15

A Model Based Approach to Enterprise Wide Information Support

P.E. Clements, I.A. Coutts and J.M. Edwards
MSI Research Institute,
Loughborough University, Loughborough, Leics, LE11 3TU, UK.
Tel: +44 1509 228250
Fax: +44 1509 267725
Email: p.e.clements@lboro.ac.uk

Abstract

This paper demonstrates an ability to bridge between conceptual information models representing a manufacturing enterprise and the realisation of these models within a relational database management system. The information architecture defined within the paper is based on the ANSI Three-Schema approach to database management which splits the data management issues into three separate parts and thereby realises important benefits in complex systems where information sharing needs change frequently.

The tools created to implement this information architecture support the following lifecycle phases of requirements definition, detailed design, and system execution and maintenance. This paper concentrates on the implementation and system execution issues.

The approach provides direct support of system change by providing structured diagramming and mapping tools which semi-automate information requirements definition (and ongoing change in requirements), and model driven systems implementation tools which facilitate the construction and runtime management of the required system using the services of an integration infrastructure.

J. Goossenaerts et al. (eds.), *Information Infrastructure Systems for Manufacturing*

Keywords

Information systems, Three-Schema, Integration Infrastructure

1 INTRODUCTION

(Sun Microsystems,1994), in their DOE project, stated that:

> "Modern-companies are increasingly becoming information based organizations dependent upon the continuous flow of data for virtually every aspect of their operations. ***However, their ability to handle data is breaking down because of the volume of information is growing at a faster rate than their ability to process it.***
> Computer hardware performance is not the problem. The last decade has seen a ten-thousand fold increase in processing performance. ***The failure lies in software: traditional approaches to designing, developing, and maintaining information processing systems continue to limit our ability to manage data.***
> The lack of parity between hardware and software systems ***along with the flexibility limitations of monolithic information architectures*** waste computer resources. Everyone is affected: the company, its projects, and users. ***The problem is not just with the developer or development process, but with the underlying technology upon which the information systems are built.***
> ***To move forward and escape the limits of traditional information processing methodology requires a better approach*** -- one which employs a more flexible paradigm for the construction and maintenance of software systems so that they can meet the ever-changing needs of companies today"

This quotation supports a general consensus view that a key issue of the 1990's is the ability for world class manufacturing enterprises to handle their information in a structured and controlled manner. Ad-hoc approaches to the design and maintenance of information systems need to be replaced by a more controlled and flexible ones. In addition an ability to readily respond to changes in information needs of the enterprise is required. However any new approach should maintain a closer correspondence between organisational procedures, adopted to support the design, construction and maintenance of information systems, and their underlying supporting information structures.

The Manufacturing Systems Integration (MSI) Research Institute have, over a number of years, developed such an approach through seeking ways of bridging between the theoretical world of system models and the physical world of implementation technology. See Figure 1.

The methodology proposed and described in this paper seeks to meet the following requirements:

(a) that enterprises must react in a flexible and speedy manner to their changing information requirements;
(b) that end-users must be disassociated from implementation details;
(c) that user applications must be isolated from implications of implementation technology.

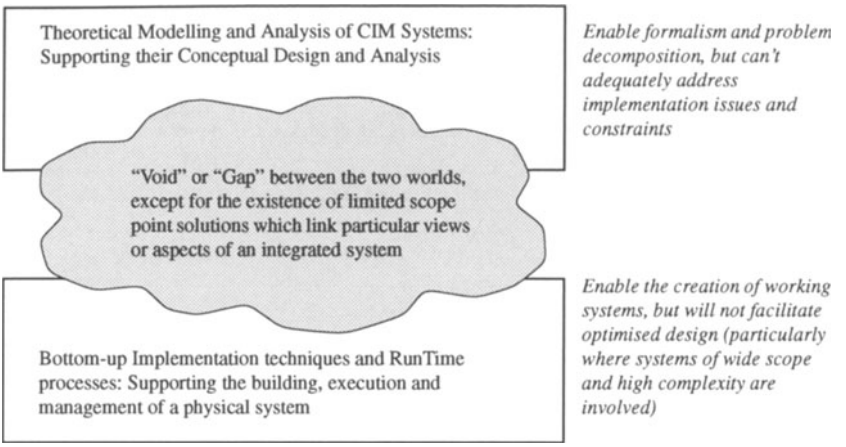

Figure 1. Need to bridge the "Gap" between Theoretical and Physical Worlds.

