NOTTINGHAM TRENT UNIVERSITY

Department of Industrial and Manufacturing Engineering

The Design and Development of an Engineering Data Management System

By : R. A. Booth

A thesis submitted to the Nottingham Trent University for the degree

of

MASTER OF PHILOSOPHY

September 1994

Acknowledgements

I hereby formally acknowledge Mr. J. Drysdale for his assistance during the research and writing of this report.

Abstract

The objectives of this project are to determine the benefits of Engineering Data Management and to develop such a system for Brush Transformers Ltd. Engineering Data Management concepts are presented which cover planning, cost/benefit analysis and implementation.

Detailed within the report is the design of the Engineering Data Management system developed for Brush Transformers Ltd and its subsequent implementation.

Finally, a number of conclusions and recommendations are discussed as a result of the research and its practical application within industry.

Contents

1.0 Introduction

Only in recent years have organisations been addressing the question of how to make use of IT to manage their engineering information systems although some have been doing this to support their physical production process for over twenty years. To survive in the 1990s most companies will have to make use of Engineering Data Management (EDM) Systems to ensure that they have more timely and accurate information and to reduce their time to market. Right first time and not recycling incorrect data is the objective for EDM. Major benefits have already been achieved using these systems: one company saved £1 million per year and 4 million pieces of paper.

The objectives of this project were to determine the benefits of Engineering Data Management Systems and to develop such a system for Brush Transformers Ltd.

Brush Transformers Ltd designs and manufactures distribution, power, cast resin and flame proof transformers. The process of designing transformers involves a high level of engineering and drawing which must be managed in order to maximise the use of data. Within most design led manufacturers it is essential that previous designs and experience are used as a foundation for growth and profitability within the market place. However, after investing in Computer Aided Design equipment the company had received very little pay back. It was considered that the reason for this was little standardisation and high levels of duplicated effort.

The scope of this project includes the methods used for system analysis and design and details a functional specification for the EDM system followed by its implementation.

2.0 Engineering Data Management

2.1 What is Engineering Data Management

Research into the definition showed that no truly accepted meaning exists due to the many different business functions and processes which may be supported by EDM. As illustrated in Figure 1 these systems can eventually provide support to many different business processes in an organisation from tendering to distribution and customer services. This support covers both handling of information and management of the processes. When fully implemented they provide a library enquiry and updating technical information services for all items of engineering/technical data to both the engineering information produces and users. Some of the data can be held within the EDM system, the rest being held in paper form or on other computer application files. The system will also control access to or authorise the issue of documents to named individuals or locations. Documents can either be computer or paper based and will usually be a mixture. The system enables the data to be viewed by or copied to any part of the company which has the appropriate equipment. They are far more than a link between CAD and MRP systems though this can be an important element. They also assist the management of all the information producers who generate engineering information. The progress of a document for a new part or a change can be planned and monitored throughout its life.

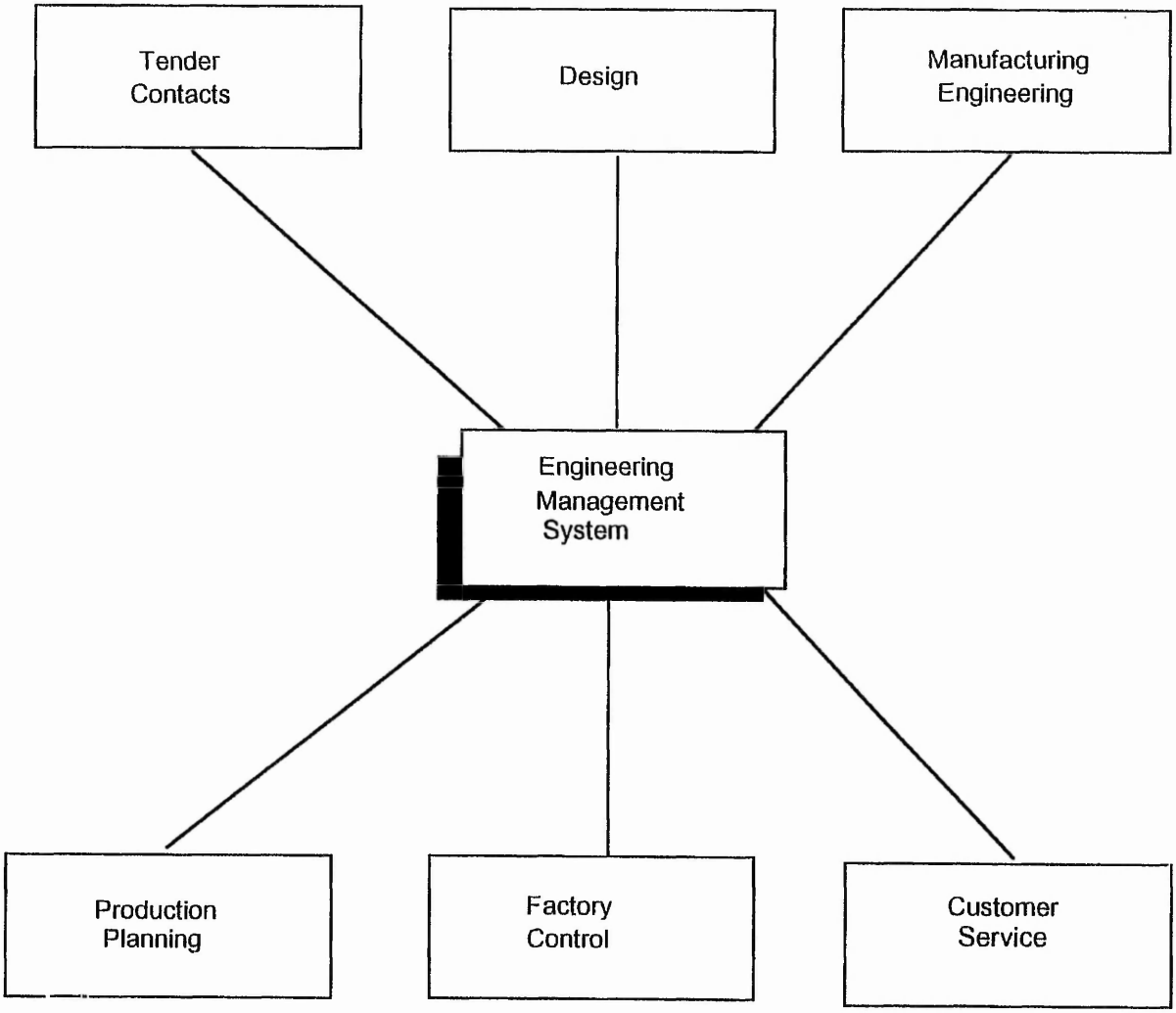


Figure 1 Functions Supported by EDM

EDM involves the management of data produced by engineers and can include drawings, bills of materials and modification information. The term management is not referring to the personnel, even though they are directly effected, but to the manner in which the data is created and stored. With computers firmly established within engineering and design offices the creation of data has become far easier and has resulted in its massive accumulation. If any information is required it is almost impossible to find the relevant piece unless the key identifier is known (e.g. Drawing/Part number). Other areas within management are change control, project management and resource management. Even though such procedures have been established for many years they are often not integrated to the Engineering Data computer systems (e.g. CAD). EDM aims to encompass these to provide a fully integrated information system.

The functions EDM systems perform can be summarised as follows :

- Management of configurations - relationships of parts and documentation in a product or project and their change control.
- Management of work - control of the priorities and allocation of work to all staff generating and commenting on engineering information.

The issues which EDM aims to address are :

- Time to market is a key issue and many companies have reduced their production lead time and now wish to reduce design time
- Customers are now demanding that a supplier must not only conform to quality standards for its products but also for its business processes
- As greater competition causes product life cycles to reduce both the complexity of the products and the use of multiple technologies in their design are increasing.

- To enable all engineering departments to work in parallel rather than in series, concurrent engineering requires rapid access to a common set of up-to-date engineering information.
- Companies need to reduce the high cost of not getting the right information to each department at the right time
- There is little extraction of management control information from existing computer based systems.
- The volumes of data to handle are very high, measured in many thousands of engineering items. In addition the relationships of the various versions of parts and associated documents must be held.
- Trying to use the production database for holding the design views of product configurations is not satisfactory. [1, 3, 34]

In order to achieve the above the EDM system must be able to provide

i, Integration

ii, Control

iii, Management

Within "One of a Kind" manufacturing the business focus is on the ability to design and subsequently manufacture the designs. Hence the greater standardisation the less the design costs. EDM tackles this by providing a method of retrieving data by factors other than the key identifier.

JL Burbidge et al considers integration to be the key to good manufacturing information systems.[2]. EDM aims to eliminate waste and this can be achieved through integrating functions and data systems. Within this area of computer systems EDM involves the development of an engineering database which includes the following fundamental information:

i, Part descriptions and identifier

ii, Bills of materials

iii, Design change information (Revisions and issues)

Other information may be included, however the above represents the core. As this data is maintained centrally all document types can access it and personnel can search one system for the majority of their information. The advantage of this is that documents use the same data which removes transcription and interpretation errors and maintains data consistency throughout. [20 & 24]

Another key factor is the control of data input and data movement to ensure that it is converted to usable information for all personnel. This involves establishing a set of rules concerning the method of describing parts and drawing parts. Traditionally rules for describing parts have rarely been used instead part coding systems have been adopted to help others find information. Now that computers have become well utilised within industry an opportunity has been presented to reduce the need for complex coding systems e.g., Brisch is currently used within Brush Transformers Ltd. Descriptors can be used within a database and attached to the unique identifier of the part with the advantage that the information is understood by many more personnel and wild card data search routines can be introduced. Once the database contains sufficient data, personnel can search on generic description, specific descriptions or strings within the description. In fact through the use of a good database management system it should be possible to search on any field.[22]

Management of the movement of data has been operating within many manufacturing companies and can be seen in the form of drawing approval/checking, archiving and retrieval. Modification information is another example within a design environment. EDM does not aim to reduce the flow of data within an organisation, although inherently it should, but aims to enforce and police the passage of information between critical points. Once a drawing is complete the draughtsman will pass it to his mentor for approval who will ensure that it is free from errors and suitable for manufacture. EDM should provide a system which enforces and monitors the flow of data between personnel and departments. In conjunction with this such a system must provide access control to allow certain users to perform tasks while not allowing others. An example of this is approval procedures where only supervisors have the facility to electronically approve data or where senior management have access to contract costs.

2.2 The Essential Features of an EDM System

Distributed file management is a fundamental requirement for all but the smallest applications, and is an absolute requirement for enterprise wide systems spanning multiple locations. It significantly effects overall system performance and the ability to share information effectively because the typical company designs and manufactures at more than one site. Distributed file storage allows these users to put the files at the location where they are most often used, while retaining central

management of the information. In this manner, users do not have to compete for network resources or time on a central server.

System interoperability is also critical for success in all but the smallest manufactures. It significantly affects the ability to share information effectively and maximise the return on hardware investment. The reason being that the typical manufacturer has acquired a number of different CAD/CAM/CAE tools, more often than not running on several different platforms.

An electronic document vault is a central component in any EDM system. It is a logical concept in that electronic documents (CAD drawings, specifications, N/C programs etc.) are not physically placed in a special secure place or on a central server. The EDM system manages the electronic documents, controlling access based on the role of the user, the documents status, what project it belongs to, and other business specifics.

Configuration management, or the identification, recording and statusing of product structures and associated documentation, is another core component to an EDM system. This capability supports the creation, maintenance and revision process of relationships between documents, parts and the overall product structure. It provides the context and intelligence for how documents and products relate to one another. However, process management is perhaps the most important piece of all for creating positive changes in the organisation and realising the full potential of EDM systems.

It is the process management features which enable the interactive real-time communication between team members working on the different parts of a product, allows automation of the engineering to manufacture release process, and the implementation of automated change systems.

Flexibility, configurability, and an open architecture are the final ingredients in a successful EDM system. The system must be flexible enough to facilitate quick changes to the processes and structures used to manage the data. Otherwise it becomes difficult to encourage and enable continuous improvements in the ways in which product and product updates are brought to market.

The EDM system must be easily configured to the particular needs of the company. The savings advantages of "off the shelf" functionality should be provided, while still allowing the organisation to implement the proprietary processes which make it a competitive force in its markets.

2.3 The Benefits of Implementing an Engineering Data Management System

The main benefits to be gained from EDM technology are: the reduction of lead times by changing from a serial to a concurrent engineering environment and reducing duplicated effort; the improved quality of products and services; and getting designs right first time by reusing relevant data in a timely manner. Product costs and

operating costs are reduced which result in a company which is better equiped to compete and generate greater profits.[3]

Companies who have implemented EDM systems have achieved major benefits. As well as improving quality one UK electronics company has reduced its overall design lead time by 50% and its time to process changes from 14 to 4 weeks. Progress on the generation of engineering activities can be established at the touch of a key rather than by lengthy meetings. It also saved 4 million pieces of paper a year, a small wood. After implementation tangible savings were valued at 1 million pounds per year. An American company saved 10% of their 300 engineers' time by reducing the time spent looking for information through implementing a data management system giving a saving of 750K per annum.[1]

Racal Avionics originally operated a manual drawing office with 20 staff and according to the management there was little data visibility and the process time for changes was taking months. The solution they decided upon was to buy in an EDM system and resulted in greater data visibility and much more efficient and effective systems. Overall, it was estimated that the savings were £250,000 per annum plus intangible benefits.[4]

EDM is of particular importance to companies whom have large design functions, as at Brush Transformers Ltd, which are the operational focus of the business. Within

Brush Transformers Ltd the design team consisted of 50 design engineers (10% of the work force) who generated engineering data via computer and paper systems. It was estimated that at Brush Transformers Ltd £60,000 would be saved through the removal of duplicated effort and that the £0.5 million rectification costs would be considerably reduced.[5]. I consider that these savings were insignificant when compared with the intangible benefits which would result from the application of an EDM system. Reduced lead times and improved quality would serve to enhance the market perception of the company. Additionally, as costs are cut due to greater part standardisation and better design profit margins can increase or more competitive prices provided to the customer.

3.0 Planning for the Development and Implementation of an Engineering Data Management System

3.1 Project Plan

A fundamental project plan was developed as outlined below :

i, Systems analysis to establish how the engineering data systems operated and the associated problems.

ii, Determine system's requirements.

iii, Software Review

iv, Development of a functional specification

v, Financial Justification

vi, System design and development

vii, System testing

viii, System Implementation

xi, System Enhancement

It was considered that the above would take approximately 2 years with the majority of time being spent performing analysis, system development and testing. Once the system was fully installed system enhancement would be required.

3.2 The Role of the Project Leader

There is an obvious need to manage an EDM implementation with care. The project manager will be responsible for introducing a system which could make or break the

supporting infrastructure around design and development. Good project managers will make all the difference. He or she will be key to ensuring that they have a direct input to activities associated with :

- *Project Planning and Control* - Handling the development timetable with due regard to hardware requirements to users, awareness, training and a phased introduction. In a project based organisation the phasing is particularly important.
- *Getting close to customers* - Ensuring that user requirements are assimilated in an environment which has sufficient business knowledge to propose innovative solutions
- *Defining the Functional Specification* - Building and reviewing the Functional Specification. The specification remains a very important milestone in the development and it requires a good deal of upfront effort. Its form may vary and it may be appropriate to make use of proptyping tools to communicate and agree system functionality.
- *Counter Balancing IT Driven Initiatives* - Championing the users' cause in the face of often forceful IT Department. If EDM is initiated as an IT project this aspect to the project manager's task is very important.
- *Providing the Link Between Vendor and Company* - EDM projects will often involve consultancy support from the EDM vendor. There is need for independent and neutral handling of the relationships with the vendor. A combined team made up of your own and vendor staff can be a very successful combination. The essential ingredient is to provide the necessary liaison to make this a partnership based on mutual trust with clear agreement on responsibilities.[10]

3.3 Systems Analysis Method

In order to help with analysis and communicating ideas to the management and staff we decided that the Group Analysis technique : Joint Application Design (JAD) was the most suitable. JAD was developed by IBM to increase the speed of systems analysis and reduce the frustration caused by lengthy structured methodologies. Whitten et al state that JAD aims to overcome the fact finding bottle necks associated with analysis.[6]. This is the process of collecting opinions, facts and priorities from the community of intended users. During analysis they address the problems and opportunities in the current system, user requirements and technical solutions. The bottle necks occur in two areas :

i, Time required to perform these activities

ii, The conflicting data gathered due to peoples conflicting views, opinions and priorities.

JAD was relatively simple to operate. It involved grouping together personnel, managers and staff, who would become the end users of the system and setting up group discussion sessions. I served as the discussion co-ordinator and mediator and consider that there were three important ingredients for success. Management must be willing to release their own time and that of their staff and the discussions must be rigidly structured. The benefit of a rigid structure was that the group understood the

objective and the various discussion routes which were to be embarked upon. This invoked thought prior to the sessions and served to enhance the results. However, the disadvantages were that you often received opinion and investigative time was required to ensure that perceptions were true. Initially, I considered that this may serve to slow progress but the intangible advantages such as personnel involvement and communications counter balanced this. Through the use of JAD I was rapidly able to establish how the current systems operated which is detailed in the following sections.

