KfK-PFT 145

ESPRIT Project 322

CAD*I

CAD Interfaces

Status Report 5

I. Bey and J. Leuridan (editors)

ESPRIT Subpogramme 5

Computer Integrated Manufacture (CIM)

R&D area CAD/CAE

Pages 221
Figures 100
Tables 5
References 45

March 1989

Übersicht

Das Ziel des ESPRIT Projektes Nr. 322: "CAD Interfaces" ist die Definition der wichtigsten Schnittstellen in CAD/CAM Systemen für: Datenaustausch, Datenbank, Finite Elemente Analyse, experimentelle Analyse und fortgeschrittene Modellierung. Die Definition dieser Schnittstellen wird in enger Zusammenarbeit mit den internationalen Standardisierungsbemühungen in diesem Bereich durchgeführt.

Das Projekt soll dazu beitragen, europäisches Know-how zusammenzutragen und den breiten industriellen Einsatz von CAD/CAM Systemen zu beschleunigen. Der europäische Einfluss auf die internationalen Standardisierungsgremien soll dabei gestärkt werden.

Der vorliegende Bericht ist eine Dokumentation der Ergebnisse der im vierten Projektjahr durchgeführten Arbeiten.

Abstract

ESPRIT Project 322, "CAD Interfaces", has been established to define the most important interfaces in CAD/CAM systems for data exchange, data base, finite element analysis, experimental analysis, and advanced modelling. The definitions of these interfaces are being elaborated in harmony with international standardisation efforts in this field.

The Project is to contribute to the compilation of European know-how and to facilitate the application of CIM methods in industry on a broad basis. In this way, the European influence on international standardisation bodies is to be strengthened.

This report is a documentation of the results of work carried out in the fourth year of the Project.

Editors' addresses

I. Bey
Kernforschungszentrum Karlsruhe GmbH
Postfach 3640, D-7500 Karlsruhe 1/FRG

J. Leuridan
Leuven Measurement and Systems
Interleuvenlaan 65, B-3030 Heverlee/Belgium

Partners in the Project are:

Bayerische Motorenwerke AG/FRG
CISIGRAPH/France
Cranfield Institute of Technology /UK
Danmarks Tekniske Hoejskole/DK
ERDISA/Spain
Gesellschaft fuer Strukturanalyse mbH/FRG
Katholike Universiteit Leuven/Belgium
Kernforschungszentrum Karlsruhe GmbH/FRG
Leuven Measurement and Systems/Belgium
NEH Consulting Engineers ApS/DK
Rutherford Appleton Laboratory/UK
Universitaet Karlsruhe/FRG.

Table of Contents

Introduction

1. Working Group 1: Wire Frame/Drafting 1

1.1 General 1
1.2 Goals stated at the beginning of the work 1
1.3 Achievements so far 1
1.4 Contributions to international cooperation within Europe 2
1.5 Contributions to the standardisation effort at international level. Achievements so far 2
1.6 On drafting data modelling 3
1.7 CAD*I drafting data specification 5
1.8 The Neutral File Adapting System (NFAS) 24
1.9 Experience gained using the Neutral File Adapting System (NFAS) 35

2. Working Group 2: Solids 43

2.1 Specification 44
2.2 Semantics of the CAD*I data structures 45
2.3 Algebraic specification of CSG 46
2.4 Scanner and parser 54
2.5 Processor development for version 3.3 54
2.6 Results of solid model transfer tests during the 3rd International CAD*I workshop in Copenhagen 85
2.7 Coordination of European efforts regarding STEP 89
2.8 Exploitation 90
2.9 References 90

3. Working Group 3: Surfaces 93

3.1 Introduction 93
3.2 Results 93
3.3 Standardisation activities 94
3.4 Outlook 95
Appendix A 96
Appendix B 99

4.1 Working Group 4: Data Base 103

4.1.0 General introduction 103
4.1.1 Problems common to CAD users and CAD developers 104
4.1.2 The proposal of Working Group 4 105
4.1.3 Achievements 111
4.1.4 Main results and future tasks 127
4.1.5 Expected benefits for the users of the CAD*I subroutines 127
4.1.6 Outlook 132

4.2 Working Group 4: Networks 133

4.2.1 Introduction 133
4.2.2 CAD data exchange and network services 133
4.2.3 File transfer 137
4.2.4 Message exchange 158
4.2.5 Conclusions 169
4.2.6 References 171

5. Working Group 5: Advanced Modelling 173

5.1 Introduction 173
5.2 Report on performed work 174
5.2.1 Handsketching Input System (HIS) 174
5.2.2 Design by Technical Terms Implementation (DTT) 176
5.2.3 Technical Modelling (TMI) and Geometric Associativity Interface (GAI) 187
5.2.4 Design by features 194
5.2.5 Relevance of STEP 199
5.3 Results 203
5.4 Outlook 206
5.5 References 206

6. Working Group 6: FEM Model Description 209

6.1 Introduction 209
6.2 Summary of Final Report 209
6.3 Achievements 209
6.4 Conclusion 210

7. Working Group 7: Dynamic Model Optimization 211

7.1 Introduction 211
7.2 Dynamic model optimization: aim and approaches 211
7.3 Contents of the final Working Group 7 report 213

8. Working Group 8: Experimental Dynamic Structural Analysis 215

8.1 Synopsis of Working Group 8 final report 215
8.2 References 217

9. Standardisation Activities 221

Introduction

This fifth Status Report of ESPRIT Project 322 "CAD Interfaces" (CAD*I) is a summary of the main results of the fourth year of the project's running time, covering the R&D work period from November 1987 to October 1988.