The paper is structured within three principal sections which describe a) the approach, b) the method of implementation and c) a case study which demonstrates the benefits of the methodology.

2 THE APPROACH: A MODEL-DRIVEN INFORMATION ARCHITECTURE

Researchers at the MSI Institute have sought to bridge between models of candidate systems and real world physical implementations of them in various ways (Aguiar, 1993), (Gilders, 1995), (Murgatroyd, 1993) by considering different perspectives (primarily function, behaviour, resource and information views) of an enterprise. This paper concentrates on using models to drive the information architecture of an enterprise in order that the following can be achieved:

(a) separation of the user/application developer from a need to understand particular characteristic properties of the underlying database system in terms of structure, mechanisms and management paradigms
(b) operation in an heterogeneous database environment;
(c) accommodation of legacy based information systems, and;
(d) support for system change

MSI adopted the ANSI Three-Schema approach, as a means of building a model based architecture. The ANSI approach defines three schemas or views of how database systems should be managed, namely:

(a) the External schema, which is the users view of the database;

(b) the Conceptual schema, which is an abstract definition of the database, and should represent the data and relationships between the data without considering the physical resources available or the database system within which it is stored;

(c) the Internal schema, which deals with the physical representation of the data and its organisation.

The relationship between the schemas is shown in Figure 2,

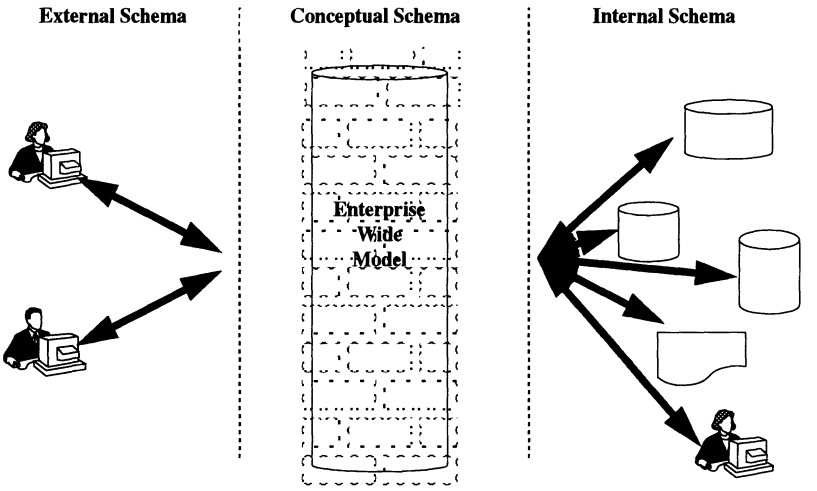

Figure 2. The ANSI Three Schema Architecture.

Based on a Three-Schema approach MSI researchers developed an information architecture which covers the various phases in the lifetime of information systems, see Figure 3. This figure demonstrates a model driven approach to lifecycle support. It also introduces the principal information tools which were produced to realise model driven database creation, population and runtime access.

The following sections of this paper detail the manner in which the three schema are supported via an integrated set of tools.

2.1 Design Phase

During the design phase of an information systems project a description of an information model is created in a neutral format from which a number of database implementations can be built. This neutral model corresponds to the conceptual information schema.

After an initial survey of the techniques available for describing a conceptual schema a decision was taken to use EXPRESS and EXPRESS-G. This decision was based on the following principal issues: (a) EXPRESS-G provides a good graphical interface to the EXPRESS language (b) EXPRESS is readily computer readable (c) within the manufacturing domain EXPRESS had already been adopted by the STEP community (ISO, 1995) and (d) at the time of the decision in 1991, there was a strong possibility of it being adopte d as an ISO standard.