3.4 An Overview of Engineering Data Systems and the Associated Problems at Brush Transformers Ltd

In order to generate an engineering drawing the draughtsman received a design information pack which contained the majority of details required to design the transformer and its components. Prior to the creation of a drawing a new number was obtained from the drawing register which contained all the used numbers with brief descriptors. Having obtained a new number a blank CAD drawing sheet was opened and filed using the new number as the file name (See figure 2). Using a macro a drawing border was placed on the drawing and the relevant details typed in. Figure 3, a typical drawing, illustrates the information contained within the border. Whilst generating the geometry the draughtsman also developed the parts list which was written onto a piece of paper. Once the parts list was complete it was typed into an Excel Spreadsheet and printed out, ready for approval (See Figure 4). The same information was subsequently entered onto the CAD drawing. Once complete the

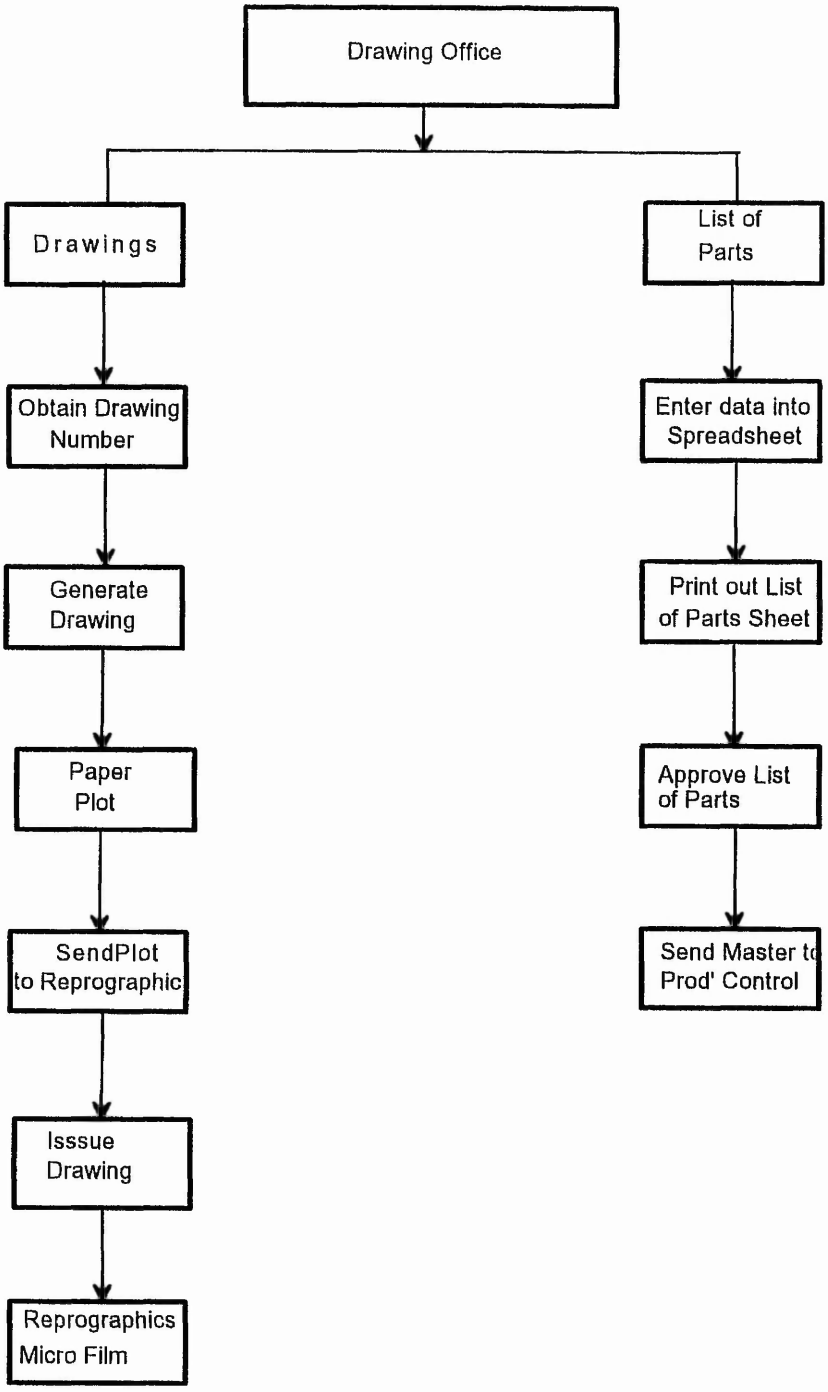


Figure 2 Drawing and List of Parts Generation Procedure

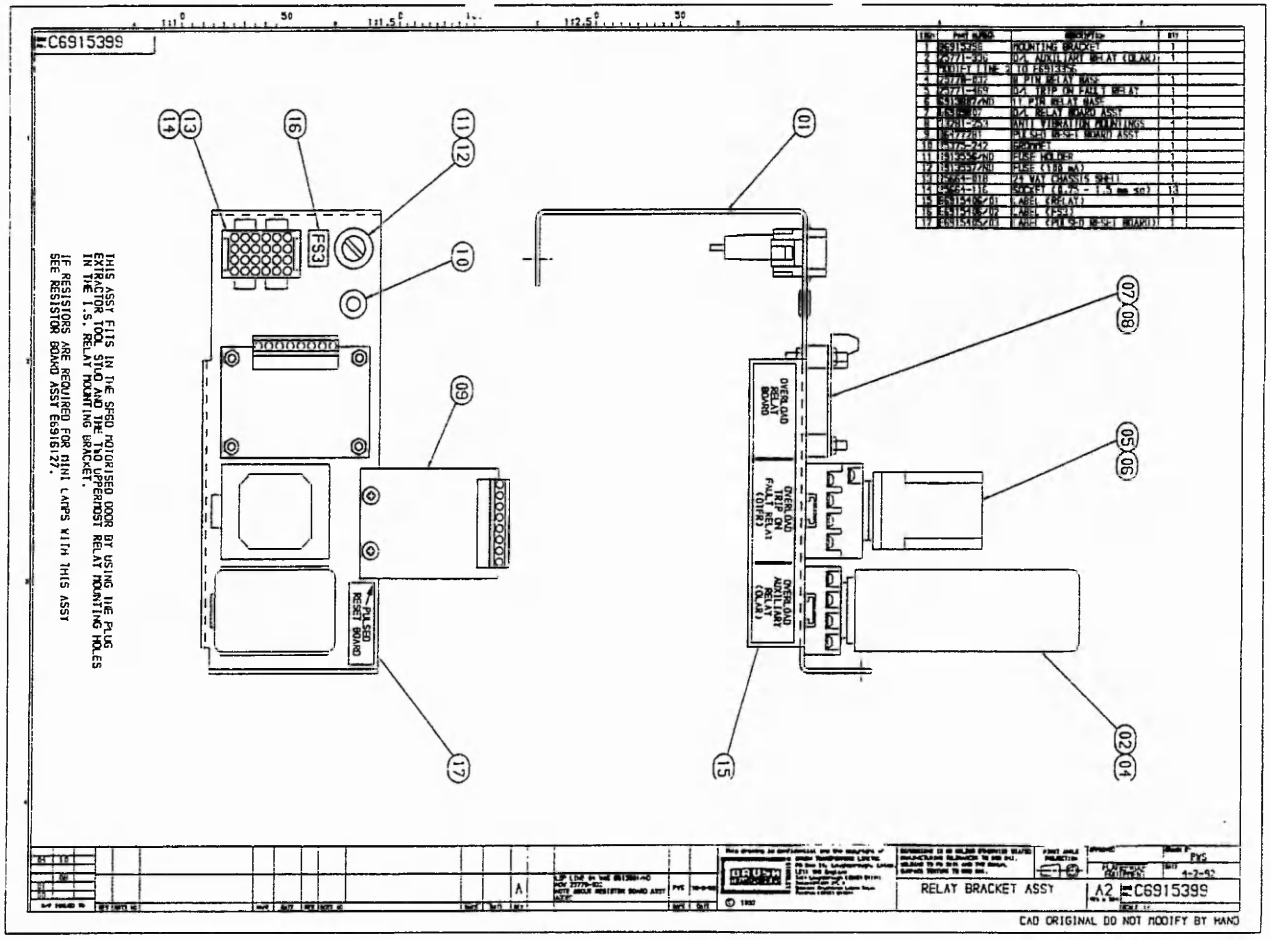


Figure 3 Sample Drawing with List of Parts

Form No. 529/3/90			Compiled By	PWS			L.O.P. B6915901
SF6N/DRM	Size Frad GF & JSC300		Approved By		Title WIRING DIAGRAM		SHEET 1 OF 1
			Date	15-9-92	Storeman	Contract No.	
Filing Code	Rev				Marshalled	Dates	Batches / No. Off
	By						
	Date						

Rev	Line No.	Part No.	Description	No. Off	Code	Qty Req'd	Man	WO /Pur. Req.	Qty Issued	Short Note
	1	13446-503	16/0.2 PVC WIRE	AS REQ	GREEN/YELLOW					
	2	13446-504	16/0.2 PVC WIRE	"	BLACK					
	3	13474-314	24/0.2 PVC WIRE	"	BLACK					
	4	13474-315	24/0.2 PVC WIRE	"	GREEN/YELLOW					
	5									
	6	13474-334	30/0.25 PVC WIRE	"	BLACK					
	7	13474-335	30/0.25 PVC WIRE	"	GREEN/YELLOW					
	8	13464-991	7/0.67 SILICON RUBBER WIRE	"	WHITE					
	9	25316-117	BLACK NEOPRENE SLEEVE	"						
	10	25316-147	BLACK NEOPRENE SLEEVE	"						
	11	17554-404	GLASS ADHESIVE TAPE	"						
	12	22458-291	SPIROL BINDING	"	3mm DIA					
	13	22458-292	SPIROL BINDING	"	6mm DIA					
	14	22458-293	SPIROL BINDING	"	12mm DIA					
	15	25415-296	WIREGRIP (RING TYPE)	"	M3					
	16	25415-311	WIREGRIP (RING TYPE)	"	M4					
	17	25415-320	WIREGRIP (RING TYPE)	"	M5					
	18	25415-331	WIREGRIP (RING TYPE)	"	M6					
	19	25415-339	WIREGRIP (RING TYPE)	"	M8					
	20	25615-028	35 WAY PLUG	1	BICC RINGLOCK					
	21	25664-013	12 WAY PLUG	1	BICC VERO					
	22	25664-116	SOCKET (0.5-1.5 WIRE)	22						
	23									
	24	25415-829	PUSH-ON CONNECTOR	2	TO SUIT 2.8 BLADE					
	25									

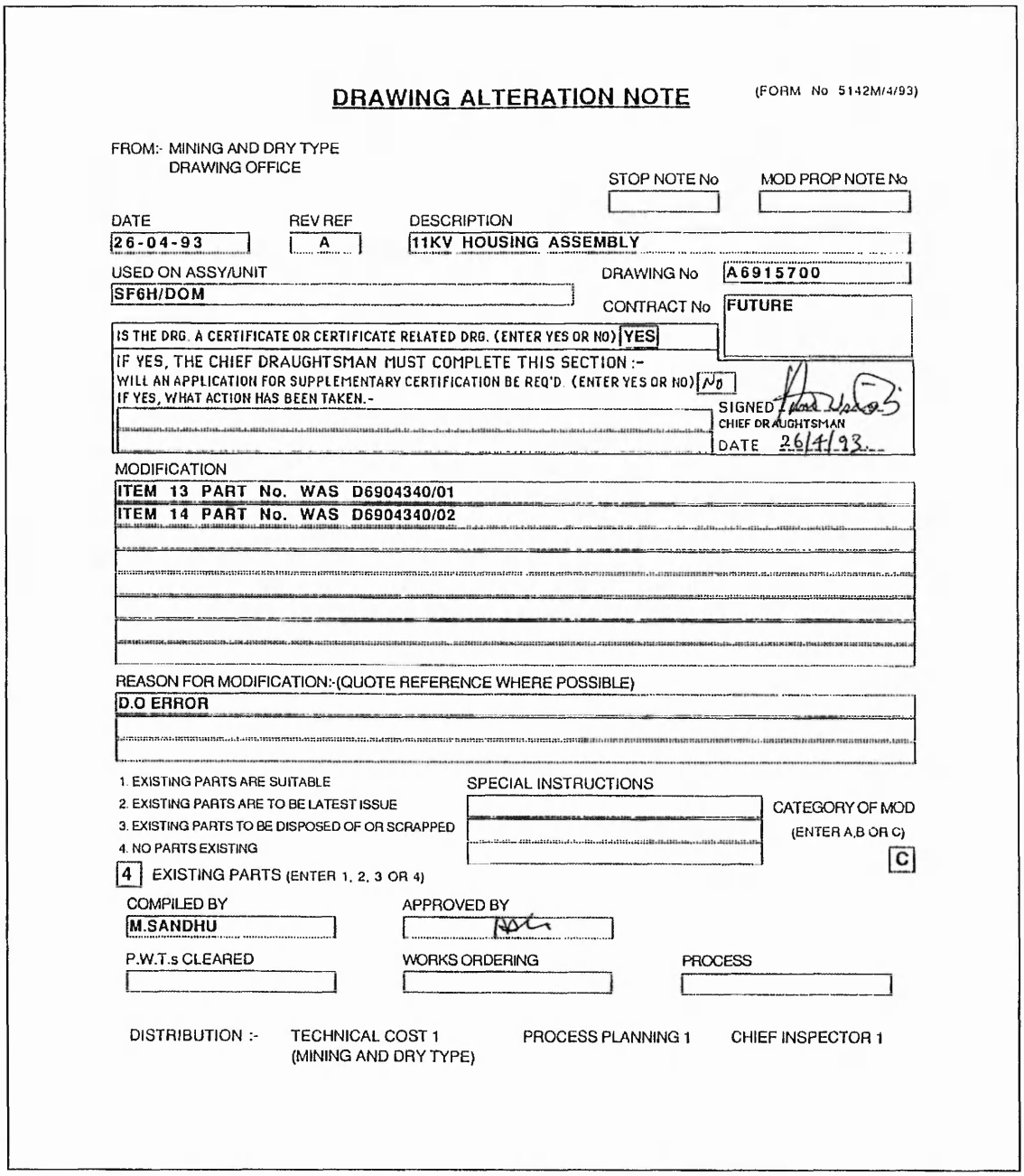
Figure 4 Sample List of Parts

drawing and parts list would be passed to a manager for a signature of approval and the drawing sent to reprographics to be copied onto micro-film as indicated in figure 2. The parts list was sent to Production Control where the master versions were kept. After the drawing was approved the CAD file was moved into the archive directories by the draughtsman. These contained approximately 2 Gbytes of data which was approximately 10,000 drawings.

When a drawing or part list was changed the draughtsmen produced a Drawing Alteration Note (See Figure 5) or a List of Parts Modification Note (See figure 6). The Drawing Modification Note was generated using an Excel Spreadsheet while the List of Parts Alteration Note was hand written. Once complete and approved they were distributed with modified drawings and parts lists to Reprographics, Production Control, Quality and Process Departments (See Figures 7 & 8). In order to modify the drawings they were moved from the archive directory to the users current working directory, modified and returned.

3.4.1 A Summary of the Problems at Brush Transformers Ltd

- High levels of duplicated effort
- Low levels of part standardisation
- A poor degree of information control



DRAWING ALTERATION NOTE

(FORM No 5142M/4/93)

FROM:- MINING AND DRY TYPE
DRAWING OFFICE

STOP NOTE No

MOD PROP NOTE No

DATE	REV REF	DESCRIPTION
26-04-93	A	11KV HOUSING ASSEMBLY

USED ON ASSY/UNIT: SF6H/DOM

DRAWING No: A6915700

CONTRACT No: FUTURE

IS THE DRG. A CERTIFICATE OR CERTIFICATE RELATED DRG. (ENTER YES OR NO) YES

IF YES, THE CHIEF DRAUGHTSMAN MUST COMPLETE THIS SECTION :-

WILL AN APPLICATION FOR SUPPLEMENTARY CERTIFICATION BE REQ'D. (ENTER YES OR NO) No

IF YES, WHAT ACTION HAS BEEN TAKEN.-

SIGNED
CHIEF DRAUGHTSMAN

DATE 26/4/93

MODIFICATION

ITEM 13 PART No. WAS D6904340/01

ITEM 14 PART No. WAS D6904340/02

REASON FOR MODIFICATION:-(QUOTE REFERENCE WHERE POSSIBLE)

D.O ERROR

1. EXISTING PARTS ARE SUITABLE
2. EXISTING PARTS ARE TO BE LATEST ISSUE
3. EXISTING PARTS TO BE DISPOSED OF OR SCRAPPED
4. NO PARTS EXISTING

4 EXISTING PARTS (ENTER 1, 2, 3 OR 4)

SPECIAL INSTRUCTIONS

CATEGORY OF MOD (ENTER A,B OR C): C

COMPILED BY: M.SANDHU

APPROVED BY

P.W.T.s CLEARED

WORKS ORDERING

PROCESS

DISTRIBUTION :- TECHNICAL COST 1 (MINING AND DRY TYPE) PROCESS PLANNING 1 CHIEF INSPECTOR 1

Figure 5 Drawing Alteration Note

MODIFICATION NOTICE AFFECTING PARTS LIST & MATERIAL ORDER SHEET
BRUSH TRANSFORMERS LTD
FORM 5060/3/92

CONTRACT No. 06/66/74781/A Date 20/4/93 Location of Negative: ~~ZF/DO~~ DO.

~~LOA~~/LOA NO. AS ABOVE Sheet No. 1 of 3 Stage No. — Grade C.

REV LINE	LINE NO	MODIFICATION	Whether interchange ability is affected
A	—	TANK DETAIL ALLOCATIONS ADDED	

TRANS PROD. CONTROL RECEIVED 16-7

Existing Parts to be : 1) *Used up.
11) ~~*Modified~~
111) ~~**Scrapped or Disposed of.~~

COMPILED BY
APPROVED BY

Reason for Modification: ADDITIONAL INFORMATION.

This modification will become effective on.....
(Date or point Production Period)

Signed : DB .....Production Control.

