daly, B. Edmund.

"Management of Software Development".

*IEEE Transactions on Software Engineering*, May 1977, pp. 229 - 242.

SM 09 \*

Emberton, John and Mann, Robb.

"Methodology for effective information system planning",

*Information and Software Technology*, Vol. 30, No. 4, May 1988, pp. 244-249.

SM 10 \*

Factor, Robert M. and Smith, William B.

"A Discipline for Improving Software Productivity",

*AT&T Technical Journal*, July/Aug 1988, pp. 2-9.

SM 11 \*

Hall, P.A.V. and Galai, G. H.

"Computer-aided software engineering",

*Computer-Aided Engineering Journal*, Aug. 1989, pp. 113-120.

SM 12

Marsden, James R. and Pingry, David E.

"End User - IS Design Professional Interaction - Information Exchange for Firm Profit or End User Satisfaction?",

*North-Holland Information & Management*, Vol. 14, 1988, pp. 75-80.

SM 13 \*

Morrison, W.

"Communicating with users during systems development",

*Information and Software Technology*, Vol. 30, No. 5, June 1988, pp. 295-298.

SM 14

Rahman, Wasifur.

"Software quality by management",

*Information and Software Technology*, Vol. 29, No. 9, Nov 1987, pp. 511-516.

SM 15 \*

Robinson, Ken.

"Cost justification for computer systems",

*Computer Mail*, Supplement to Financial Mail, February 24, 1989, pp. 54-55.



SM 16 \*

Shoval, Peretz.

**"ADISSA: Architectural Design of Information Systems based on Structured Analysis",***Information Systems*, Vol. 13, No. 2, 1988, pp. 193-210.

SM 17 \*

Slusky, Ludwig.

**"Integrating software modelling and prototyping tools",***Information and Software Technology*, Vol. 29, No. 7, Sept. 1987, pp. 379-387.

SM 18 \*

Sumner, Mary and Benson, Robert.

**"The Impact of Fourth Generation Languages on Systems Development",***North-Holland Information & Management*, Vol. 14, 1988, pp. 81-92.

SM 19 \*

Veryard, Richard.

**"Implementing a methodology",***Information and Software Technology*, Vol. 29, No. 9, Nov 1987, pp. 469-474.

SM 20 \*

Martin, J.

**"Design of Man-Computer Dialogues",***International Journal of Man-Machine Studies*, Vol. 3, 1971.

SM 21

Barnard, P. J; Morton, J; Long, J. B. and Otterly, E. A.

**"Planning Menus for Display: Some Effects of their Structures on User Performance",***IEE Conference Publication*, No. 150, April 1977, pp. 130-133.

SM 22 \*

Jones, P. F.

**"Four Principles of Man-Machine Dialogue",***Computer Aided Design*, Vol. 10, No. 3, Nov 1978, pp. 197-202.

SM 23 \*

Edmonds, E.

**"The Man-Machine Interface: A Note on Concepts and Design",***International Journal of Man-Machine Studies*, Vol. 3, 1982, pp. 231-233.

SM 24 \*

Guest, S. P.

"The Use of Software Tools for Dialogue Design",

*International Journal of Man-Machine Studies*, Vol. 3, 1982, pp. 263-286.

SM 25 \*

IEW Manuals

Knowledgware, 3340 Peachtree Rd., N.E., Suite 1100, Atlanta, GA30026

SM 26 "

SSADM Manuals

National Computing Centre (NCC) Limited, Oxford Road, Manchester, M13 2ED, England.

SM27 \*

Chang, Carl K and Trubow, George B.

"Joint Software Research Between Industry and Academia",

*IEEE Software*, November 1990, pp. 71-77.

**Configuration Management:**

CM 01

Ambriola, Vincenzo and Bondix, Lars.

"Object-Oriented Configuration Control",

*ACM SIGSOFT Software Engineering Notes*, Vol. 14, No. 7, Nov. 1989, pp. 133-136.

CM 02

Clemm, Geoffrey M.

"Replacing Version-Control with Job-Control",

*ACM SIGSOFT Software Engineering Notes*, Vol. 14, No. 7, Nov. 1989, pp. 162-169.

CM 03

Falkenberg, Bernhard.

"Configuration Management for a Large (SW) Development",

*ACM SIGSOFT Software Engineering Notes*, Vol. 14, No. 7, Nov. 1989, pp. 34-37.

CM 04

Gustavsson, Anders.

"Maintaining the Evolution of software objects in an integrated environment",

*ACM SIGSOFT Software Engineering Notes*, Vol. 14, No. 7, Nov. 1989, pp. 114-117.