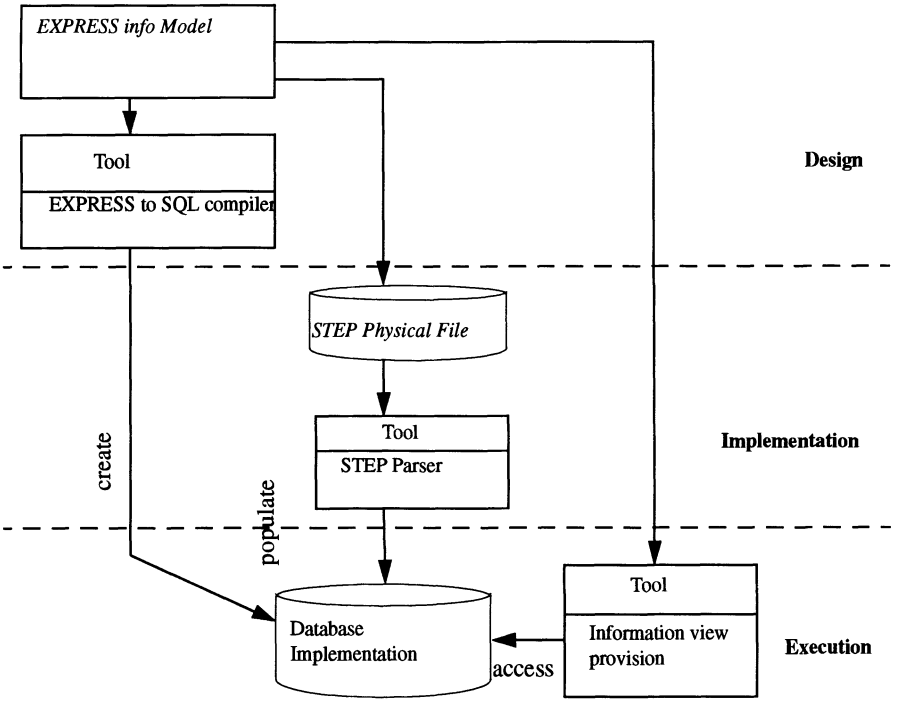

Figure 3. The information architecture viewed against the enterprise engineering life-cycle.

One major problem was that in 1991 when the research reported in this paper began, there were no tools available to transform an EXPRESS model in to a set of relational tables. (Eggers, 1988) had proposed an initial mapping of the EXPRESS constructs onto the relational paradigm but no actual tools were available. Hence, the first stage of implementing the conceptual schema involved creating a tool to transform an EXPRESS model into a relational form (Clements, 1991) and in particular be consistent with requirements of the INGRES database. An EXPRESS-to-SQL compiler was written which generates as outputs: (a) a set of *create table* statements which correspond to the EXPRESS model, and (b) a set of files which allow the EXPRESS model to be reconstructed for subsequent use during implementation and execution life-phases.

2.2 Implementation Phase

Once the structure of the database has been created the next logical stage was to develop a method populating data repositories.

As part of the STEP standardisation work which occurred alongside effort aimed at realising the EXPRESS language description, the STEP standard sought to define a physical file

description which allows a data repository defined by the conceptual model to be populated by an ASCII file. As for the conceptual schema in 1991, no tools were available hence a STEP parser was created which:

(a) checked the STEP physical file for syntactic correctness according to the proposed standard;
(b) checked the STEP physical file for semantic correctness against the particular EXPRESS model being populated, and;
(c) created the *insert table* statements required to populate the data repository.

2.3 Execution Phase

The main design criteria for the execution phase of the information architecture is the ability to separate logical descriptions of information from their physical implementation.

Hence the information architecture contains a tool which allows application developers to specify a logical name which maps onto a set of attributes specified by the conceptual model. Thus at runtime an application or end-user can execute the logical objects, and mechanisms within the information architecture will map the logical name onto the appropriate EXPRESS attributes and thereby automatically create SQL statements. These statements can then be executed by engines provided by the integration infrastructure, and presented back to the end-user/ application on successful completion of the database request.

3 THE IMPLEMENTATION: AN EXECUTION ENVIRONMENT

The mechanisms described in the following sections are concerned primarily with the runtime execution environment. This environment, see Figure 4 was created to demonstrate the model driven Three schema approach to information integration.

3.1 Implementation of the External Schema

When a user application needs to access particular information contained within the underlying databases it can realise this via a number of services. For each of these services the user needs to pass a limited set of identifiers (the specifics of which will depending upon their security profile) to the information view provider, Eventually the user application will eventually receive data back from the view provider in a format previously specified by the user/user application.