Figure 6 List of Parts Modification Note

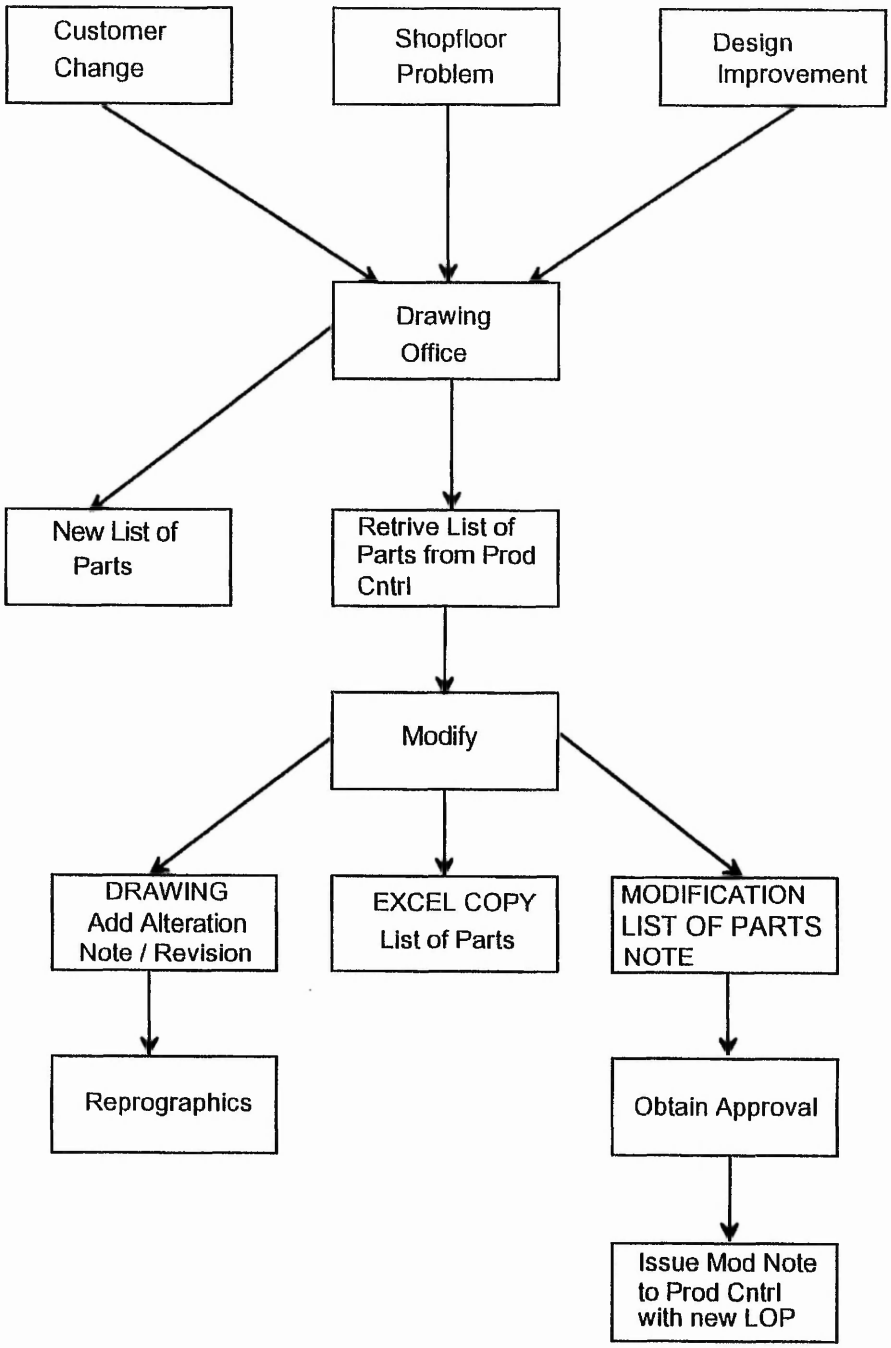


Figure 7 List of Parts Modification Procedure

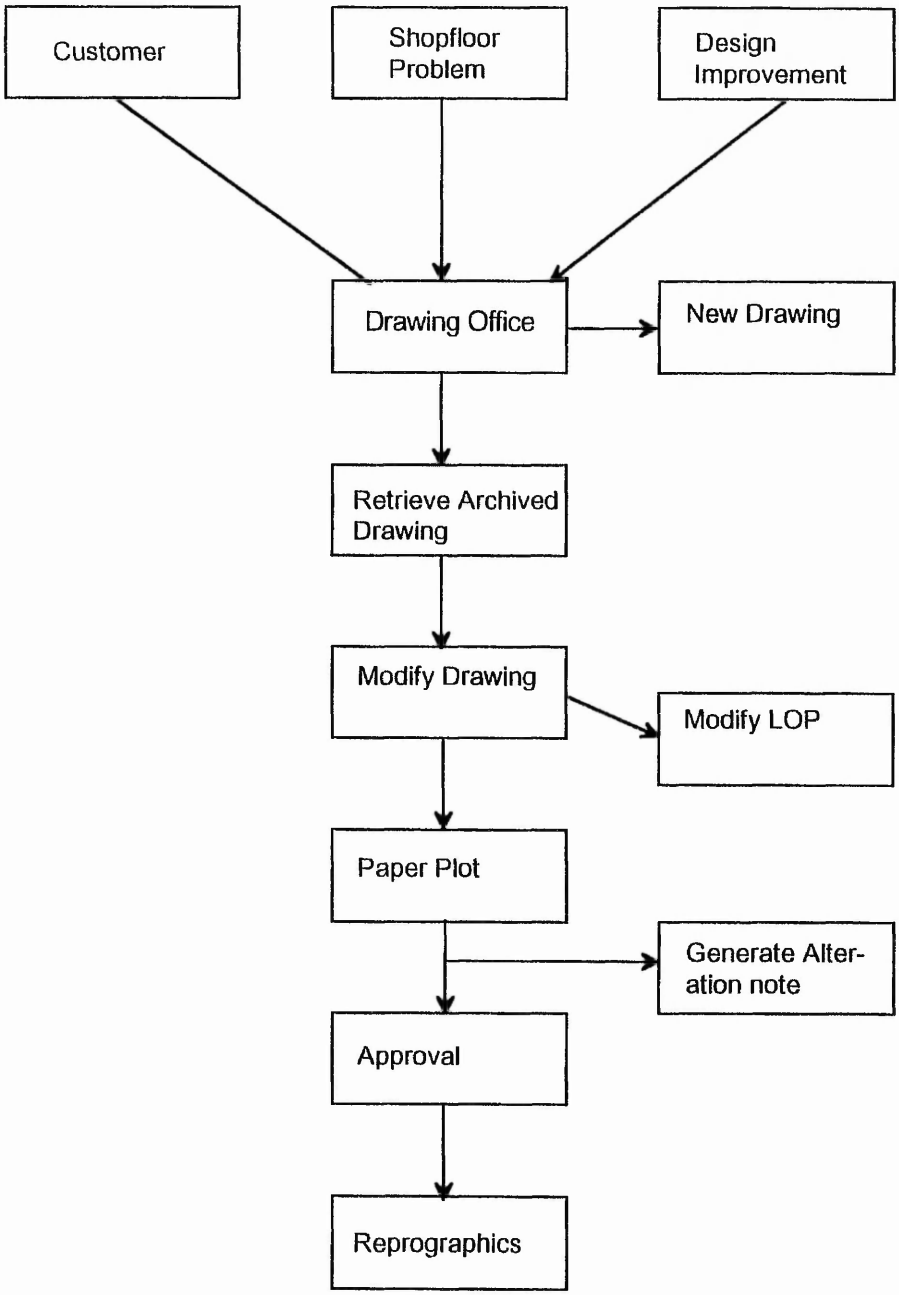


Figure 8 Drawing Modification Procedure

3.5 The System Requirements

As we had established the problems we were able to propose a number of system requirements which would address these.

- The system must improve data integration so to remove the high levels of duplicated effort.
- A part search facility must be included to increase the levels of part standardisation.
- Data entry and flow controls must be incorporated to enable improved information and more efficient systems.

3.6 Measures of the Benefits

Although it will take a reasonable amount of time before the benefits are reflected in the profit of a company we decided it was important to establish a set of measures by which the benefits may be illustrated. Sherpa with Coopers & Lybrand teamed up to study the actual financial benefits derived from EDM systems installed at over 30 companies in Europe, Japan and the USA.[7]. This enabled a methodology for evaluating the impact of EDM systems in three key areas and baseline results for the aerospace and electronics industries. The study has been encapsulated into a program which can be adopted by all manufacturers implementing EDM systems who wish to calculate the potential savings from EDM systems in three key areas as explained below :

3.6.1 Cost of Quality

Analysis was done on the costs in each area (scrap or rework) and the reasons why the costs were incurred. At Brush Transformers Ltd this was performed using the Shop Floor Quality Data Capture system.[5]. Throughout the EDM system usage the initial figures are compared with the current and the impact of the EDM system established.

3.6.2 Cost of Non-valued-added Labour

Sherpa and Coopers & Lybrand studied information handling activities in the design, drafting, production engineering, quality assurance, programme management and field service functions. Five major information handling activities were identified and quantified across these groups in terms of time spent. They considered these areas as non-value-adding as did not directly improve quality, manufacturing cost or time-to-market for the product from the customers' viewpoint. An example of this at Brush Transformers Ltd was the duplicated effort required to development engineering documentation.

3.6.3 Marginal Revenue Improvement

The study group also examined the incremental increase in revenue resulting from :

i, lead time reduction (reduce time-to-market)

ii, improved product quality

iii, increased ability to produce product variants to match specific customer needs more closely or enter more niche markets.[7]

Time-to-market impact was determined based on actual reductions in lead time resulting from the EDM system.

Quality impact was based on specific results of customer complaints and rework.

Marginal revenue improvements were based on earnings before interest and taxes.

4.0 System Specification and Financial Cost/Benefit Analysis

4.1 Computer Hardware and Software Review

Having a clear picture of what was required from the EDM system a review of the available computing tools was performed. It was necessary to identify the tools which would be required and the cost implications.

Three hardware platforms were being utilised within Brush Transformers Ltd. Approximately 40 SUN workstations, used to generate engineering drawings, were running SUN OS and Diad 2d CAD. An ICL Series 39 mainframe was used for commercial data and was running inhouse software and the MRP II package OMAC. The company also had 100 Apple Macintosh PCs which were all networked and running Excel spreadsheets and word processing packages.

Of particular importance to this research were the SUN workstations as these were used to create drawings and one was provided at each of the designers work areas. Each contained a maximum of 8Mbytes of RAM and a quarter had 120 Mbyte hard disks. The CAD package was supplied by CADCentre Ltd and in addition to the draughting functionality contained a programming language called TMP. The package was running through Sunview, a Graphical User Interface (GUI), which sat on a Unix platform. Each workstation was networked via TCP/IP on an ethernet backbone and acted as a client to the 2 file servers (hosts). Network File System

(NFS) enabled directories to be exported and mounted onto client directories. An example of this was the use of the archive directories which were available to all workstations and the provision of software by the servers. Network Integrated System (NIS), commonly known as Yellow Pages, provided global login facilities allowing users to login on any workstation.

During the JAD sessions it had been revealed that DOMA an EDM system had been purchased with the CAD software. However, it had never been used because of the systems poor functionality. As this decision had been made 4 years earlier I decided to investigate the possibility of using this as the basis for the EDM system. In order to judge its functionality a visit was arranged with CADCentre Ltd who provided a software demonstration. To provide reasonable comparisons similar demonstrations were arranged with ICL and Cimlinc who were also marketing EDM systems. Due to the expense of and limited advantages in functionality DOMA was the most cost effective software to use. The alternative options would have cost a minimum of £60,000 with additional set-up, customisation and hardware expense. With DOMA the only costs were system administration training and hardware.

4.1.1 DOMA

DOMA was the software used as the basis for the Engineering Data Management system. It included a database management system (DBMS) called TIDE which was

described as utilising the advantages of both hierarchical and relational DBMSs. Its features included the following :

Feature
Hierarchical & network data structures
Relational Features
Low level file & area locking
High integrity database
Concurrent access control

Fundamentally, DOMA was a package which provided many of the tools, described below, from which the EDM system was constructed.

- *Data Modelling* - this enabled the input of a data model which represented the data and its relationship. Whilst using this facility field attributes e.g. field name, field type (numeric, alpha, text), data entry prompt text and file relationships, were entered.
- *File Management* - This enabled the execution of a Unix script which could be modified to suit the particular file management needs. The execution of custom scripts written in unix formed many of the file management procedures. Transaction logging was also available. DOMA was delivered with one basic script as an example of file management (See Appendix).
- *Reports* - This allowed the construction and running of reports which queried the DBMS. A facility was provided to allow unix scripts to be run on the data output from the report.
- *Generator* - Custom menus were created using this feature.

Macro and batch mode facilities were also provided in the software.

4.1.2 Diad 2d & TMP

Diad 2d was the CAD package used by the draughtsmen at Brush Transformers Ltd. It was typical in its functionality and contained TMP, a macro language which enabled geometry and text to be generated automatically. Another interesting feature was TABLES which provided the opportunity to create data tables using unix data files.

4.1.3 Unix

The SUN workstations utilised a Unix operating system (SUN OS). This consisted of many compiled 'C' programs which represent operating system commands e.g. ls - list cd - change directory. Within this operating system there were many utilities and commands which could be written into scripts. As these scripts only contain commands which run 'C' programs they are called shell scripts. However there are a variety of shells with the most common being 'C' Shell and Bourne Shell. We decided to use 'C' Shell as on site training was available at Brush Transformers Ltd and there were no advantages in using Bourne Shell.

4.2 Functional Specification of the EDM System

Once we had determined the current methods used to generate and distribute engineering data and the tools available the next stage was to produce a functional

specification for the EDM system. In order to develop a realistic cost/benefit analysis we established how the system would operate and hence the resources required to develop and implement it. We were also able to focus the concepts of EDM and express them to the company's management within the context of the business. Again JAD provided a useful forum in which the concepts within EDM were conveyed and ideas on there application were formulated. Via such sessions I developed a functional specification to enable a structured design approach and to provide personnel with a clear statement of how the system would perform. The functional specification was split into the three fundamental areas of EDM as detailed by the following sections.

4.2.1 Data Integration

- Parts list and drawing border information entered into a multi-user database and the same data utilised for drawing borders, parts list, drawing modification notes and parts list modification notes. These documents generated via the EDM system.
- All documentation generated through the system to conform to company standards and the requirements of other departments and systems.
- Data held within the system must be consistent with that held on documents.
- The system will contain standard parts list which can be modified.
- A facility allowing copying of data
- The ability to view, search and print data.
- Global data change facilities.
- "Where used" report

Through the development of such facilities documents would be generated from a single data source allowing quality to rise and duplicated effort to fall.

4.2.2 Part Standardisation

- A set of standard general and specific descriptions applied to drawings of components and stored within the database
- A procedure developed to enable the introduction of new descriptions
- Standard search and report facilities readily available to users
- Ability to view on screen and print reports

The objective of this was to give users a rapid and simple method of finding data which may be re-used. Hence encouraging standardisation and reducing prototype manufacture.

4.2.3 Control

- Engineering data will be created within the draughtsman working directory. If other personnel require to modify/approve the data it must be moved to their directory.
- All drawing numbers will be electronically produced.
- A parts general description will be provided by the system
- Management will have read/write access to approval areas and approval fields
- Non-management will have read access to the approval area and approval fields
- The archive area will be set to read access to all system users.

The objective of the above was to provide the frame work and discipline for data input and document flow.

4.3 Financial Justification of an EDM System

The aim of developing a financial justification was to provide guidance regarding the investment decision and to determine whether the invest would be economically viable. When performing a cost justification it is essential that all of the cost implications are fully understood as they will be directly related to the benefits. Hence, during this research the cost/benefit analysis was performed once the resources and system requirements had been collected and understood.[8]

Investment appraisal is a management tool with the following objectives :

- To help select the most important areas of the company where initial investment in computers should be concentrated
- To help choose the correct technical specification
- To help establish the objectives and implementation plan
- To quantify all costs and savings, so that the investment is correctly reflected in the company's costing system and balance sheet
- To ensure that the investment will be profitable

Four main techniques of investment appraisal are used within industry :

i, Payback

ii, Accounting rate of return (ARR)

iii, Internal rate of return (IRR)

iv, Net present value (NPV)

Payback is the simplest to understand and is the most commonly used. It is the calculation of the payback period required to recover the initial cost of the project and uses the simple formulae :

Payback period = Capital cost / (Annual savings - Annual costs)

Using this method may slightly understate the return on investment, but it is unable to predict the timings of cash flows with any accuracy. The understatement of return is due to the method taking savings as occurring at the end of each year, instead of throughout the year. However this can be partially off set with computer investment the savings start at zero and accumulates during the year with the costs remaining at a constant level.

In order to make the investment decision we used pay back methods to perform cost / benefit analysis and combined this with the strategic advantages.

4.3.1 Cost / Benefit Analysis Using Pay Back Methods

In order to justify the expenditure we assessed the time saved through the removal of duplicated effort. We decided to take the most pesamistic view and not include the less tangible benefits within the analysis, providing the worst case scenario. These were based on average times to produce average engineering documents within a typical contract and multiplied by the number of contracts processed by the drawing

offices within a single year. Our objective was to determine the amount of duplicated effort and the associated savings once it had been eradicated.

Activity	**Time (Man/Yrs)**	**Saving (£/ Yr)**
Enter Parts List onto Drawing	2	30,000
Generate Parts List Sheet	2	30,000
Enter Modification Data onto Drawing	0.3	4,500
Total		**64,500**

Year	1	2	3	4	5	6
Cumulative Savings	64,500	129,000	193,500	258,000	322,500	387,000
Cumulative Cost	300,000	2,500	2,500	2,500	2,500	2,500

The £300,000 represents the one off hardware, software and training costs and £2500 was the increase in maintenance costs after the warranty had expired.

Payback Period = 300000 / (64500 - 2500)

= 4.8 years

The calculation above illustrates that the pay back was just under five years which was considered acceptable by Brush Transformers Ltd. In this case study a reasonable costing for the new system was established. When constructing a cost/benefit analysis it is important that both the savings and the costs are viewed with equal diligence. Often a less sophisticated proposal may achieve the majority of the savings.

4.3.2 Alternative Cost Justification Methods

Research during the investigation stage was undertaken to determine alternative methods of investment appraisal to aid future investment justifications concerning EDM systems.

A paper by M Bromwich identifies that "British companies are continually exhorted to invest Advanced Manufacturing Technology".[9]. The practices of financial institutions are said to favour this investment type which may be due to the lack of close and long term relationships between themselves and companies. Another is the lack of appraisal and monitoring systems which encompass the special characteristics of technology. Many of these investment benefits flow from their company wide impact and there interaction with other systems which may be unexpected and long term. As such investments are difficult to appraise an alternative approach is to emphasise strategic benefits.

Strategic benefits can be classified by market oriented or internal. Market oriented strategies involve product enhancement, new products and diversification, and risk reduction. Internal strategies involve giving cost advantages and improved structures and synergy.

Regarding EDM the following details the market oriented benefits which are likely to be realised :

- Improved product design
- Improved product quality
- Less risk due to more reliable products
- Reduced lead time
- Increased customer satisfaction due to improved products

In addition to the above EDM also provides the following internal benefits :

- Reduced labour costs
- Improved data visibility
- Greater control
- Improved integration
- Improved personnel effectiveness and efficiency
- Greater shop floor efficiency due to reduced scrap and reject work

Through the provisions of both types of justification a far more realistic proposal can be presented.

Whilst debating the investment decision it was also essential to determine the effects of not making the investment as this can provide a good comparison. Regarding the case at Brush Transformers Ltd if the investment was rebuked then product quality and efficiency would maintain its position or possibly slump. This was a position which the company could not afford to be in and therefore the investment was made.

5.0 System Design and Development

5.1 System Design Plan

Understanding the tools available and the system requirements was essential to comprehending the design task which lay ahead. The system design plan, below, indicates the various stages of design :

i, Develop a data model and input it into DOMA

ii, Design a method of integrating CAD drawings with data held in DOMA and construct relevant scripts.

iii, Design file management procedures and construct scripts

iv, Design and construct menus, automatic assignment of drawing numbers and general description procedures.

v, Design and construct standard reports and part search facility.

The remainder of this chapter will be dedicated to explaining the designs and procedures developed for the EDM system at Brush Transformers Ltd.

5.2 Data Model

The data model was the basis of the system and contained the characteristics of each data field and their relationships with other fields. In order to develop a data model which represented the EDM system's requirements a clear understanding of the data

used within the company and a comprehensive knowledge of how it would be reflected in DOMA was necessary.

Figure 9 shows the data model generated for Brush Transformers Ltd. To create this we held many JAD sessions and established the data fields required and their relationship to each other. The foundation of the data model was that all pieces of data used the same key attribute - the drawing number. Hence, each drawing number may have been associated with a drawing and /or a parts list and/or modification information. The outcome was that the model was required to encompass all this data. Due to the considerable number of attributes we built a number of data models which inter-linked but this gave poor system response and we returned to the use of a single data model as displayed in figure 9. This shows the relationships between data and illustrates that DRAWING NUMBER was the key attribute, i.e. that which owns all other data. Attributes which are indented below other attributes are owned by the above attribute.

e.g.. ITEM NO is owned by DRAWING NUMBER but owns ITEM REV, PART NUMBER and QTY.

Attributes which contained multiple records e.g.. ITEM NO (There are many item numbers within each List of Parts) show a (*) after them. Another useful feature was the ability to create a link between attributes. This was used to generate the List of Parts sheets as it enabled the PART NUMBERs contained under the ITEM N0 attribute to be linked to the key attribute DRAWING NUMBER. When printing Parts

Lists, general descriptions were easily retrieved from the data base (attribute GENERAL DESCRIPTION).

The implication of the above was that the database was required to hold information on every part used within the company. A useful feature of DOMA was that the attribute PART NUMBER was configured so that only part numbers which had associated records i.e.. DRAWING NUMBER attribute values could be entered. The benefit of this was that all items on a parts list consistently had descriptions.

Other attribute characteristics were entered into the data model and these included :

i, Whether the attribute was a compulsory field i.e.. had to be filled in.

ii, The display message for data entry. These appeared in the same order as the data model.

The following sections within this chapter will describe how the various software tools were used to produce an EDM system.

5.2.1 File Management and the Data Model

In order to perform file management it was necessary to build the data model to reflect this, as seen in Figure 9. The model required a TRANSACTION attribute owned by the key attribute DRAWING NUMBER to perform the file management actions and keep an audit trail.