CM 05

Lacroix, M. and Laverney, P.  
 "The change request process",  
*ACM SIGSOFT Software Engineering Notes*, Vol. 14, No. 7, Nov. 1989, pp. 122-125.

CM 06

Miller, Terrence C.  
 "A Schema for Configuration Management",  
*ACM SIGSOFT Software Engineering Notes*, Vol. 14, No. 7, Nov. 1989, pp. 26-29.

CM 07

Renuart, Robert F. and Venkatraman, Raju.  
 "Electronic document management improves design change process",  
*Power Engineering*, April 1989, pp. 48-50.

CM 08

Simmonds, I.  
 "Configuration management in the PACT software engineering environment",  
*ACM SIGSOFT Software Engineering Notes*, Vol. 14, No. 7, Nov. 1989, pp. 118-121.

CM 09

Westfechtel, B.  
 "Revision control in an integrated software development environment",  
*ACM SIGSOFT Software Engineering Notes*, Vol. 14, No. 7, Nov. 1989, pp. 96-105.

## GLOSSARY OF ABBREVIATIONS

AI	Artificial Intelligence
ASCII	American Standard Coding Information Interchange
BASIC	3GL Computer Language
BFS	Component Identification Coding System
C	3GL Computer Language
C&I	Control and Instrumentation
CAD	Computer-Aided Drafting
CAE	Computer-Aided Engineering
CASE	Computer-Aided Software Engineering
CBT	Computer Based Training
CCC	Code Configuration Control
CEEDS	Centralised Electrical Engineering Database Scheme
CPU	Central Processing Unit
CRED	Cable Racking and Equipment Digitising
CUA	Common User Access
CUE	Scheduling Package
D.O.	Drawing Office
DC	Direct Current
DIS	Drawing Information System
DPA	Design Process Automation
DPM	Design Process Model
EB/DIC	Extended Binary-Coded Decimal Interchange Code
EDM	ESKOM Development Methodology
ESKOM	Electricity Utility in South Africa
FM	Financial Model
FORTRAN	3GL Computer Language
GIS	Geographical Information System
GUI	Graphical User Interface
IBM	International Business Machines
IC	Integrated Circuit
ID	Identification
IEEE	Institute of Electrical and Electronic Engineers
IEIS	Integrated Engineering Information System
IEW	Information Engineering Workbench
IPF2	Indexed sequential database system
ISO	International Standards Organisation
IT	Information Technology
IVI	Interactive Video Instruction
JAD	Joint Application Development
Kendal	ESKOM Power Station
KKS	Unique Identification Coding System
LAN	Local Area Network
LIFEX	Life Extension
LV	Low Voltage
Majuba	ESKOM Power Station
MMI	Man Machine Interface
MV	Medium Voltage
NFS	Network Filing System
OO	Object Oriented
OOP	Object Oriented Programming
ORACLE	Relational Database Product

OS	Operating System
OS/2	IMB PC Operating System
OSF	Open Software Foundation
PC	Personal Computer
PCB	Printed Circuit Board
PERMAC	Plant Maintenance Package
PLC	Programmable Logic Controller
PM	Physical Model
PMMF	Physical Model Manipulation Function
PSEED	Power Station Electrical Engineering Department
PSSU/E	Power System Simulation package
QMX	Query Management Executive (ORACLE Product)
SSADM	Structured System Analysis and Design Methodology
SQL	Structured Query Language
UNIX	AT&T Operating System
VM	Virtual Machine (Operating System)
WAN	Wide Area Network
WBS	Work Breakdown Structure
XEDIT	Text editor product
2D	Two Dimensional
3D	Three Dimensional
3GL	Third Generation Language



**Author: Blake Anthony V.**

**Name of thesis: Investigation Into The Requirements For An Integrated Computer-aided Engineering Environment.**

***PUBLISHER:***

**University of the Witwatersrand, Johannesburg**

**©2015**

***LEGALNOTICES:***

**Copyright Notice:** All materials on the University of the Witwatersrand, Johannesburg Library website are protected by South African copyright law and may not be distributed, transmitted, displayed or otherwise published in any format, without the prior written permission of the copyright owner.

**Disclaimer and Terms of Use:** Provided that you maintain all copyright and other notices contained therein, you may download material (one machine readable copy and one print copy per page) for your personal and/or educational non-commercial use only.

The University of the Witwatersrand, Johannesburg, is not responsible for any errors or omissions and excludes any and all liability for any errors in or omissions from the information on the Library website.