The following services are provided in the current implementation:

(a) obtain - this allows the user to receive information concerning the particular identifier passed to the view provider;
(b) submit - this allows the user to add information to the underlying database;
(c) modify - this allows the user to modify particular information within the database;
(d) erase - this allows the user to delete a particular item from the database;

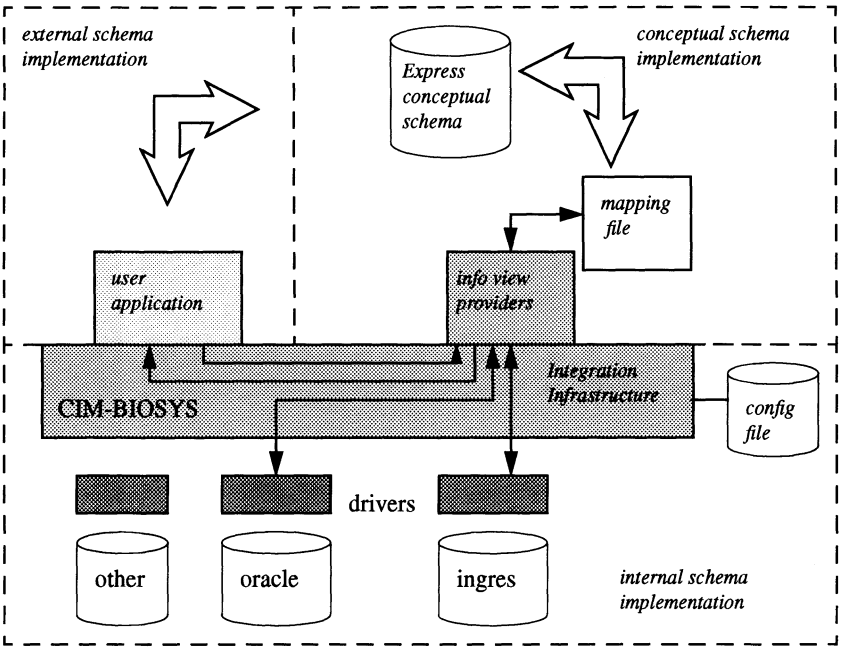

Figure 4. Overview of the runtime information architecture.

3.2 Implementation of the Conceptual Schema

On receipt of an information request, the view provider has to perform a number of tasks, namely:

(a) map the logical name onto respective attributes identified by the EXPRESS models. For example, CARwheel has the attributes radius, make, frontOrBack;

(b) generate a set of SQL statements which collectively service the requested call. This function is enabled by storing an internal representation of the EXPRESS models in the view provider environment (set of tree structures) from which it can deduce the set of necessary statements;

(c) pass this set of statements to relevant device drivers for submission to their respective local database management systems; and

(d) receive the results of the query and format the results in which is meaningful to the requesting application.

3.3 Implementation of the Internal Schema

The integration infrastructure provides the location transparency required to implement the internal schema. It does this by providing the services necessary to enable interaction between various software objects (viz: user applications and the view provider application) which com-

prise a system. It provides a consistent set of services which is independent of the objects's physical location, or the operating system and network protocols used. Researchers at MSI created such an integration infrastructure called CIM-BIOSYS (CIM Building Integrated Open SYStems) in the late 80's, a detailed treatment of which is given in (Coutts, 1992).

A message packet generated by the view provider contains information regarding necessary SQL requests to a database at some "logical" location. Using a method analogous to the view provider approach, CIM-BIOSYS uses configuration data to identify the physical location of the database and thereby directs messages to appropriate drivers. The driver interprets the request, interrogates the database and creates reply messages sending them via CIM-BIOSYS to the view provider.

4 AN INDUSTRY BASED CASE STUDY

To evaluate and demonstrate the benefits of MSI's approach to information systems design and construction a proof of concept case study system has been produced which is based on plant systems and data currently in production at the site of a UK contract manufacturer of high complexity printed circuit boards (PCB's). The application area chosen was the control of a surface mount PCB assembly line, see Figure 5.

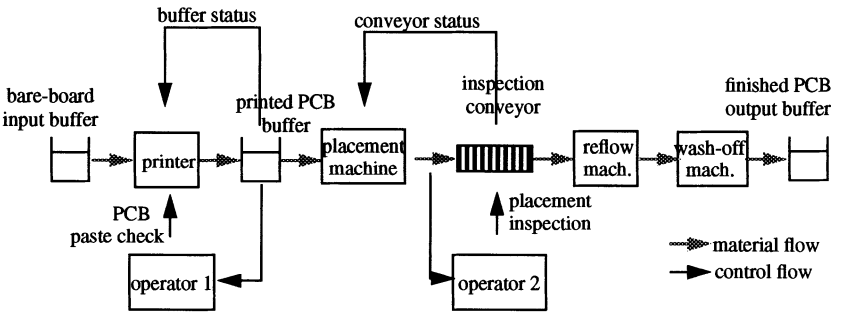

Figure 5. Surface mount technology assembly line.