Figure 9 Data Model

DRAWING NUMBER
....PRODUCT GROUP
....DRAWING SIZE
....PART TYPE
....DO ASSY
....DRG/REC - CREATED BY
....DRG/REC - DATE CREATED
....DRG/REC - CHECKED BY
....GENERAL DESCRIPTION
....PART DESC1
....PART DESC2
....PART DESC3
....PART DESC4
....MANU DESC1
....MANU DESC2
....MANU DESC3
....BRISCH CODE
....UNITS OF MEASURE
....CONTRACT NUMBER
....CERT NUMBER
....CUST APPVE NO
....LABOUR COST
....LABOUR COST (ACT)
....MATERIAL COST
....MATERIAL COST (ACT)
....LOP - CREATED BY
....LOP - DATE CREATED
....LOP - CHECKED BY

....ITEM NO (*) *List of Parts*
.....ITEM REV
.....PART NUMBER (---> DRAWING NUMBER)
.....QTY

....DRG - REV (*) *Drawing Alteration Note*
.....DRG - REV BY
.....DRG - REV DATE
.....DRG - CONTRACT NO (*)
.....DRG - MODIFICATION
.....DRG - MODIFICATION1
.....DRG - MODIFICATION2

Figure 9 Data Model (Cont'd)

.....DRG - MODIFICATION3
.....DRG - USED ON (*) (---> DRAWING NUMBER)
.....DRG - EXISTING PARTS
.....DRG - APPROVED BY
.....DRG - APPROVED DATE
.....DRG - CATEGORY
.....DRG - MOD REASON
.....DRG - MOD REASON1
.....DRG - MOD REASON2
.....DRG - MOD REASON3
.....DRG - INSTRUCTION
.....DRG - INSTRUCTION1
.....DRG - MOD PROP NOTE NO
.....DRG - STOP NOTE NO

....LOP - SHT NO *Parts List Modification Note*
.....LOP - REV (*)
.....LOP - REV BY
.....LOP - DATE
.....LOP - MOD REASON
.....LOP - APPROVED BY
.....LOP - STAGE
.....LOP - GRADE
.....LOP - ACTIONS
.....LOP - ITEM NO
.....LOP - MODIFICATION
.....LOP - CONTRACT NO (*)

....TRANSACTION (*) *File Management*
.....TRANSACTION TYPE
.....AREA (---> POOLS)
.....DATE
.....FILENAME (*)
.....BY WHOM

POOLS *File Management & Control*
....DIRECTORY
....USERS TO (*)
....USERS FROM (*)
....CONTAINS (*)
....USERS IN (*)
.....FILTER

The TRANSACTION owning attribute was a multiple attribute and contained the transaction number.

e.g.. If a drawing was created, then moved and subsequently archived there would be 3 records containing numbers 1 - 3.

Within attribute TRANSACTION TYPE record 1 would contain the word CREATE and 3 would contain the word ARCHIVE. This was dependant on the system design. The attribute AREA represented the area where the action was to occur.

e.g.. If the drawing was moved from ROB to JOHN the second record would contain the value JOHN.

DATE represented the date on which the transaction occurred while FILENAME represented a list of files which were to be effected by the transaction. Attribute BY WHOM took the name of the user who was performing the action.

When a record was input into the system it initially entered an approval area which was hard coded into DOMA. The record was then registered and during this process it was ensured that all compulsory fields and correct data types had been entered. Additionally, the system looked for a transaction value of "next" and performed the relevant instructions in the record. An example of a series of transaction records is shown below ;

Transaction	Transaction Type	Area	Date	Filename	By Whom
1	CREATE	ROB	01/09/92	61499700.DR	R BOOTH
next	MOVE	JOHN	10/09/92	61499700.DR	R BOOTH

The above shows that drawing file 61499700.DR was created by R Booth in area ROB on 01/09/92 and it was to be moved to JOHN on 10/09/92. To create and move the files the action of registering a record executes a unix C Shell file called domafm_ex.csh. This script was supplied with DOMA as an example and was modified to include improved system feedback and to call a variety of specific file management scripts.

Another important section of the data model was that contained under the attribute POOLS which provided the majority of system control. DOMA gave the facility to have work pools where users would store records. Attribute POOL contained the names of all the users and attached to each were a number of control mechanisms. DIRECTORY was used to store the users home directory path and USERS TO and USERS FROM stored the names of users who were able to take records and files to and from the user's pool. CONTAINS stored the names of the files and records currently in the pool. Attribute USERS IN contained the usernames of all those users who were allowed to modify or view data within a pool. To each name we assigned a FILTER value e.g.. Draughtsman or Checker. The relevance of this was that whilst entering the data model we had assigned these names to the key attribute. Subsequently each attribute was defined with the rights of each of these filters. This was utilised when allowing only checkers to enter data into the approval fields and for create read only access to areas within the system.

5.3 *Creating a New Drawing and Parts List*

The EDM system required that the drawing number was automatically created and that a general description was assigned to the part. To achieve this we utilised the DOMA menu generation system called Generator and developed a menu with each of the general descriptions agreed within the JAD sessions (See figure 10). Once this was selected it activated a DOMA script which passed two variables into a unix C Shell script. These were the general description and a 3 digit prefix which represented the first three numbers of the new drawing number. The part numbering system at Brush Transformers Ltd consisted of 9 digits with the first three representing the generic part type, the proceeding 4 being unique identifiers and the last 2 used for references to parts within a drawing. Hence, several sets of numbers were being incremented in parallel. To overcome this a file containing the latest unique identifier was created for each 3 digit prefix and the unix script read the contents of the relevant file and incremented it by one. The resulting value was then replaced into the file and converted into a string and merged with the prefix to equal the new drawing number. Both this value and the general description were written into a DOMA template script along with the users product group obtained from his working directory path name. A DOMA template script was a program which contained a series of actions and dummy values. Using the sed unix command the dummy values were replaced with the real values and saved as a new copy within the users working directory. The original DOMA script then executed this script which created a new record with new values. The INTER command presented the data entry screen to the user as illustrated in

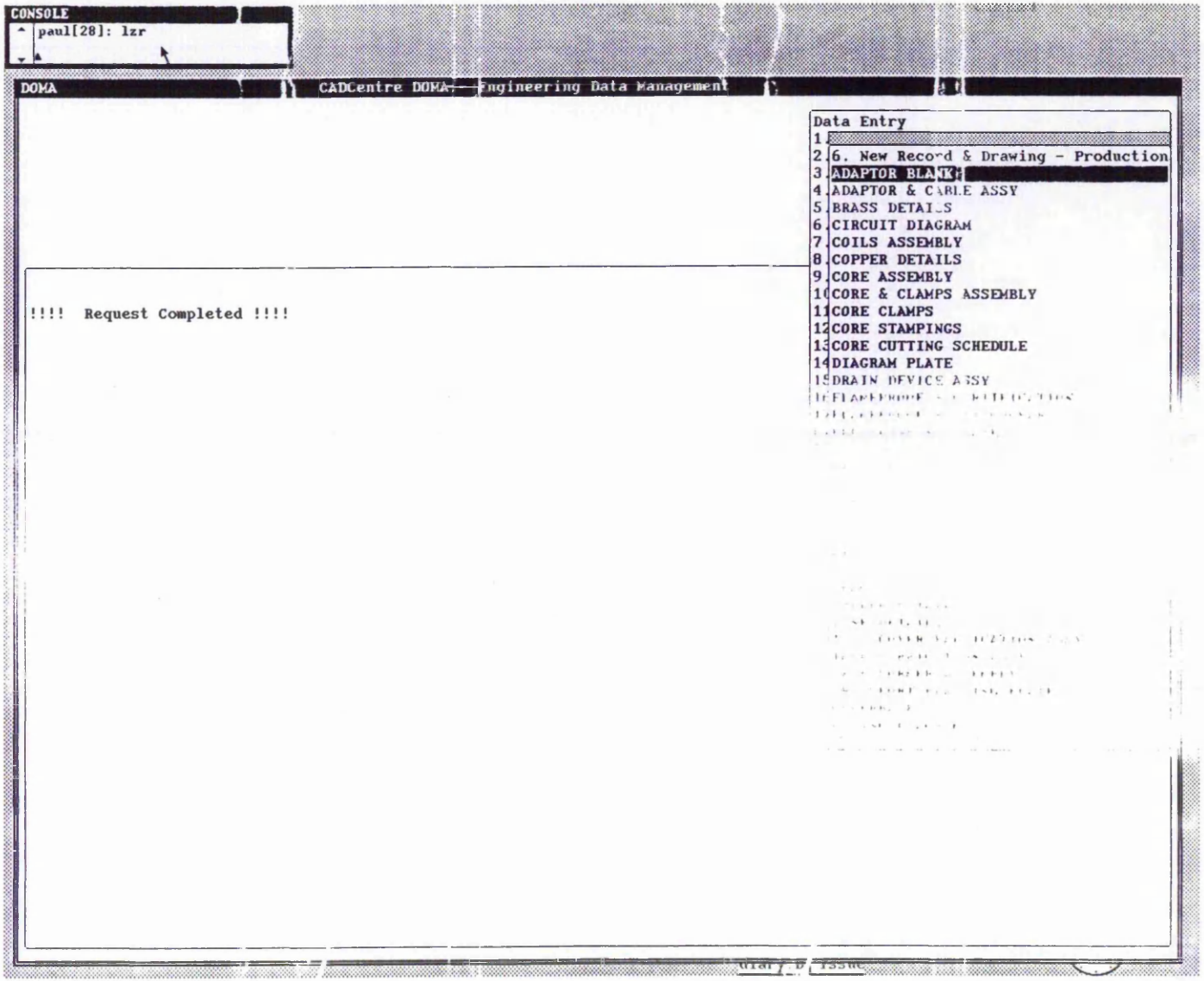


Figure 10 DOMA General Description Menu

figure 11 so that the user may enter data against the drawing size field and specific description fields. Through the data model these fields were made compulsory.

e.g.. DOMA template script

```
CREATE  KEY ATT DRAWING NUMBER 'XXXXXXX'
        PRODUCT GROUP:='GGGGGG'
        DRG/REC - CREATED BY:=$doma_user
        DRG/REC - DATE CREATED :='AAAAAA'
        GENERAL DESCRIPTION := 'ZZZZZZZZZ'
        TRANSACTION:='next'
        TRANSACTION TYPE :='CREATE'
        AREA:=$doma_user
        DATE='DD/MM/YY'
        BY WHOM :=$doma_user

        INTER
```

Once the data had been entered the macro took the record through registration and performed file management. As previously described file management called a script which contained a variety of cases. DOMA passed the case CREATE to the script which then performed a series of actions. We also designed it to pass the drawing size to the script which then chose a blank drawing of the correct size and placed it in the users working directory using the new drawing number as the filename. To achieve this a report was activated from the DOMA script which output the drawing size and this value was piped into a file which was read by the file management script. At the same time DOMA placed the record within the users pool and the data was set up for management through the EDM system.

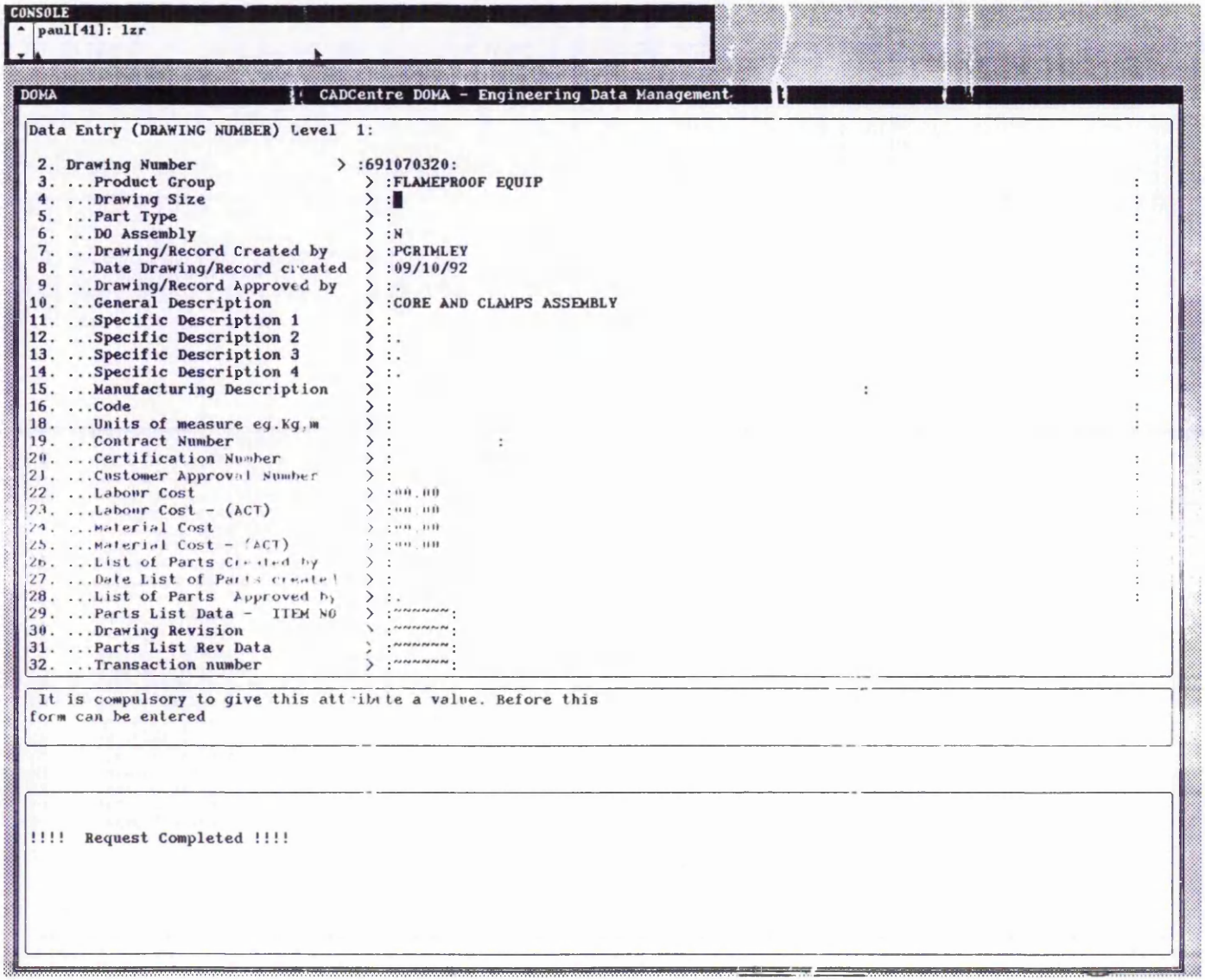


Figure 11 Data entry screen

Whilst testing the above system we encountered problems generating new drawing numbers as it was possible for parallel access to the file containing the last number. To avoid this a lock procedure was placed in the unix script as the first action which placed a flag in the directory containing the number file. The script looked for the flag relevant to the prefix before accessing the number file and if present would loop until the flag had been removed. Flag removal was performed by the user who had gained access to the number file first through the unix script.

In order that the drawing be maintained by the EDM system a series of Diad 2d macros were written and set up to be executed from the Diad 2d menu options. The first one opened the drawing and stored the drawing number as an environment variable which made the value available to all subsequently run macros. Within Diad 2d TMP the command to open a drawing file was the same as that used to create one. This posed a problem as when opening the drawing file it would enable typographical errors to go unchecked allowing a drawing to be generated outside the system. To overcome this a C Shell script was called from the TMP macro which checked that the file already existed. If it did not then an error message was displayed indicating that an incorrect number had been entered or the drawing had not been created by the EDM system.

Once the drawing had been opened the next task was to place a border onto it. It was important that this procedure could be used for both new and old CAD drawings because it would simplify training. The process was designed so that the user selected a menu option from Diad 2d which executed a TMP macro. This created the border

geometry and a number of tables. Tables were a useful feature within Diad 2d as they enabled unix files to be displayed on the drawing. However to achieve this it was essential that the text files were correctly formatted. In order to produce the following effect :

Item No	Drawing No	Description
1	691705000	Bolt
2	689234000	Washer

the unix file had to be formatted in the following manner :

| 1 | 691705000 | Bolt |
| 2 | 689234000 | Washer |

The outcome was that for each of the tables a file of the above format was required. Figure 12 illustrates the information utilised by the drawing border which included the following :

- Drawing number (top left & bottom right)
- Last 2 modifications (bottom left)
- General description (bottom right)
- 4 specific description (underneath general description)
- Originator details (bottom right)
- Approval detail (bottom right)

Using the previously stored environment variable which equalled the drawing number a report was extracted from the database. The advantage of using the environment variable was that the user could never place the details of a different drawing onto the current one. Formatting for Diad 2d tables was subsequently performed in Unix. To produce the report two options were available :

i, Execute DOMA locally in batch mode

ii, Send commands to DOMA running on a host computer

After experimenting with option i, it was considered too slow due to the load up and close down stages of DOMA execution. The second method was far faster but had one major disadvantage as the command processor was only capable of executing commands sequentially. This meant that when two users sent a set of instructions to the remote system they would form a queue which resulted in delays. We tested this and found it to still be faster than the first option.

The design of this procedure was similar to that used to create a new record and drawing. A TMP macro was executed from the Diad 2d menu which passed the environment variable to a C Shell script which edited it into a DOMA template script. This contained commands to run a report which had previously been designed in DOMA and required the drawing number to access the relevant information. Once complete the DOMA script was added to end of a file which DOMA recognised as a watch file. DOMA could be set up in watch mode which meant that it used the

contents of a file as its commands instead of the key board. Additionally, once it had executed a batch of commands it only read new lines added to the file. Once the report was complete it was formatted through a C Shell (used the command nawk) which added the | field separators. The file was then read into Diad 2d tables to produce the effect shown in figure 12. Obviously, changes would be made to drawing and modifications made to the border. To reflect these the modification information would be entered into the EDM system and then the above procedure re-run to generate a new border.

A similar procedure was used to create parts lists on the drawing. Again the data was first entered into DOMA via the data entry screen and registered. Selecting a menu option from Diad 2d a DOMA macro was generated which produced a report. A C Shell script was used to format the data and it was read into Diad 2d as a table. The procedure also enabled the user to place the parts list at any location on the drawing and to include a variable number of rows. This was facilitated through a C Shell script which cut and formatted the data according to the users instructions. The number of rows required was entered via a C Shell prompt while the locations were entered through clicking on the intended position of the top right hand corner of the table. In order to easily replace the data the tables were placed on the same layer of the drawing sheet allowing all the data to be deleted without removing geometry or other text (See Figure 12).