At this stage and taking into account the results of the **international standardisation** efforts in the field of CAD interfaces (especially after the Tokyo ISO TC 184 SC 4 meeting of Nov/Dec 1988) it is possible to state that the CAD*I project has achieved most of the original goals projected at the beginning: STEP, the first international Standard for Exchange of Product Definition Data, has reached the status of a Draft Proposal on ISO level. CAD*I work and CAD*I staff have influenced this standardisation process technically and politically in many important issues in a very strong way, helping to get European requirements to be included and accepted by the international community active in this field. An overview on the recent standardisation activities of the project is given in section 9.

After four years of research and development work a lot of results have been merged into **prototype software products**. They are based on the specification for neutral files done in the first part of the project and were written using modern software tools. Hints to available preprocessors, postprocessors, software tools, and other software packages at the partners sites are given in the different sections describing the project results. These descriptions follow the line of the 8 different working groups into which the R&D work was subdivided at the beginning of ESPRIT Project 322: CAD*I. Especially in the field of CAD data exchange and CAD-FEM interfaces pre- and postprocessors have been developed for a big number of commercial systems. Among them are: BRAVO3 (Applicon), CATIA (IBM), EUCLID (MATRA DATAVISION), PROREN (ISYKON), ROMULUS (Shape Data), STRIM (CISIGRAPH), TECHNOVISION (Norsk Data), ANSYS, NASTRAN, PAFEC. The interest and commitment of CAD and FEM vendors has steadily increased in the last time and there is a very good chance for vendors applying CAD*I results to be the very first ones to have STEP processors available on the market.

The CAD*I project will continue its work in the track of good partnership and cooperation also in its fifth year. Efforts to push standardisation and to convert research results into finally marketable products will be done with the usual commitment to the goals of the ESPRIT programme.

With this report the CAD*I staff wants to present the new CAD*I results again to a broad audience of experts in industry and research institutions. Detailed final reports of Working Groups 6 through 8 are to be published in spring 1989 at Springer Verlag Heidelberg. Acknowledgements are given to all partners, especially for those who left the project as scheduled after four years of productive work and personal commitment.

Karlsruhe, December 1988.

1. Working Group 1: Wire Frame / Drafting

1.1 General

During the fourth project year emphasis was placed on drafting data specification. The CAD*I Wire Frame Specification, which was developed at the beginning of the CAD*I project, is fundamental to CAD*I solids, hence these data structures have been frozen, see: Specification of a CAD*I Neutral File for CAD Geometry, Version 3.3 /1/.

1.2 Goals stated at the beginning of the work

- Specification of wire frame data to be integrated in the CAD*I Geometry Specification.

- Transcription of atomic statements on contents of technical drawings to have a basis, i.e. the so-called statement catalogue, for the CAD*I Drafting Data Specification.

- Specification of drafting data in a data definition schema, which describes how technical drawings can be represented in a neutral file for exchange or archiving. Emphasis shall be placed on influencing design decisions for STEP because no CAD*I processor for drafting data has been planned.

- Improvement of CAD/CAM data exchange by adapting neutral files to the requirements of the CAD system receiving the data and of the applications for which the CAD system is used for. The Neutral File Adapting System is the subject of chapter 1.8.

1.3 Achievements so far (considering more the strategic relevance of the results)

- The CAD*I geometry specifications had a strong influence on STEP.

- CAD*I WG1 initiated a working relationship with German experts on drawing standards. The aim of this cooperation is to develop a statement catalogue which is in fact a CAD drawing standard. First steps towards international liaison have been taken.

- Both CAD*I requirements for drafting and CAD*I views on this topic are in ISO discussion. Some important facts which have to be considered by each developer of data structures for data describing drawings are subject of chapter 1.6.

- Experience gained using the Neutral File Adapting System (NFAS) to adapt CAD models to the NC-programming environment shows that the expectations could be exceeded. A large number of applications require the use of NFAS to be integrated in an industrial production process (see chapter 1.9).

1.4 Contributions to international cooperation within Europe

All CAD*I WG 1 members attend meetings of ISO/TC184/SC4/WG1 with a DIN NAM 96.4 mandate. Up to now no European standardisation body is represented within the STEP community.

1.5 Contributions to the standardisation effort at international level. Achievements so far

The CAD*I working groups 1 to 3 developed the CAD*I Geometry Specification which became a contribution to ISO/TC184/SC4/WG1. The specification was published /1/.