The demonstrator system comprises a number of user applications which supervise, control and monitor the "printing" of PCB's with solder paste, the "population" of PCB's with components and the "inspection" of PCB's following placement and reflow soldering. A single information view provision application handles the requirements off all user applications in the case study system.

4.1 External Schema of the Demonstrator System

An EXPRESS model was created which structures the information entities and entity relationships of the demonstrator system, see Figure 6.

```
SCHEMA execution;
    ENTITY assemblyLine;
        name            : STRING;
        machines        : SET[0:?] OF machine;
        operators       : SET[0:?] OF operator;
        batch           : batch;
        printInspRate   : INTEGER;
    END_ENTITY;
    ENTITY machine ABSTRACT SUPERTYPE OF (ONEOF (printer, conveyor));
        currentBoard    : board;
        name            : STRING;
    END_ENTITY;
    ENTITY operator;
        name            : STRING;
    END_ENTITY;
    ENTITY printer SUBTYPE OF (machine);
        pasteHeight     : REAL
    END_ENTITY;
    ENTITY conveyor SUBTYPE OF (machine);
        speed           : INTEGER;
    END_ENTITY;
    ENTITY batch;
        boards          : SET[0:?] OF board;
        noOfBoards      : INTEGER;
    END_ENTITY;
    ENTITY board;
        quality         : STRING;
        status          : STRING;
        id              : INTEGER;
    END_ENTITY;
END_SCHEMA;
```

Figure 6. An abridged version of the EXPRESS file.

Then a number of information identifiers were created and assigned to information entities to allow user applications to interrogate the data repositories without needing to know the underlying database structures. In this case study an Ingres database was used to store the physical data. Typical identifiers included PrinterInspectionRate and StatusOfBoardsOnLine.

4.2 Conceptual Schema of the Demonstrator System

For this example two information identifiers were specified, PrinterInspectionRate and StatusOfBoardsOnLine which in the conceptual schema are mapped as below:

(a) PrinterInspectionRate-> execution.assemblyLine.printInspRate;

(b) StatusOfBoardsOnLine -> execution.assemblyLine.name, execution.board.status.

On receipt of an information request from a user application (which is transmitted via the integration infrastructure), the view provider makes use of the EXPRESS model to create the set of SQL statements required to execute the query, see Figure 7. The SQL statements are then

```
create view v1 (en2id, en3a13)
as select en2.en2id, en3.en3a13
from en2, e8, en3
where en2.en2id = e8.en2id and e8.en3id = en3.en3id;\g
create view v2 (e1a0, en3a13)
as select e1.e1a0, v1.en3a13
from e1, e4, v1
where e1.e1id = e4.e1id and e4.en2id = v1.en2id;\g
```

Figure 7. SQL statements necessary to execute query "obtain (StatusOfBoardsOn Line)".

packaged in a message and submitted to the integration infrastructure which transmits them to the Ingres device driver so that the queries can be serviced by the database.

Subsequently the view provider will receive an incoming message from the integration infrastructure which contains the results of the query. The results are then transformed into a predefined presentation format and transmitted to the requesting application via the integration infrastructure.

4.3 Internal Schema of the Demonstrator System

To enable the Ingres database to be accessed via standard CIM-BIOSYS services, a device driver (Leech, 1993) was created which essentially enables CIM-BIOSYS and Ingres to communicate in a meaningful manner. The basic functionality of the Ingres driver was to:

(a) create a physical link using UNIX pipes between CIM-BIOSYS and the Ingres database management system;
(b) initiate a database session for CIM-BIOSYS within the Ingres database;
(c) submit SQL statements generated by the view provider to the Ingres database, and;
(d) pass the results of the queries back to the view provider.

Figure 8 shows the internal schema of the Ingres database, that represents the EXPRESS model shown in Figure 6 and which was created automatically from the EXPRESS model by the compiler. It can be seen by examining Figure 8 and Figure 6 that the compiler creates intermediate tables (for example e4) which are used in join statements to link the entity "assemblyLine" to the entity "batch".

5 CONCLUSIONS

The approach described in this paper provides support for two classes of system change, (a) the system information requirements through its model driven approach, and (b) the system implementation technology through its deployment on an integration infrastructure.