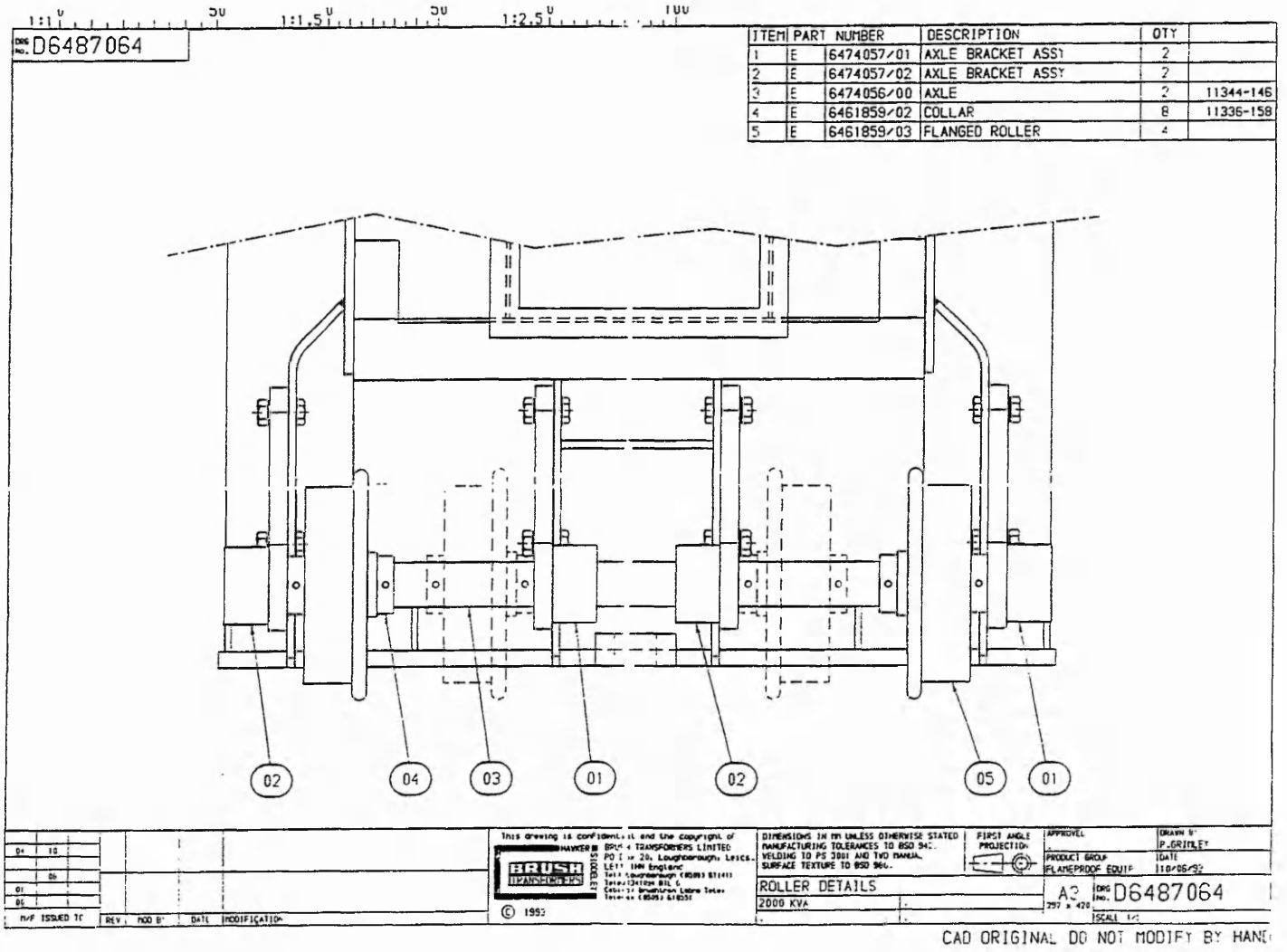
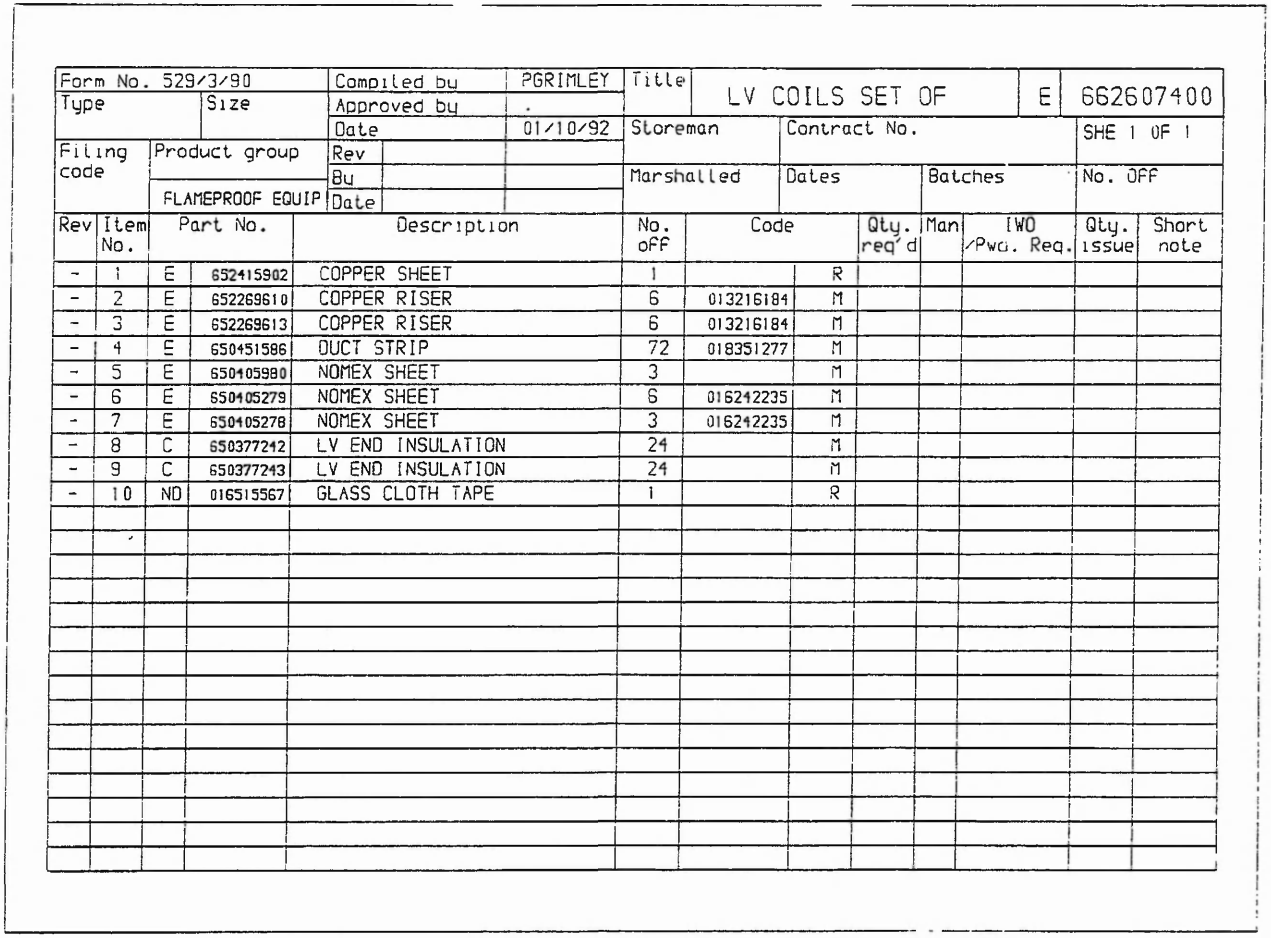


Figure 12 DOMA Drawing Sheet with Parts List

Within this particular area a problem was encountered which resulted in the procedure completing without placing the data into the table even though the table displayed the correct number of rows. After much experimentation the cause was relatively simple and was due to the sequence in which data files where created in Unix and published through NFS. Unix creates the file and then inserts the data into the file. The result was that Diad 2d was reading the file into the table before any of the data had been added to it. The obvious solution was to place beginning and end flags into each file and use these as triggers to start appending the data to the table. Unfortunately, this was not possible with the software available as no footer could be added to the report output from DOMA and as Unix performed actions in "stream" it was not possible to resolve the matter using this solution. The solution we devised was that of adding delays into the system so that once the file arrived in the directory a counter would begin giving the system enough time to insert the data. An additional solution was to increase the NFS server's RAM from 8MBytes to 32MBytes so that its speed would increase.

To generate a List of Parts sheet as shown in figure 13 another Diad 2d menu option was provided. Upon selecting this the user was prompted to enter the relevant drawing number. Again this was passed into a script which produced a formatted data file and was read into tables within Diad 2d.

The benefits of the above were that duplication of data entry was removed and a far higher level of data integration and control was provided. All information was entered



Form No. 529/3/90		Compiled by	PGRIMLEY	Title	LV COILS SET OF	E	662607400
Type	Size	Approved by	.				
		Date	01/10/92	Storeman	Contract No.		SHE 1 OF 1
Filing code	Product group	Rev					
		By		Marshalled	Dates	Batches	No. OFF
	FLAMEPROOF EQUIP	Date					

Rev	Item No.	Part No.		Description	No. off	Code		Qty. req'd	Man	IWO /Pwo. Req.	Qty. issue	Short note
-	1	E	652415902	COPPER SHEET	1		R					
-	2	E	652269610	COPPER RISER	6	013216184	M					
-	3	E	652269613	COPPER RISER	6	013216184	M					
-	4	E	650451586	DUCT STRIP	72	018351277	M					
-	5	E	650405980	NOMEX SHEET	3		M					
-	6	E	650405279	NOMEX SHEET	6	016242235	M					
-	7	E	650405278	NOMEX SHEET	3	016242235	M					
-	8	C	650377242	LV END INSULATION	24		M					
-	9	C	650377243	LV END INSULATION	24		M					
-	10	ND	016515567	GLASS CLOTH TAPE	1		R					

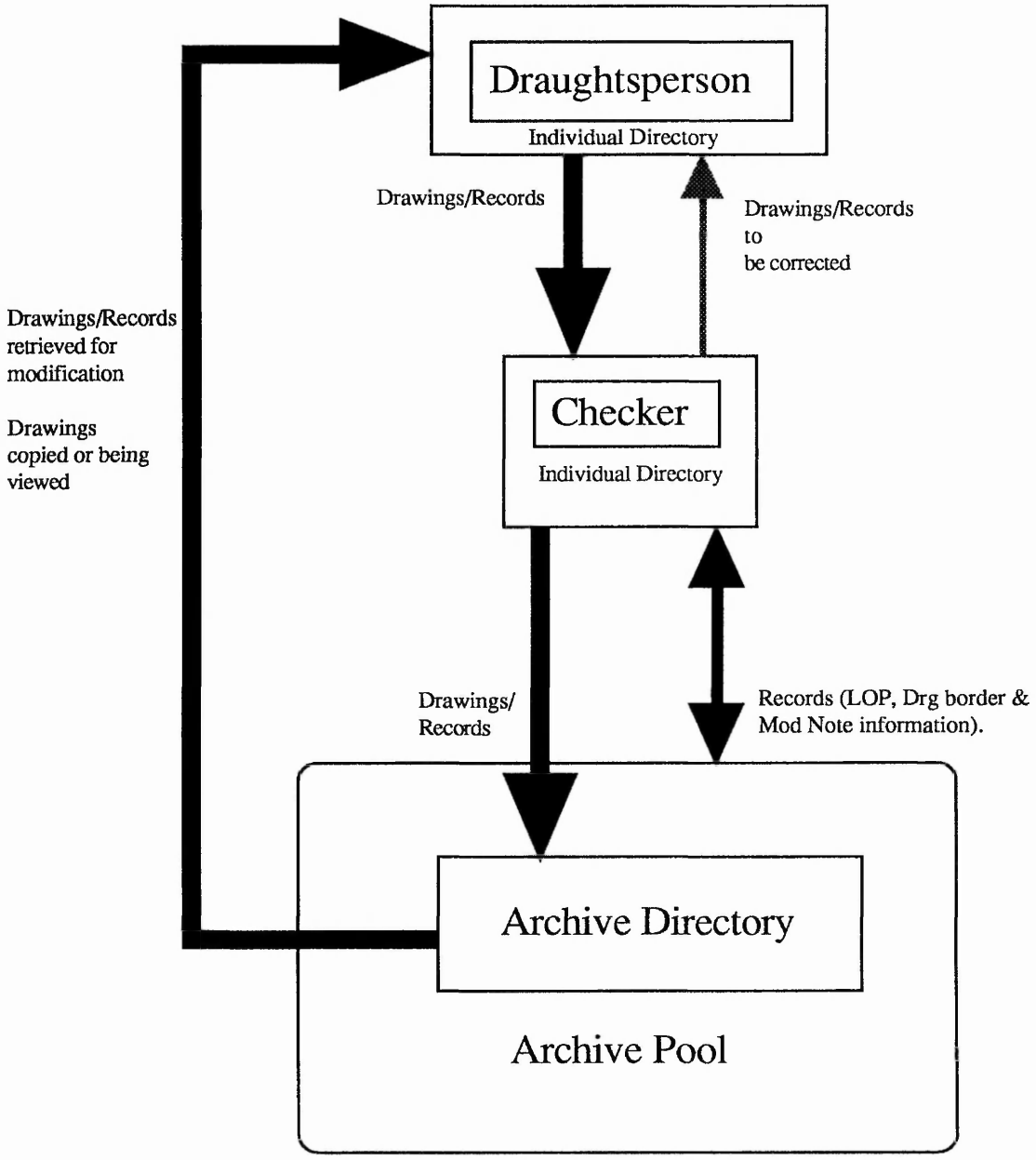
Figure 13 DOMA List of Parts Sheet

once and all personnel could access it. Documentation contained better information due to the addition of the 4 specific description and product group on the drawing border and the product group and part type on the Parts List Sheet.

5.4 Approving and Modifying Engineering Data

After creating a drawing, list of parts or modification notes it was necessary to pass the data to be checked by a manager. Figure 14 shows the procedures which we decided to adopt and included passing the data to different work pools. A DOMA menu was generated which contained the names of all the users who utilised the EDM system. Upon selecting one of the names the system was designed to prompt the user for the drawing number. Upon finding the data in the users work pool it was passed to the selected users work pool and the details of this transaction recorded. Fundamentally, the menu called a DOMA script which passed the drawing number and recipients username into a DOMA template script. The values were placed into the transaction section of the record and upon registration file management was activated. Hence the DOMA record and associated drawing were moved in a controlled manner around the system. The work pools were set up so that users could only move data into another users pool and, therefore, not take it out which maintained a high level of control by the user on his/her own data. If a user wished data to be approved then the username of the checker was selected and the data was passed into the relevant work pool. As filters had been set up on the attributes within the data model only checkers had access to the approval fields. To approve the data a DOMA menu option called Approve

Figure 14 DOMA / Approval



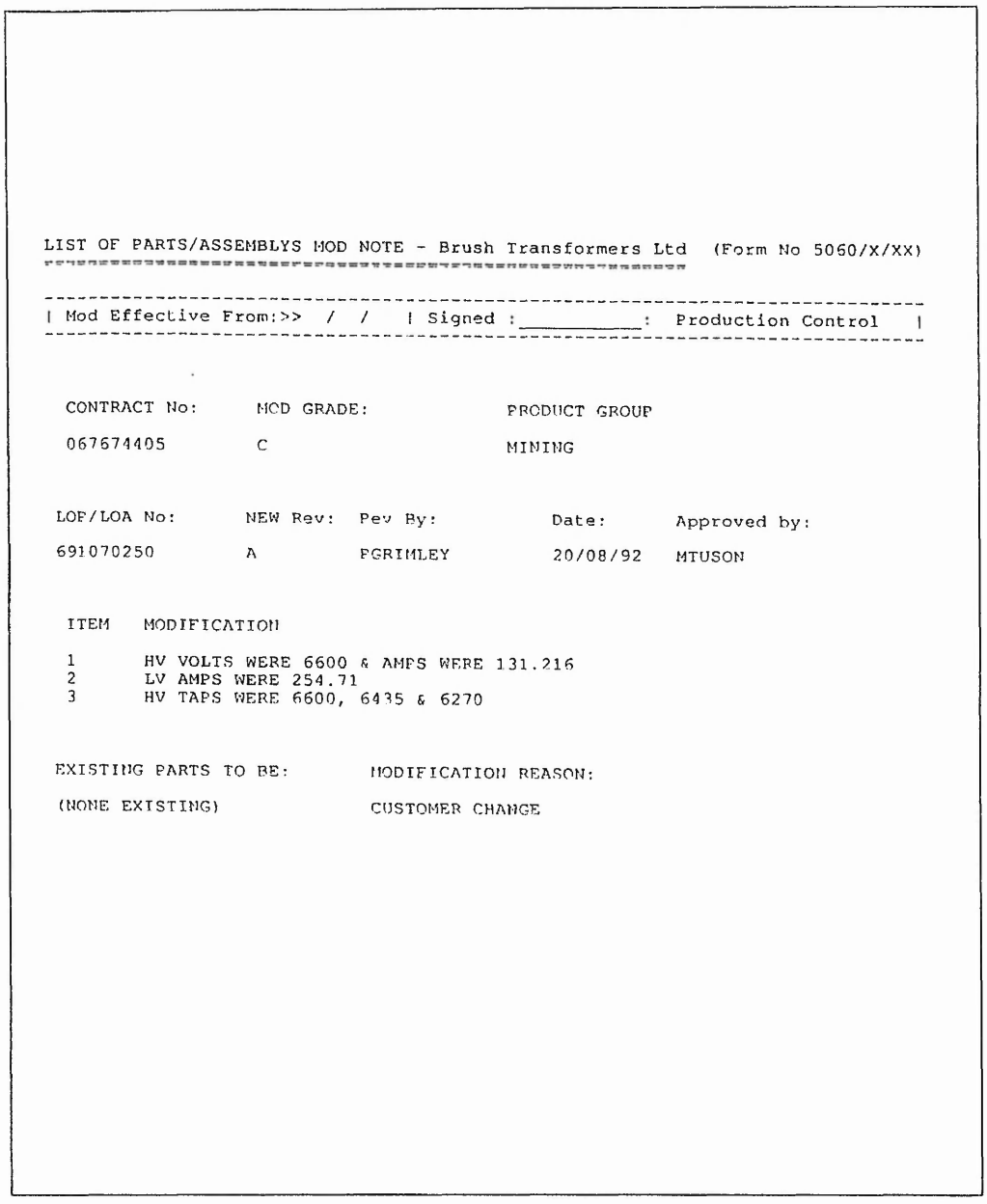
Data was set up which prompted the checker for the relevant drawing number. It was designed so that only checkers could see and use this option. Once this had been entered the option APPROVE was edited into the TRANSACTION TYPE field and presented the relevant record for data entry. As access was given to the approval fields these were filled in with the users name and the date. However, it was felt that the checkers must not be able to enter another persons name . To combat this when the checker entered the record it was registered which activated file management and contained within the APPROVE case a procedure was designed which compared the DOMA username, NIS username and the name entered into the approval fields. If any discrepancies were found then the approval procedure was halted. Once the system had passed through this stage the record and associated CAD file was passed into the archive work pool. This was set so that all the data contained was read only to all users and drawing files could only be accessed for copying, viewing or printing.

During the testing of this process we discovered a problem with checker's awareness that documents and record were waiting for approval. Initially the SUN OS mail tool was considered as a means of highlighting that approval was required. However, it did not provide a prompt or reminder and was rendered as limited in its approach. The solution to this problem was rather archaic and involved the passing of the printed drawings and documents to the checkers. This immediately prompted them and as the data still required signing did not cause any problems.

Ideally, upon approving the DOMA record the drawing should have been accessed and the approver's name added to the table in the border. Unfortunately, this facility was not possible to provide as the CAD drawing file could not be accessed unless it was open and displayed on the screen.

Often data was required for modification. To retrieve the data from the archive another menu option was generated, called Retrieve from Archive, which prompted the user to enter a drawing number. The relevant record was then modified by adding MOVE to the transaction section and registered. The result was that the data and associated CAD file were moved into the users directory ready for modification.

Once the user had modified the drawing the DOMA record needed modifying. The revision was incremented and the modification reason and actions were entered into the record. Once this had been registered a Drawing Alteration Note or List of Parts Modification Note could be produced by selecting the menu option called Modification Notes. This procedure displayed a further menu giving the above options. After selecting the relevant one the menu was designed to prompt whether the details were to be printed or sent to the screen. A DOMA macro was then executed to run a DOMA report which output all the modification details within the record. A unix script was then invoked which edited the last modification out and passed it through a text formatting script to produce the modification note (See Figures 15 & 16). In order that the details be output to the screen the modification data was piped into a file



LIST OF PARTS/ASSEMBLYS MOD NOTE - Brush Transformers Ltd (Form No 5060/X/XX)

| Mod Effective From:>> / / | Signed :_____: Production Control |

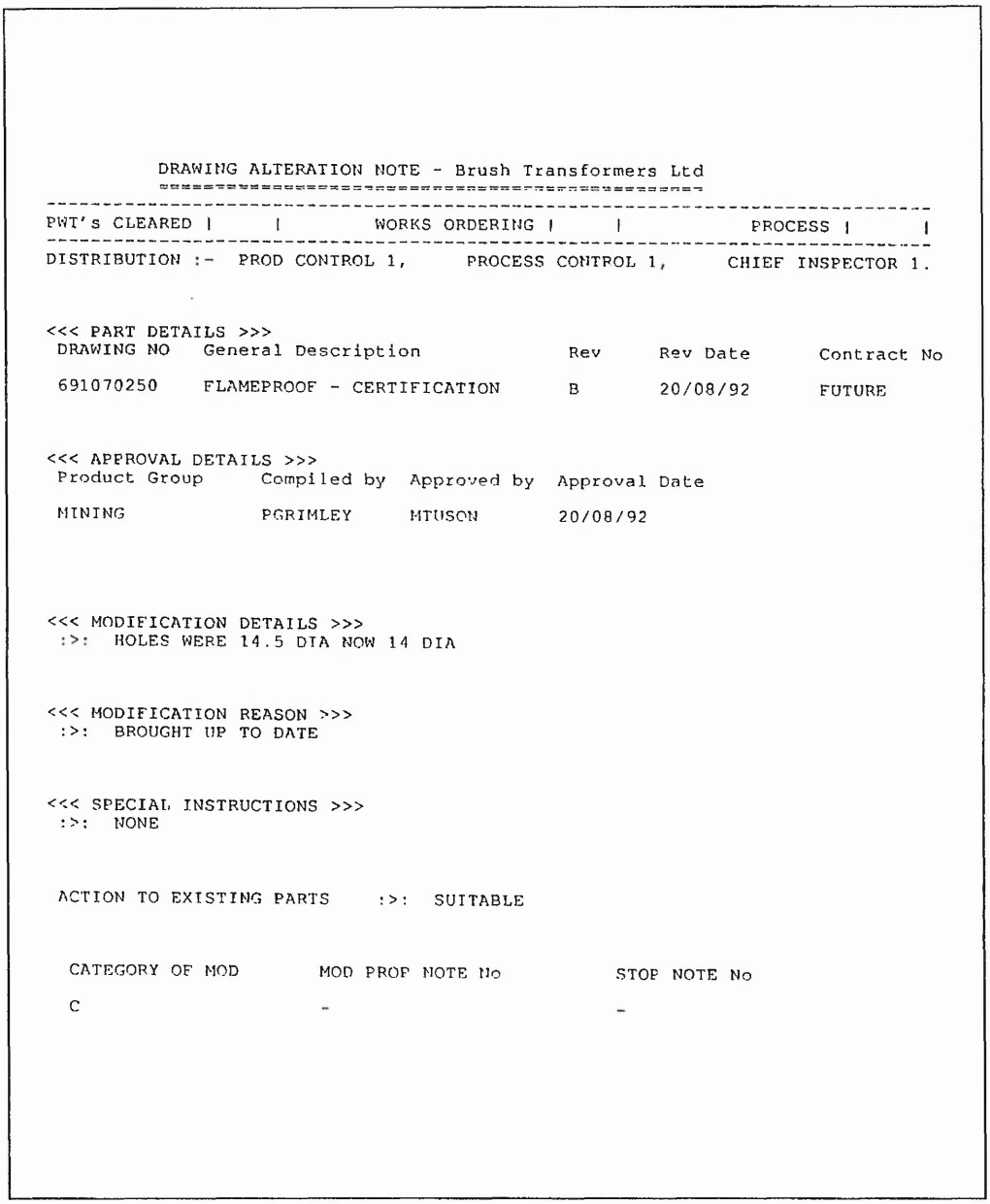
CONTRACT No:	MOD GRADE:	PRODUCT GROUP
067674405	C	MINING

LOF/LOA No:	NEW Rev:	Rev By:	Date:	Approved by:
691070250	A	PGRIMLEY	20/08/92	MTUSON

ITEM	MODIFICATION
1	HV VOLTS WERE 6600 & AMPS WERE 131.216
2	LV AMPS WERE 254.71
3	HV TAPS WERE 6600, 6435 & 6270

EXISTING PARTS TO BE:	MODIFICATION REASON:
(NONE EXISTING)	CUSTOMER CHANGE

Figure 15 DOMA List of Parts Modification Note



DRAWING ALTERATION NOTE - Brush Transformers Ltd

PWT's CLEARED		WORKS ORDERING		PROCESS	

DISTRIBUTION :- PROD CONTROL 1, PROCESS CONTROL 1, CHIEF INSPECTOR 1.

<<< PART DETAILS >>>

DRAWING NO	General Description	Rev	Rev Date	Contract No
691070250	FLAMEPROOF - CERTIFICATION	B	20/08/92	FUTURE

<<< APPROVAL DETAILS >>>

Product Group	Compiled by	Approved by	Approval Date
MINING	PGRIMLEY	MTUSON	20/08/92

<<< MODIFICATION DETAILS >>>
:>: HOLES WERE 14.5 DIA NOW 14 DIA

<<< MODIFICATION REASON >>>
:>: BROUGHT UP TO DATE

<<< SPECIAL INSTRUCTIONS >>>
:>: NONE

ACTION TO EXISTING PARTS :>: SUITABLE

CATEGORY OF MOD	MOD PROF NOTE No	STOP NOTE No
C	-	-

Figure 16 DOMA Drawing Alteration Note

rather than the print command (lpr). The DOMA command DISPLAY was then used to display the files contents within the DOMA window.

Obviously, it was essential that both drawing and parts list reflected these modifications. Hence it was necessary to create a new border for the drawing and re-execute the List of Parts Sheet options from the Diad 2d menu.

5.5 Part Search

This facility was designed to address the need for greater part standardisation and the reduction of duplicated effort. The design consisted of a General Description and 4 Part Descriptions which could be searched on. When creating the part record the general description was selected and through data entry at least 1 of the 4 part description were input.

Initially we looked at a variety of part coding and numbering methods for part classification and retrieval but discovered that they were too complex for the application. It was intended that all personnel should easily be able to learn and execute the part search procedure and the most effective method was to describe the part in words. A general understanding was then provided, even though strict rules on describing features were required. Such rules were defined by the engineering management and provided on the computer for reference as a windowed text file.

As previously described the data model contained 5 descriptive attributes and a report was generated which enabled the entry of attribute values to be searched upon. DOMA provided a good facility which enabled wild card searches of the database to be performed.

e.g.. If the user required to find all part records which contained the word COIL in their General Descriptions the following value would be input *COIL*

Another wild card operator was ? which could be put in place of a single character (e.g.. CO?L)

As we had 5 fields to search on we built the report to include & and OR statements. When entering the data the user was prompted by the following messages:

General Description ---------

Specific Decription1 ---------

Specific Decription2 ---------

Specific Decription3 ---------

Specific Decription4 ---------

The -------- notation meant that by double clicking with the mouse on the prompt a list of values could be entered. For each attribute many values could be entered and these represented OR statements and the logical link between the separate attributes was AND. Hence the relationships could be considered as the following :

General Description	:CORE AND COILS ASS*
	(OR)
	:CORE CLAMPS*
	(&)
Specific Decription1	:1000KVA
	(&)
Specific Decription2	:HV 3300V
	(&)
Specific Decription3	:LV 1130V
	(&)
Specific Decription4	:STRIP LV

This provided a simple and flexible method of searching for part records within the database. A typical output from a part search is shown in figure 17 and was designed to output to the printer and the screen.

5.6 Copying Drawing Files and Records

We decided on four variants of copying data :

- Copy Record & Drawing to New Number
- Copy Record to New Number
- Copy Last Record Opened
- Copy Record

<< ENGINEERING DATA SEARCH REPORT >> Search Criteria :
C

DRG No	GENERAL DESC	Desc 1	Desc 2	Desc 3	Desc 4
611702700	CORE STAMPINGS	RTYY	.	.	.
612703200	CORE CLAMPS	20MVA	1M2	TWO HANDL*	.
615704000	CABLE BOX ASSEMBLY	GRYHTRDYT	.	.	.
615704300	CORE AND CLAMPS AS*	SEFR	.	.	.
617702700	CORE AND COILS ASS*	WER	.	.	.
617702800	CORE AND COILS ASS*	1000KVA	HV 3300V	LV 1130V	STRIP LV
617702900	CORE AND COILS ASS*	erte	.	.	.
632702400	TANK COVER FABRICA*	20MVA	1M2	TWO HANDL*	.
648705500	ADAPTOR AND CABLE *	10MVA	100mm	.	.
691070250	FLAMEPROOF - CERTI*	fdgfh	.	.	.

Figure 17 Part Search Report

It was important to determine the needs of the users as to deny particular copying facilities would encourage an increase in duplicated engineering data and a reduction in standardisation.

One of the limitations of DOMA was the inability to prompt for greater than 1 variable and have IF.... THEN.... ENDIF statements. The need for these command sets was that before copying a drawing it was imperative that it existed and if not the system indicated accordingly. As such features where not available through DOMA, a procedure, called from a DOMA script, was designed which utilised the Graphical User Interface (GUI) Sunview in a C Shell script. The script was designed to open a small window in the middle of the DOMA window and prompt the user for the drawing number to be copied. After entering the number the archive directory was searched for the drawing file and if not found the user was given the choice of retrying or quitting. If the drawing number was found then a new drawing was generated for that generic part type and the drawing and associate part record copied to the new number and placed in the users pool. In addition to copying the new record the DOMA script removed all transaction and revision details as the history was only relevant to the original data.

Copying the record only did not require a search of the archive drawing directory and the DOMA script prompted the user for the drawing number. In this instance it was within DOMA's capabilities to report if the record had not been found. Again once the record had been copied all historical transaction and revision data was deleted.

The remaining two copy procedures were designed to allow the user to retrospectively enter part records and to add records which were references of drawing numbers. An example of this was when a drawing sheet contained more than one part drawing the master record e.g.. 691234500 would contain master data and separate records would be created for the individual part drawings and given the same number with a reference e.g.. 691234501. Copy Last was used when the references were very similar and therefore the part records contained almost identical information.

The above procedures further enhanced data integration and standardisation which were included in the EDM systems objectives.

5.7 Incorporating Existing Engineering Data

Unless the EDM system is to be implemented within a new business it is likely that a facility to encompass engineering data retrospectively is required. This was the case within Brush Transformers Ltd who had thousands of drawings in the archive which needed to be managed by the EDM system. Data contained within the drawings was inaccessible and this meant that it could not be extracted and placed into the database. In the long term this was good as all data entered into the system would still conform to the EDM rules and maintain consistency.

The procedures were relatively simple in their design and involved the user selecting a menu option from DOMA which called a C Shell script. This opened a Sunview

window which prompted the user to enter the drawing number (7 Digits). Again the script searched through the archive and if it was located a new record was generated with the old drawing number plus 2 zeros e.g.. 6912345 ----> 691234500. This was displayed to the user for the user to enter the general description, specific descriptions, originator and approval information. As this information formed the core of the record and was restricted to checkers they only had access to this procedure and would be responsible for the consistency and integrity of data. Once all the information was input the record was registered and through file management the drawing file name was changed to match the key attribute value e.g.. 6912345.DR ----> 691234500.DR. During this process the drawing remained within the archive as the data had already been approved. Additionally, the main reason for converting the data was to modify the drawing and by keeping the data within the archive the first modification transaction was recorded by the system.

Apart from the drawing information all other data had to be input into the record e.g.. parts lists and revision information so we decided that conversion should only occur when the drawing or parts list was required.

5.8 Hardware Requirements

During system build tests were performed on the hardware to determine the feasibility of all the procedures and to ensure that hardware requirements were understood. As the software was to operate within a multi-user environment it was essential to

represent this in the tests. Five workstations were placed in a row and with the help of five draughtsmen we proceeded to test each feature. As expected the slowest operations were those which required write access to the database and the Diad 2d procedures which utilised DOMA reports (DOMA watch mode). The only way to resolve these problems was to increase the response of the client and host workstations which meant that the network, client and host computer specifications were reviewed.

The speed of the network was a product of the client and host computer hardware and depended on the amount of data communicated across it. There were 40 workstations, each with 8MBytes of RAM, from which 10 were dataless clients and 30 were discless. Two hosts (servers) contained 8MBytes of RAM each and 1200MBytes of fixed disc and these supplied all workstations with DOMA software, working directories, archive directories. In addition, the 30 discless clients received the operating system and Diad 2d software. Overall this meant that even before DOMA was fully operational the network was already over burdened and resulted in poor response borne out with our experiments.

The solution was to upgrade each client workstation so that it reduced the level of swapping data in and out of memory, onto disc and across the network. Each was upgraded to 16MBytes of RAM and 210MByte fixed discs which were to contain the operating system (SUN OS) and Diad 2d software. All other data and software was supplied by the file servers. One of them was upgraded to 32MBytes of RAM and this

was selected to run DOMA in watch mode for producing drawing borders and part lists.

A further burden on the system was the amount of disc memory consumed by the DOMA database. However as we had deleted all the software from the discless clients an additional 600MBytes had become available which would last for two years.

6.0 The Implementation Plan for an Engineering Data Management System

Before implementing an EDM system the needs of the organisation must be reviewed. At this point it may be worth performing system re-engineering and a good strategy for the phasing of any implementation is essential. EDM is like any other computer based solution in that if a company is badly managed "bolting on" a system is unlikely to deliver the expected benefits. However, even a well organised design/development organisation can gain the benefits from the process of selecting and implementing EDM.

According to PA Consultant Rob Fisher there is no universal implementation strategy but a number of essential ingredients :

- ensure the highest user involvement in defining what to implement
- document user requirements and employ these to define Functional Specifications for each phase of the implementation
- plan for an incremental approach tackling problems one at a time or in small manageable chunks
- clearly define the scope of the implementation and back this up with a good awareness programme
- resource the project with good people led by a well respected team leader/project manager.

PA consultants have encountered many EDM implementations since 1986 and provides a good set of pitfalls :

- EDM is a set of tools which can be usefully employed to help good companies get better. The motivation should be driven by the people who will benefit, that is to say, "Engineering", "Development" or "Production". It must therefore be

owned by the likes of the Engineering Directors. EDM will always be hard work if it is started as an IT initiative with some vague motive about it being a good thing. This leads to grandiose plans with over ambitious targets to change the way the business operates. Users are rightly sceptical to this approach and resistive if it is not handled correctly.

- Too big too soon is the easiest trap to fall into for any computer system. The scope gets bigger as more possibilities are included within the implementation plan. To avoid this there needs to be two views of the way forward: the long term vision and the short term plan focused on the most important business issues. The temptation to broaden the scope must be controlled.

- Once the system is installed the users will have a much better perception of what is required next and are often enthusiastic to help in the specification stage. This enthusiasm should be harnessed with frequent updates which respond to users needs. Locking the EDM programme away in some IT ivory tower is a recipe for the mainframe mentality which has made so many systems hated by users.

- Because EDM is unlike systems that existing staff will be used to, it is easy for obvious misconceptions to set in.

- As a set of tools EDM systems are generally powerful. That power does however need focus and it is relatively easy to take the passive route of supporting existing working practices and missing a golden opportunity to improve the way things are done. It is worth taking the time to sort out problems before starting to implement the EDM system. This may well mean a certain amount of analysis of the current situation.

- EDM is a long term strategy and commitment. This means that it will require an adequate level of funding and staff support over a number of years. This must be realised and if the commitment is not there at the start the project should not be started.

- Any systems vendor is likely to push for the solution which is easiest for them to implement. It is highly unlikely that this is what the business needs or wants. During the specification phase you need to be aware both of what you want and what are the real boundaries of what you can expect.[10]

At Brush Transformers we used JAD to address many of the areas explained above and once we had developed the initial version of the system we proceeded as is detailed below. When developing the EDM system periodically demonstrating it helped us involve the users and gain essential feedback.

6.1 Pilot Project

The first stage in the system implementation at Brush Transformers was that of ensuring that the software was free from errors and user friendly. Through setting up a small number of pilot projects within the working environment this was achieved. Prior to running the project all relevant staff were trained and their colleagues informed about what they were doing. Management were informed that the output of those involved would reduce while they were learning how to operate the new system. Once the training was complete then the users were given access to the software and provided with on-line supervision. At Brush Transformers Ltd we ran the pilot project for 1 month which we considered a reasonable amount of time. The results were favourable even though changes were made to the software as users made suggestions. These were mainly concerned with document presentation and the organisation of data entry screens.

6.2 Staff Training and Phased System Implementation

Once the pilot project has been completed and alterations made to the system all the staff were trained. At Brush Transformers Ltd we chose to do this in groups of four over 2 days. We decided to use a hands on method for training and set up 4 workstations in a training room and work through realistic examples. With large numbers of staff and limited training and on-line supervision resources training was staggered. Eight draughtsmen were to be trained and then the system implemented

within their section. Once this had settled down the next eight would be trained and the system implemented within their department.

Even though the staff were trained and the system implemented we decided to phase in the EDM system by selecting a variety of contracts which we considered would exemplify the need for any enhancements. In fact this was extremely useful and provided many new ideas and suggestions which were incorporated into the system.[11]

6.3 Full system review

Once the system was fully implemented and had settled down a full system review was planned for. This would include the development of full system documentation and the improvement of procedures.

6.4 Continuous Enhancement

With the system fully implemented for some time continuous reviews should be performed. The objective being to improve the EDM system and incorporate new ideas.

Conclusion & Recommendations

The research resulted in the development of an Engineering Data Management system which utilised SUN workstations and DOMA, Diad 2d CAD and Unix C Shell software. In accordance with our estimations the project fell within the predicted cost and time scale.

Through the development of this system a number of issues were highlighted. Our start point had been with a number of tools from which a system was built. This certainly enabled a custom system to be designed which suited the companies needs exactly. However unless considerable manpower can be designated to the software development it can take a considerable amount of time. Once the system has been built it requires full documentation and personnel will require training on how to modify the system to suite future needs. Without this flexibility is relinquished.

I consider that DOMA was well pitched and gave a good level of functionality but lacked a good querying system which would have made software development simple. When considering multi-user systems we learnt that the method of database security is imperative to the systems success. I considered the Tide database was inadequate as when one person was entering data it locked the whole database. In practice the delay was limited but it did provide a potential future problem.

Another area which was inadequate was the inability to access drawings unless they were open and on the screen. This meant that documentation still required signing when initially approved. Even though it was reasonable to do this in the future paper copies will become less necessary as better documentation viewing products become available.

Whilst developing the system the potential to include and link up to other systems became obvious. With "One of a Kind" manufacture MRP is difficult to use due to the constantly changing Bill of Materials (BOMs). However, this could be overcome by producing reports in ASCI format within the EDM system which could be input into the MRP system, saving entry time and increasing integration and control.