More specifically the approach addresses the three requirements identified in the introduction:

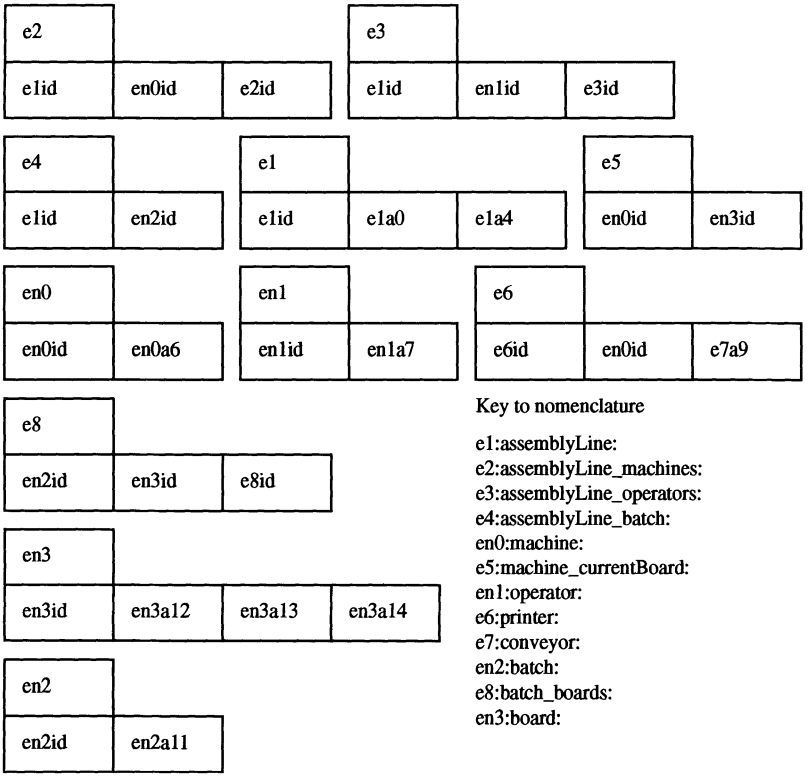

Figure 8. Underlying database structure of EXPRESS file.

(a) enables enterprises to react in a flexible and speedy manner to their changing information requirements through enabling tool supported creation of information elements which automates much of the system programming which has always been the cause of delay in information systems in the past.;

(b) it decouples end-users from implementation details by realising a logical to physical mapping which associates user information requests to database fields;.

(c) it realises a level of implementation independence through its use of an integration infrastructure. The use of CIM-BIOSYS meant that the methodology development could be focused on information design, creation and execution whilst the integrating infrastructure allowed location transparency and provided configurable connection to external packages.

6 ACKNOWLEDGEMENTS

The authors wish to express their thanks to the members of MSI who aided in the development of this methodology and in particular Marcos Aguiar, Jack Gascoigne, Paul Gilders, Allan Hodgson, Shaun Murgatroyd and Richard Weston. Also thanks to the CDP directorate of EPSRC for their continued support of the work.

7 REFERENCES

Aguiar, M.W.C., Roberts, S. and Weston, R.H. (1993) A Model Based Approach Supporting the Life Cycle of Integrated Manufacturing Enterprises, *Proc. of ICCIM Conf.*, Singapore. September pp 246-256.

Clements, P.E. (1991) Internal Report on EXPRESS to SQL Database Translation

Coutts, I. A. et al. (1992.) Open Applications within Soft Integrated Manufacturing Systems, *Proc. of Int. Conf. on Manufacturing Automation*, Hong Kong, ICMA

Eggers, J., (1988) Implementing Express in SQL, McDonnell Douglas Aerospace Information Services Company, USA, October 1988.

Gilders, P.J., Murgatroyd, I.S, Wright, C.D. and Weston, R.H., (1995) The Implementation of a Graphical Estelle Generation Tool Using MetaCase Technology, *Proc. of MetaCase '95*, Sunderland, UK

Leech, M., (1993) MSI Internal Report on the Ingres Database Driver

ISO 10303-1,(1995) International Standards Organisation.

Murgatroyd, I.S., Edwards, J.M. and Weston R.H. (1993)"Tools to Support the Design of Integrated Manufacturing Systems - An Object Oriented Approach - *IEPM,* , Mons, Belgium.

Sun Microsystems (1994) Project DOE: Future Solaris Technologies, TR-94-30,