Overall EDM is a philosophy and not just a piece of software and the plan should not just include drawing offices as selected for this research. In fact a 5 year plan should be developed with the goal being integration, simplification and automation of data and processes

Since developing this EDM system many good proprietary systems have entered the market place. Even though this does not remove the need for detailed analysis and implementation they do provide a reasonable solution and at a far reduced cost. Therefore for those selecting and implementing an EDM system the in house development and design should only be tackled after carefully reviewing available software. The advantage being a shorter implementation time and reduced costs.

Throughout the project careful leadership and a good relationship was imperative to the projects success. If we had not had these then the project would have failed and the chance to improve lost.

Bibliography

1 "Managing Your Engineering Information Factory in the 1990s Using Engineering Systems" : Peter Gould, BPICS/Control April/May 1992

2 "Integration Audit" : JL Burbidge, P Falster & JO Riis, Computer Integrated Manufacturing Systems Vol 2 No 3 August 1989

3 "Engineering Data Management" Ken McIntosh, Engineering April 1992

4 "Visably Managing the Data" ICL, Engineering February 1993

5 "Shop Floor Quality Data Capture" SS Coates, Teaching Company Quality Seminar Sept 1992

6 "Systems Analysis and Design Methods" : Whitten, Bentley and Barlow, pub IRWIN 1989

7 "Product Information Management" Danny Rogers, Industrial Management & Data Systems Vol 94 No 1 1994 MCB University Press Ltd

8 "Investment in manufacturing technology" PL Primrose Chapman and Hall, 1993

9 "Advanced Manufacturing Technology - the effect on profits" Bromwich M, Institute of Production Engineers Seminar 23 October 1986

10 "Concurrent engineering Project Management of Supporting Systems" Rob Fisher, PA Consultancy 1993

11 "Planning a Tailored Incremental Implementation of PIM" JL Roche, EDS 1993

12 "Out of the Paper Jungle" Manufacturing Systems May 1992

13 "Managing Manufacturing History" P Noaker Manufacturing Engineering Nov 1993

14 "Technology for World Class Engineering" R Mills, B Beckert, L Kempfer, J Chalsma, CAD/CAM Planning 1992 (Supplement to Penton Publications)

15 "Product Engineering Manufacturing and Supporting Infrastructure for an Enterprise" SN Sheth, EDS Technical Systems Development 1992

16 "PIM Systems Manage the Information Morass" SB Hunter, Machine Design May 28 1993

17 "Managing Technical Information in an Open Systems Environment" JC More, Industrial Management & Data Systems Vol 92 No 1 1992 MCB University Press Ltd

18 "Engineering Data Management: Achieving Integration through Database Technology" SD Urban, JJ Shah & MT Rogers, Computing & Control Journal June 1993

19 "The Practical Integration of Manufacturing Applications" JS Busby & D Hutchinson, Software Practice and Experience Vol 22(2) Feb 1992

20 "Product Data Management" M Puttre, Mechanical Engineering Oct 1991

21 "Designs on a Global Market" L Costanzo, Engineering Oct 1993

22 "Mastering the Information Explosion" T Shelley, Eureka April 1993

23 "The Data Management Fast Track" K Kosh & DR Smiedt, Bobbin Aug 1993

24 "Data Direkt!" K McIntosh, Manufacturing Engineer Feb 1992

25 "Global Co-operation is No Just Talk" D Palframan, Engineering Computers April 1992

26 "Crossing the Data Divide" T Gold, Computerised Manufacturing Sept 1992

27 "Take a Step Inside" S Brett, CADCAM April 1994

28 "Technology Holds Key to Integration" T Shelley, Eureka April 1994

29 "Engineering Data Management in the Automotive Industry" NB Anthony & RA Whale, C389/104 IMechE 1992

30 "The Quest for Database Unity" DR Smiedt, Bobbin Oct 1993

31 "Product Data Management: Applications in Production Instructions and Batch Records" B Meserve, Hybritech Incorporated 1993

32 "Using Prevention Techniques" JL Boteler, Quality Progress July 1993

33 "Follow the Rules and Ease the Pain" A Gregory, Engineering Computers September 1992

34 "Manage Your Data" P Gould, Computerised Manufacturing April /May 1992

Appendix

Example C Shell Scripts

```
#  /bin/csh -fx
#          BTL - DOMA File Management Example Script
#          ========================================
# 25/9/89 rs sunos 4.0
#
#  Description:  It reads file management functions and moves
#                files from one pool/directory to another according to the
#                function.
#
#  Database   :  /axe1/DOMA/FLP
#
#  Arguments  :  [1] function   -  File management function
#                                  CREATE
#                                  APPROVE
#                                  ADD
#                                  SUBTRACT
#                                  MOVE
#                                  DELETE
#                                  anything else
#
#                [2] parameter  -  filename of parameter file
#
#  ------------------------------------------------------------------
#
# Initialise variables
   set parameter = $argv[2]
# cat $parameter
   set function = $argv[1]
# set template file variables
   set fmdir = /axe1/DOMA/scripts        #  <-- site dependent
   set fmtemp = DUMMY.DR                 #  <-- site dependent
   if (-e $HOME/domamessage) \rm $HOME/domamessage
#

# The attributes below match the 'FLP DATABASE' which is supplied with
# the product. In order to create your file management system
# please consult your local support office for details.

#      Filename(s)     - 69
#      Pool            - 62
#      Pool directory - 61
   set curratt69
   set curratt62
   set curratt61
   set prevatt69
   set prevatt62
   set prevatt61

# create a command file from the parameter file
#   first a set of commands to initialise the variables
   sed -e 's/^Current Attribute \([0-9]*\).*/set curratt\1/' \
       -e 's/^Previous Attribute \([0-9]*\).*/set prevatt\1/' \
       <$parameter \
   | sort -u >C_$parameter

#   then generate the commands to set the variables, eg
#     Current Attribute N <att name> :=att_value     to
#     set currattN = ( $attN att_value )


  cat $parameter | \
  sed -e 's/^Current Attribute /curratt/'  \
      -e 's/^Previous Attribute /prevatt/'  \
      -e 's/ <.*> :=/:g "/' \
  | awk '{print "set",$1,"="$2 $3 $4 $5 $6 $7 $8"\""}'  \
  | sed 's/:g //g' >> C_$parameter
```

```
#
#    now set the variable from the command file
     source C_$parameter
#cat  C_$parameter
#    rm C_$parameter

#    give meaningful names to the parameters that we wish to use
     if ( $#prevatt69 > 0 ) then
        set prevfiles = ( $prevatt69:q )
     else
        set prevfiles = ""
     endif
     if ( $#curratt69 > 0 ) then
        set currfiles = ( $curratt69:q )
     else
        echo "======= No Current Files ======="
        exit 1
     endif

     if ( $#prevatt62 > 0 ) then
        set prevpool = $prevatt62[1]
     else
        set prevpool = ""
     endif
     if ( $#curratt62 > 0 ) then
        set currpool = $curratt62[1]
     else
        echo "======= No Current Pool ======="
        exit 1
     endif

     if ( $#prevatt61 > 0 ) then
        set prevdir = $prevatt61[1]
     else
        set prevdir = ""
     endif
     if ( $#curratt61 > 0 ) then
        set currdir = $curratt61[1]
     else
        echo "======= No Current Pool Directory ======="
        exit 1
     endif
#    All parameters are now set
#    Do the job
     set stat = 0
     echo ============ `date '+%a %h %d, 19%y at %T'` ============
     switch ( $function )
     case "CREATE":
# copy template file to location
          foreach f ( $currfiles )
               set topath = $currdir/$f
               echo Allocating $f to $currdir > $HOME/domamessage
               cp $fmdir/$fmtemp $topath
               sed -e "s/nnnn/$curratt61/g" < /axel/DOMA/scripts/SIZE.SAVE > SIZE
 end
 breaksw
   case "COPY"
      echo 'dummy copy function $1 '
   end
   breaksw
case "APPROVE":
#
#   BTL added function - which archives drawing files ensuring that they are
#   safe from duplication or uncontrolled modification.
```

```
#  /axe1/DOMA/script/domaarchive script modified from DRput
#
         set check = 1
         foreach f ( $currfiles )
             set frompath = $prevdir/$f
             echo Checking request to move $f from $prevpool to $currpool
             if ! (-r $frompath ) then
                 echo Cant find file $frompath > $HOME/domamessage
                 set check = 2
                 exit 1
             endif
             set topath = $currdir/$f
             if (-e $topath ) then
                 echo  file $topath already exists > $HOME/domamessage
                 set check = 2
                 exit 1
             endif
         end
         foreach f ( $currfiles )
             set topath = $currdir/$f
             set frompath = $prevdir/$f
             echo Moving $frompath to $topath
               csh -fx /axe1/DOMA/scripts/domaarchive $f $currdir $prevdir
         end
        if ( $check == 1 ) then
              touch domamessage
              echo " " >> $HOME/domamessage
              echo "Record Successfully Archived"  >> $HOME/domamessage
        endif

         breaksw
case "MOVE":
         set check = 1
         foreach f ( $currfiles )
             set frompath = $prevdir/$f
             echo Checking request to move $f from $prevpool to $currpool
             if ! (-r $frompath ) then
                 set check = 2
                 echo Cant find file $frompath > $HOME/domamessage
                 exit 1
             endif
             set topath = $currdir/$f
             if (-e $topath ) then
                 echo  file $topath already exists > $HOME/domamessage
                set check = 2
                 exit 1
             endif
         end
         foreach f ( $currfiles )
             set topath = $currdir/$f
             set frompath = $prevdir/$f
             mv $frompath $topath
             echo Record/Drawing moved from $frompath to $topath > $HOME/domame:
         end
        if ( $check == 1 ) then
              touch domamessage
              echo " " >> $HOME/domamessage
              echo "Record Successfully Moved"
        endif
         breaksw
case "MOD":
#
#  BTL added function - enables drawings to be modified in a secure environment
#  calls script /axe1/DOMA/scripts/domadrmod
#
        set check = 1
```

```
     foreach f ( $currfiles )
         set frompath = $prevdir/$f
         echo Checking request to move $f from $prevpool to $currpool
         if ! (-r $frompath.Z ) then
            set check = 2
             echo Cant find file $frompath > $HOME/domamessage
             exit 1
         endif
         set topath = $currdir/$f
         if (-e $topath.Z ) then
            set check = 2
             echo  file $topath already exists > $HOME/domamessage
             exit 1
         endif
         if (-e $topath ) then
             echo  file $topath already exists > $HOME/domamessage
            set check = 2
             exit 1
         endif
     end
     foreach f ( $currfiles )
         set topath = $currdir/$f
         set frompath = $prevdir/$f
         echo Moving $frompath to $topath
         csh /axe1/DOMA/scripts/domadrmod $f $prevdir $currdir
     end
    if ( $check == 1 ) then
          touch domamessage
          echo " " >> $HOME/domamessage
          echo "Record Successfully Retrieved From Archive" >> $HOME/domame:
          echo " "
          echo "Now stored within your work pool"  >> $HOME/domamessage
    endif
     breaksw
case "REVUPDATE"
      sed -e "s/xxxx/$prevatt71/" /axe1/DOMA/scripts/LATEST.SAVE >! ~/LATES'
       breaksw
case "DELETE":
     foreach f ( $currfiles )
         set topath = $prevdir/$f
         echo Checking request to delete $f from $currpool
         if ! (-r $topath ) then
             echo Cant find file $topath
             exit 1
         endif
     end
     foreach f ( $currfiles )
         set topath = $prevdir/$f
         echo Removing $topath
         rm $topath
     end
     breaksw
case "SUBTRACT":
     foreach f ( $currfiles )
         set topath = $currdir/$f
         echo Checking request to delete $f from $currpool
         if ! (-r $topath ) then
             echo Cant find file $topath > $HOME/domamessage
             exit 1
         endif
     end
     foreach f ( $currfiles )
         set topath = $currdir/$f
         echo Removing $topath
         rm $topath
     end
```

```
     breaksw
default:
     echo =========  ILLEGAL FUNCTION $function  ========= > $HOME/domamess
     exit 1
endsw
exit
```

```
# /axe1/DOMA/scripts/domaarchive ver 1.0 10/04/92
#
#  Called from domafm_ex.csh - DOMA file management script as part of the
#  APPROVE procedure.
#
#  Ensures that drawing files are archived in the correct area, and modified
#  only in a controled environment.  Checks that file does not already exist
#
#
        set fromdir = $3
        set todir = $2
        set filename = $1
         if (-e $HOME/domamessage ) then
                rm $HOME/domamessage
                 touch $HOME/domamessage
         endif

          if  !(-e $fromdir/$filename) then
            echo  ""
            echo  "$filename : INCORRECT FILE NAME (FILE DOES NOT EXIST)" >> $H
            set filename = " "
            echo ""
            goto FAIL
          endif
          if (-e $todir/$filename.Z) then
            echo "CANNOT ARCHIVE $filename  - FILE ALREADY EXISTS" >> $HOME/do
            goto FAIL
          else
            cd $fromdir
            chmod ugo-w $filename
            chmod o+r $filename
            compress $filename
            rsh axe /bin/syscopy $fromdir $filename.Z $todir $filename.Z
            if (-e $todir/$filename.Z) \
            rm -f $filename.Z
            echo "Drg $filename archived from $3 on 'date +%D-%H:%M'"\
              >> $todir/archivelog
               echo "Drawing file & record $1 have been archived successfully"
               echo " " >> $HOME/domamessage
               echo "! Request Complete ! " >> $HOME/domamessage
            tail -1 $todir/archivelog
            echo    ""
          endif
       cd
      exit
#
FAIL:
echo ""
echo ""
echo "                DRput - JOB HAS BEEN ABANDONED"
echo "                ********************************"
echo ""
echo ""
      exit 1
```

```
#
#    @(#)/axe1/DOMA/script/doma_drmod - Retrieve drawing file from library
#    Called from /axe1/DOMA/scripts/OLD
#
#    Set up flag file to indicate that macro is running
#    If macro is already in use,loop until previous user has finished
#
    alias rm rm -f
#
#    Handle interrupts
#
    onintr ABORT
loop:
    if !(-e ../DRmodflag) then
       touch ../DRmodflag
    else
       goto loop
    endif
#
    echo " "
#
#    Set drawing library depending on who is running macro
#    (or rather where macro is being run from)
#
    set section = `pwd | awk -F/ '{print $4}' `
    if ($section == 'con') then
        set lib = '/DRG/CON'
    else if ($section == 'ctrl') then
        set lib = '/DRG/CTRL'
    else if ($section == 'min') then
        set lib = '/DRG/MIN'
    else if ($section == 'dpg') then
        set lib = '/DRG/DPG'
    else
        echo '*** ERROR - PLEASE MOVE TO YOUR WORKING DIRECTORY ***'
        goto FAIL
    endif
#
    set homedir = `pwd`
#
label20:
    set newfileno = $1
    echo ""
    case y:
           if (-e $homedir/$fileno.DR)then
              echo "ERROR - DRAWING FILE ALREADY EXISTS IN YOUR DIRECTORY"
              goto FAIL
           endif
           if !(-e $lib/$fileno.DR.Z) then
             if !(-e $lib/$fileno.DR.old.Z) then
               echo "***  ERROR - FILE DOES NOT EXIST  ***"
             else
               echo -n "***  ERROR - FILE IS ALREADY BEING MODIFIED BY "
               echo -n `cat $lib/archivelog | grep $fileno | tail -1 | awk -F/
               echo " ***"
             endif
             goto FAIL
           else
             rsh axe /bin/sysmove $lib $fileno.DR.Z $lib $fileno.DR.old.Z
             rsh axe /bin/syscopy $lib $fileno.DR.old.Z $homedir $fileno.DR.Z
             if !(-e $homedir/$fileno.DR.Z) then
               echo "***  ERROR - DRAWING NOT RETRIEVED FROM ARCHIVE  ***"
               rsh axe /bin/sysmove $lib $fileno.DR.old.Z $lib $fileno.DR.Z
               goto FAIL
             else
               uncompress $homedir $fileno.DR
```

```
            endif
          endif
          breaksw
    case Y:
          if (-e $homedir/$fileno.DR)then
             echo "ERROR - DRAWING FILE ALREADY EXISTS IN YOUR DIRECTORY"
             goto FAIL
          endif
          if !(-e $lib/$fileno.DR.Z) then
            if !(-e $lib/$fileno.DR.old.Z) then
              echo "***  ERROR - FILE DOES NOT EXIST  ***"
            else
              echo -n "***  ERROR - FILE IS ALREADY BEING MODIFIED BY "
              echo -n `cat $lib/archivelog | grep $fileno | tail -1 | awk -F/
              echo " ***"
            endif
            goto FAIL
          else
            rsh axe /bin/sysmove $lib $fileno.DR.Z $lib $fileno.DR.old.Z
            rsh axe /bin/syscopy $lib $fileno.DR.old.Z $homedir $fileno00.DR.Z
            if !(-e $homedir/$fileno.DR.Z) then
              echo "***  ERROR - DRAWING NOT RETRIEVED FROM ARCHIVE  ***"
              rsh axe /usr/bin/sysmove $lib $fileno.DR.old.Z $lib $fileno.DR.Z
              goto FAIL
            else
              uncompress $homedir/$fileno.DR
            endif
          endif
          breaksw
    case r:
          goto label20
          breaksw
    case R:
          goto label20
          breaksw
    default:
          goto FAIL
    endsw
    echo "Drawing $fileno.DR RETREIVED from $lib to $homedir on `date +%D-%H:%M
          >> $lib/archivelog
    echo " THE MASTER COPY OF $fileno IS IN $homedir"
#
RUNOK:
    echo ""
    echo ""
    echo ""
    echo "                          DRmod - JOB RUN O.K."
    echo "                          ********************"
    echo ""
    echo ""
    echo ""
    rm ../DRmodflag
    alias rm rm -i
    exit
#
ABORT:
#
#   Clear up home directory and return archive to initial state
#
    if !($?lib && $?fileno) then
      goto FAIL
    else
      set dummy = `ls -1 $lib`
      if (!(-e $lib/$fileno.DR.Z) && (-e $lib/$fileno.DR.old.Z)) then
        rsh axe /usr/bin/sysmove $lib $fileno.DR.old.Z $lib $fileno.DR.Z
        if (`ls -1t | head -1` =~ $fileno.*) then
```

```
          rm $fileno.*
        endif
      endif
    endif
#
FAIL:
    echo ""
    echo ""
    echo "               DRmod - JOB HAS BEEN ABANDONED"
    echo "               ******************************"
    echo ""
    echo ""
    rm ../DRmodflag
    alias rm rm -i
    exit
```

```
#
#    /axe1/DOMA/scripts/domadrmod
#
#    Script called from DOMA file management script domafm_ex.csh  CASE MOD.
#
#    Enables drawing file to be modified in a safe environment ie, file becomes
#    until change has been made and file approved by checker.
#
#    Set up flag file to indicate that macro is running
#    If macro is already in use,loop until previous user has finished
#
     alias rm rm -f
#
#    Handle interrupts
#
     onintr ABORT
loop:
     if !(-e ../DRmodflag) then
        touch ../DRmodflag
     else
        goto loop
     endif
#
     echo "domadrmod version 1.0  13/04/92"
     echo " "
#
#    Set drawing library depending on who is running macro
#    (or rather where macro is being run from)
#
     cd
     set lib = $2
     echo $1
     echo $2
#
     set homedir = $3
     echo $3
#
#
     set fileno=$1
     set newfileno = $fileno
#
            if (-e $homedir/$fileno)then
               if (-e $HOME/domamessage) rm $HOME/domamessage
               echo "ERROR - DRAWING FILE ALREADY EXISTS IN YOUR DIRECTORY" > $H(
               goto FAIL
            endif
            if !(-e $lib/$fileno.Z) then
              if !(-e $lib/$fileno.old.Z) then
              if (-e $HOME/domamessage) rm $HOME/domamessage
                echo "***  ERROR - FILE $fileno DOES NOT EXIST  ***" > $HOME/dom
              else
              if (-e $HOME/domamessage) rm $HOME/domamessage
                echo -n "***  ERROR - FILE $fileno IS ALREADY BEING MODIFIED BY '
                echo -n `cat $lib/archivelog | grep $fileno | tail -1 | awk -F/
              endif
              goto FAIL
            else
              rsh axe /bin/sysmove $lib $fileno.Z $lib $fileno.old.Z
              rsh axe /bin/syscopy $lib $fileno.old.Z $homedir $fileno.Z
              if !(-e $homedir/$fileno.Z) then
              if (-e $HOME/domamessage) rm $HOME/domamessage
                echo "***  ERROR - DRAWING  $fileno NOT RETRIEVED FROM ARCHIVE
                rsh axe /bin/sysmove $lib $fileno.old.Z $lib $fileno.Z
                goto FAIL
              else
                uncompress $homedir/$fileno
```

```
            endif
          endif
          breaksw
        default:
          goto FAIL
    endsw
    echo "Drawing $fileno RETREIVED from $lib to $homedir on `date +%D-%H:%M`"\
          >> $lib/archivelog
           if (-e $HOME/domamessage) rm $HOME/domamessage
           echo " THE MASTER COPY OF $fileno IS IN $homedir" > $HOME/domamessa
#
RUNOK:
    echo ""
    echo ""
    echo ""
    echo "                          DRmod - JOB RUN O.K."
    echo "                          ********************"
    echo ""
    echo ""
    echo ""
    rm ../DRmodflag
    alias rm rm -i
    chmod 644 $homedir/$fileno
    exit
#
ABORT:
#
#   Clear up home directory and return archive to initial state
#
    if !($?lib && $?fileno) then
      goto FAIL
    else
      set dummy = `ls -l $lib`
      if (!(-e $lib/$fileno.Z) && (-e $lib/$fileno.old.Z)) then
        rsh axe /usr/bin/sysmove $lib $fileno.old.Z $lib $fileno.Z
        if (`ls -lt | head -1` =~ $fileno.*) then
          rm $fileno.*
        endif
      endif
    endif
#
FAIL:
    echo ""
    echo ""
    echo "               DRmod - JOB HAS BEEN ABANDONED"
    echo "               ******************************"
    echo ""
    echo ""
    rm ../DRmodflag
    alias rm rm -i
    exit 1
```

```
# /bin/csh
#
setenv DOMAADMINDIR /axe1/DOMA/work
#  Script called from Diad macros DOMA-NEWBORDER.DI and DOMA-MODXXXX.DI
#  Edits the Diad variable $DRG value into the two DOMA scripts MODREP
#  and BORDEREP.
#
#  Starts DOMA logging the user in as batch username and enters DOMA
#  in batch mode - batch mode initiated by DACS20_ID.SF.
#
#
touch temp3.DOMA1 temp3.DOMA2
\rm -f $HOME/*.DOMA1 $HOME/*.DOMA2
sed -e "s/xxxx/$1/" /axe1/DOMA/batch/MODREP.SAVE  >! $HOME/MODREP
sed -e "s/xxxx/$1/" /axe1/DOMA/batch/BORDERREP.SAVE  >! $HOME/BORDERREP
echo SETENV HOME $HOME  >> $DOMAADMINDIR/domawatch
echo \$M $HOME/MODREP $HOME >> $DOMAADMINDIR/domawatch
echo \$M $HOME/BORDERREP $HOME >> $DOMAADMINDIR/domawatch
loop:
if ( ! (-e update.DOMA2  )) then
goto loop
endif

exit
# old stuff
mv ~/DACS20.IN ~/DACS20.IN.SAVE
\cp ~/DACS20.BM ~/DACS20.IN
\cp  /axe1/DOMA/batch/BATCH.SF  ~/DACS20_ID.SF
/axe1/DOMA/scripts/start.csh
\rm ~/DACS20.IN
mv ~/DACS20.IN.SAVE ~/DACS20.IN
\rm -f ~/*.DOMA1
exit
```

```
# /axe1/DOMA/batch/BORDEROUT
#
#  Macro file called from DOMA report B0RDER, device BORDEROUT, which splits
#  the report output file BORDERDATA into its individual fields and places
#  data within files with the | character denoting the start and finish of the
#  field.  eg.  file ~/DRGNO would resemble |  6040567900 |
grep -v '^$' $HOME/BORDERDATA > $HOME/temp.DOMA1
tail -1 $HOME/temp.DOMA1 > $HOME/sp9.DOMA1
head -8 $HOME/temp.DOMA1 > $HOME/tem.DOMA1
tail -1 $HOME/tem.DOMA1 > $HOME/sp4.DOMA1
head -7 $HOME/temp.DOMA1 > $HOME/temp1.DOMA1
tail -1 $HOME/temp1.DOMA1 > $HOME/sp3.DOMA1
head -6 $HOME/temp.DOMA1 > $HOME/temp2.DOMA1
tail -1 $HOME/temp2.DOMA1 > $HOME/sp2.DOMA1
head -5 $HOME/temp.DOMA1 > $HOME/temp3.DOMA1
tail -1 $HOME/temp3.DOMA1 > $HOME/sp1.DOMA1
head -4 $HOME/temp.DOMA1 > $HOME/temp4.DOMA1
tail -1 $HOME/temp4.DOMA1 > $HOME/sp8.DOMA1
head -3 $HOME/temp.DOMA1 > $HOME/temp5.DOMA1
tail -1 $HOME/temp5.DOMA1 > $HOME/sp5.DOMA1
head -2 $HOME/temp.DOMA1 > $HOME/temp6.DOMA1
tail -1 $HOME/temp6.DOMA1 > $HOME/sp6.DOMA1
head -1 $HOME/temp.DOMA1 > $HOME/sp7.DOMA1
nawk -f /axe1/DOMA/batch/addpipe7 $HOME/sp7.DOMA1 > $HOME/DRGNO.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/sp6.DOMA1 > $HOME/NAM.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/sp5.DOMA1 > $HOME/DT.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/sp8.DOMA1 > $HOME/GD.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/sp9.DOMA1 > $HOME/PG.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/sp1.DOMA1 > $HOME/desc1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/sp2.DOMA1 > $HOME/desc2.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/sp3.DOMA1 > $HOME/desc3.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/sp4.DOMA1 > $HOME/desc4.DOMA2
touch $HOME/update.DOMA2 temp1.DOMA1
\rm -f $HOME/BORDERDATA $HOME/*.DOMA1
```

```
# /axe1/DOMA/batch/DOMADEL
#
#  Called from Diad Macro scripts and used to delete *.DOMA2 files
#
#
#
\rm *.DOMA2  *.DOMA1 SHT* noofrows* noofcols*
if (-e LOPDATA) \rm LOPDATA
if (-e LOPDRGREP ) \rm LOPDRGREP
if (-e MODREP) \rm MODREP
if (-e BORDERREP) \rm BORDERREP
if ( -e MODDATA) \rm MODDATA
if ( -e LOPSHTREP) \rm LOPSHTREP
if (-e LOPSHTHEADREP ) \rm LOPSHTHEADREP
if ( -e LOPHEADER ) \rm LOPHEADER
exit
```

```
# /bin/csh
# Command file called from DOMA which divides the DOMA report(DRGMODINFO) output
# file MODDATA into its constituent parts ,as files, and place pipes around the
#
#
#
\rm REV1 REV2 NAME1 NAME2 DATE1 DATE2 REASON1 REASON2 DATATEMP
grep -v '^$' MODDATA > DATATEMP
tail -14 DATATEMP > temp
head -1 temp > REVB
tail -4 temp > REASONA
head -10 temp > temp1
tail -1 temp1 > DATEA
head -9 temp > temp2
tail -1 temp2 > NAMEA
head -8 temp > temp3
tail -1 temp3 > REVA
head -7 temp > temp4
tail -4 temp4 > REASONB
head -3 temp > temp5
tail -1 temp5 > DATEB
head -2 temp > temp6
tail -1 temp6 > NAMEB
nawk -f /axe1/DOMA/batch/addpipe15 REVA > REV1
nawk -f /axe1/DOMA/batch/addpipe15 REVB > REV2
nawk -f /axe1/DOMA/batch/addpipe15 NAMEA > NAME1
nawk -f /axe1/DOMA/batch/addpipe15 NAMEB > NAME2
nawk -f /axe1/DOMA/batch/addpipe15 DATEA > DATE1
nawk -f /axe1/DOMA/batch/addpipe15 DATEB > DATE2
nawk -f /axe1/DOMA/batch/addpipe REASONA > REASON1
nawk -f /axe1/DOMA/batch/addpipe REASONB > REASON2
\rm temp* DATATEMP REVA REVB NAMEA NAMEB DATEA DATEB REASONA REASONB
```

```
# /axe1/DOMA/batch/DRGMODOUT
# Command file called from DOMA which divides the DOMA report(DRGMODINFO) outpu
# file MODDATA into its constituent parts ,as files, and place pipes around the
#
#
#
grep -v '^$' $HOME/MODDATA > $HOME/DATATEMP.DOMA1
set chk = `wc $HOME/DATATEMP.DOMA1 | awk '{ printf $1 }'`
if ( $chk > 13 ) then
tail -14 $HOME/DATATEMP.DOMA1 > $HOME/temp.DOMA1
head -1 $HOME/temp.DOMA1 > $HOME/REVB.DOMA1
tail -4 $HOME/temp.DOMA1 > $HOME/REASONA.DOMA1
head -10 $HOME/temp.DOMA1 > $HOME/temp1.DOMA1
tail -1 $HOME/temp1.DOMA1 > $HOME/DATEA.DOMA1
head -9 $HOME/temp.DOMA1 > $HOME/temp2.DOMA1
tail -1 $HOME/temp2.DOMA1 > $HOME/NAMEA.DOMA1
head -8 $HOME/temp.DOMA1 > $HOME/temp3.DOMA1
tail -1 $HOME/temp3.DOMA1 > $HOME/REVA.DOMA1
head -7 $HOME/temp.DOMA1 > $HOME/temp4.DOMA1
tail -4 $HOME/temp4.DOMA1 > $HOME/REASONB.DOMA1
head -3 $HOME/temp.DOMA1 > $HOME/temp5.DOMA1
tail -1 $HOME/temp5.DOMA1 > $HOME/DATEB.DOMA1
head -2 $HOME/temp.DOMA1 > $HOME/temp6.DOMA1
tail -1 $HOME/temp6.DOMA1 > $HOME/NAMEB.DOMA1
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/REVA.DOMA1 > $HOME/REV1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/REVB.DOMA1 > $HOME/REV2.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/NAMEA.DOMA1 > $HOME/NAME1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/NAMEB.DOMA1 > $HOME/NAME2.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/DATEA.DOMA1 > $HOME/DATE1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/DATEB.DOMA1 > $HOME/DATE2.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/REASONA.DOMA1 > $HOME/REASON1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/REASONB.DOMA1 > $HOME/REASON2.DOMA2
endif
if ( $chk == 7 ) then
/axe1/DOMA/scripts/lineget $HOME/DATATEMP.DOMA1 1 > $HOME/REVA.DOMA1
/axe1/DOMA/scripts/lineget $HOME/DATATEMP.DOMA1 2 > $HOME/NAMEA.DOMA1
/axe1/DOMA/scripts/lineget $HOME/DATATEMP.DOMA1 3 > $HOME/DATEA.DOMA1
/axe1/DOMA/scripts/lineget $HOME/DATATEMP.DOMA1 4 > $HOME/REASONA.DOMA1
/axe1/DOMA/scripts/lineget $HOME/DATATEMP.DOMA1 5 >> $HOME/REASONA.DOMA1
/axe1/DOMA/scripts/lineget $HOME/DATATEMP.DOMA1 6 >> $HOME/REASONA.DOMA1
/axe1/DOMA/scripts/lineget $HOME/DATATEMP.DOMA1 7 >> $HOME/REASONA.DOMA1
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/REVA.DOMA1 > $HOME/REV1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/NAMEA.DOMA1 > $HOME/NAME1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe15 $HOME/DATEA.DOMA1 > $HOME/DATE1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/REASONA.DOMA1 > $HOME/REASON1.DOMA2
endif
touch temp2.DOMA1
\rm -f $HOME/*.DOMA1
exit
```

```
# /bin/csh
#
# /axe1/DOMA/batch/LOPHEADOUT
# DOMA macro - which is called from DOMA report/device  LOPHEADER/LOPHEADER
# DESIGNED TO BE EXECUTED IN DOMA BATCH MODE ONLY.
# Data IN LOPHEADER output from DOMAt hen passed through this
# program which splits data in different files and
# adds | character to enable data to be passed into Diad.
#
#
# LIST OF PARTS SHEET DATA
#
cd
grep -v '^$' $HOME/LOPHEADER > $HOME/DATATEMP.DOMA1
head  -1 $HOME/DATATEMP.DOMA1 > $HOME/DRGNO.DOMA1
head  -6 $HOME/DATATEMP.DOMA1 > $HOME/DATATEMP2.DOMA1
tail -1 $HOME/DATATEMP2.DOMA1 > $HOME/CHECK.DOMA1
tail -5 $HOME/DATATEMP2.DOMA1> $HOME/DATATEMP3.DOMA1
head -1 $HOME/DATATEMP3.DOMA1 > $HOME/PRODGRP.DOMA1
tail -4 $HOME/DATATEMP3.DOMA1 > $HOME/DATATEMP4.DOMA1
head -1 $HOME/DATATEMP4.DOMA1 > $HOME/DESC.DOMA1
tail -3 $HOME/DATATEMP4.DOMA1 > $HOME/DATATEMP5.DOMA1
head -1 $HOME/DATATEMP5.DOMA1 > $HOME/ORIG.DOMA1
tail -2 $HOME/DATATEMP5.DOMA1 > $HOME/DATATEMP6.DOMA1
head -1 $HOME/DATATEMP6.DOMA1 > $HOME/DATE.DOMA1
set tot = `wc $HOME/DATATEMP.DOMA1 | awk '{ printf $1 }'`
@ rem = $tot - 6
tail -$rem $HOME/DATATEMP.DOMA1 > $HOME/DATATEMP7.DOMA1
if ( $rem == 6 ) then
head -1 $HOME/DATATEMP7.DOMA1 > $HOME/REV2.DOMA1
tail -1 $HOME/DATATEMP7.DOMA1 > $HOME/REVDATE1.DOMA1
tail -5 $HOME/DATATEMP7.DOMA1 > $HOME/DATATEMP8.DOMA1
head -1 $HOME/DATATEMP8.DOMA1 > $HOME/NAM2.DOMA1
tail -4 $HOME/DATATEMP8.DOMA1 > $HOME/DATATEMP9.DOMA1
head -1 $HOME/DATATEMP9.DOMA1 > $HOME/REVDATE2.DOMA1
tail -3 $HOME/DATATEMP9.DOMA1 > $HOME/DATATEMP10.DOMA1
head -1 $HOME/DATATEMP10.DOMA1 > $HOME/REV1.DOMA1
tail -2 $HOME/DATATEMP10.DOMA1 > $HOME/DATATEMP11.DOMA1
head -1 $HOME/DATATEMP11.DOMA1 > $HOME/NAM1.DOMA1
endif
if ( $rem == 3 ) then
head -1 $HOME/DATATEMP7.DOMA1 > $HOME/REV1.DOMA1
tail -1 $HOME/DATATEMP7.DOMA1 > $HOME/REVDATE1.DOMA1
tail -2 $HOME/DATATEMP7.DOMA1 > $HOME/DATATEMP8.DOMA1
head -1 $HOME/DATATEMP8.DOMA1 > $HOME/NAM1.DOMA1
endif
if ( $rem > 4 ) then
if ( $rem < 5 ) then
echo "!!! Incomplete DOMA record !!!"
endif
endif
if ( $rem > 7 ) then
tail -6 $HOME/DATATEMP.DOMA1 > $HOME/DATATEMP7.DOMA1
head -1 $HOME/DATATEMP7.DOMA1 > $HOME/REV2.DOMA1
tail -1 $HOME/DATATEMP7.DOMA1 > $HOME/REVDATE1.DOMA1
tail -5 $HOME/DATATEMP7.DOMA1 > $HOME/DATATEMP8.DOMA1
head -1 $HOME/DATATEMP8.DOMA1 > $HOME/NAM2.DOMA1
tail -4 $HOME/DATATEMP8.DOMA1 > $HOME/DATATEMP9.DOMA1
head -1 $HOME/DATATEMP9.DOMA1 > $HOME/REVDATE2.DOMA1
tail -3 $HOME/DATATEMP9.DOMA1 > $HOME/DATATEMP10.DOMA1
head -1 $HOME/DATATEMP10.DOMA1 > $HOME/REV1.DOMA1
tail -2 $HOME/DATATEMP10.DOMA1 > $HOME/DATATEMP11.DOMA1
head -1 $HOME/DATATEMP11.DOMA1 > $HOME/NAM1.DOMA1
endif
nawk -f /axe1/DOMA/batch/addpipe $HOME/DRGNO.DOMA1 > $HOME/DRGNO1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/CHECK.DOMA1 > $HOME/CHECK1.DOMA2
```

```
nawk -f /axe1/DOMA/batch/addpipe50 $HOME/DESC.DOMA1 > $HOME/DESC1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/PRODGRP.DOMA1 > $HOME/PRODGRP1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/ORIG.DOMA1 > $HOME/ORIG1.DOMA2
nawk -f /axe1/DOMA/batch/addpipe $HOME/DATE.DOMA1 > $HOME/DATE1.DOMA2
if ( -e $HOME/REV1.DOMA1 ) then
   nawk -f /axe1/DOMA/batch/addpipe $HOME/REV1.DOMA1 > $HOME/REV11.DOMA2
endif
if ( -e $HOME/REV2.DOMA1 ) then
   nawk -f /axe1/DOMA/batch/addpipe $HOME/REV2.DOMA1 > $HOME/REV21.DOMA2
endif
if ( -e $HOME/REVDATE1.DOMA1 ) then
   nawk -f /axe1/DOMA/batch/addpipe $HOME/REVDATE1.DOMA1 > $HOME/REVDATE11.DOMA
endif
if ( -e $HOME/REVDATE2.DOMA1 ) then
   nawk -f /axe1/DOMA/batch/addpipe $HOME/REVDATE2.DOMA1 > $HOME/REVDATE22.DOMA
endif
if ( -e $HOME/NAM1.DOMA1 ) then
   nawk -f /axe1/DOMA/batch/addpipe $HOME/NAM1.DOMA1 > $HOME/NAM11.DOMA2
endif
if ( -e $HOME/NAM2.DOMA1 ) then
   nawk -f /axe1/DOMA/batch/addpipe $HOME/NAM2.DOMA1 > $HOME/NAM21.DOMA2
endif
touch $HOME/update.DOMA2
exit
```