Chapter 1.7 of this report contains data structures for the neutral representation of dimensions. These structures are realisations of CAD*I requirements and illustrate the necessary improvement on structures which are well-designed for processor implementors but which do not represent knowledge about the respective applications. The schema has been written in Express because it needs to be discussed within the ISO TC184/SC4 community which has chosen this description language. Express is similar to CAD*I HDSL and has been influenced by it.

Some examples showing how data according to the proposed structures may appear are attached to the schema, that has been contributed to ISO/TC184/SC4/WG1 (doc no. N252) /2/.

1.6 On Drafting Data Modelling

Eight CAD*I theses for discussion within the ISO are noted in the following.

I Awareness of ambiguity in technical drawings has led to restrictive drawing standards. Queries which have to be answered by interpreting drawings that comply with conventional drawing standards can be answered without any element of doubt, guesswork or having to refer back to the originator. This certainty that has been achieved for manually prepared drawings is also required for drawings described according to an exchange standard like STEP. That is to say, any queries made concerning the received data must be able to be answered without any element of doubt, guesswork or having to refer back to the sender. Otherwise additional effort at the receiving end would be necessary and in the extreme instance it may be senseless to archive exchange files.

II A fundamental fact for the specification of data structures is that developers of data structures and/or processors never have equivalent know-how about the applications. Consideration of this and of the role of application experts, who may not have any data processing knowledge, leads to the realisation that an application data exchange specification needs to be self-contained, i.e. for drafting: a drafting data specification has to include the drawing standard.

III When comparing a drawing standard that has been incorporated in exchange standard documentation with conventional drawing standards, it can be stated that

(1) ambiguity must be avoided to the same extent regardless whether the technical drawings are manually prepared or described by data according to the exchange standard

(2) exceptional cases should be minimised for CAD (conventional drawing standards cannot be perfectly implemented in a CAD system due to too many admissible exceptions)

(3) conventional drawing standards presuppose an understanding typically possessed by technical draughtsmen but not by programmers. Therefore, within an exchange standard, documentation of constraints has to be free of implicit assumptions, contain integrity constraints (which are mostly self-evident for

technical draughtsmen), and contain constraints for the graphical form.

IV The main task of data model development is to recognise the syntactical rules to which the elements described by the data will be subject. In the context of the drawing description these rules are determined by conventional drawing standards. However, the drafting rules are not always explicitly given. Therefore, developers of a CAD exchange standard who are concerned with drafting run the risk of looking over many implicit rules. The recognised rules will become the constraints of each drawing description.

V Within data structure development constraints can be taken into consideration in two ways:

(1) the data structures can be designed in such a manner that many constraints are inherent, i.e. the data according to the data structures cannot violate such inherent constraints

(2) the data structures can be documented together with the constraints (by WHERE-clauses or informally) for the purpose of instructing processor developers to provide consistent data by program code.

VI Only the exclusion of data processing knowledge by presentation of the constraints which have been considered by the design of data structures will enable communication regarding the adequacy of the specified data structure collection for a specific application. Only if a common understanding is guaranteed by a document that includes the following three points potential users of the exchange standard will be able to evaluate this standard. In this case they can leave data structure design decisions to a small number of experts.

(1) Which assumptions have been made about the applications,

(2) which constraints governing the information represented in exchange files have been considered,

(3) the degree of responsibility, that exchange data will conform to applications remaining at the sending system.

VII Concerning the requirement "to transfer consistent data only", many developers take the contrary view that only "functionality" has to be provided by exchange data.

This would mean, for instance, that nearly arbitrary graphs, whether or not they can be recognised and interpreted, can be described as carrying specific semantics. In other words, nonsense may be represented by data which is syntactically correct. This opinion is due to the fear of being too restrictive.

VIII Developers of data structures for data describing technical drawings have to consider what happens if the projected shape 2D geometry is derived from 3D shape geometry applying presentation information in the receiving system. In this instance the geometry to which the dimensions have to be related is not represented in the exchange file by named data. It is therefore difficult to ensure that dimensions are consistent and connected with the geometry to be dimensioned.

1.7 CAD*I Drafting Data Specification

```
SCHEMA drafting;

(* Type names that are written in upper case letters are of
types not specified in this schema. *)

(* SUBSCHEMA dimensions *)

(* NB, the supertypes POINT and DIRECTION which are used as
attribute types can be realised by simple coordinates or by
data of complex structure which express the relations to
geometry. Thus these structures are really upward-
compatible. *)

(* If dimension descriptions according to the structures are
linked, the dimensions are connected. Syntactically correct
dimension descriptions always describe valid dimension
graphs. They are free of redundancy. *)

(* The informal descriptions of the structures may look
somewhat complicate - but they contain no constraints
because all constraints have been described formal or are
inherent constraints, i.e. the data according to the
structures cannot violate such inherent constraints. NB.,
structures that provide redundancy (like those of IGES)
require a greater amount of formal or informal description.
*)
```

```
ENTITY dim_predecessor SUPERTYPE OF (initial_dim_attributes,
                                     lin_dim, ang_dim,
                                    arc_dim_par, arc_dim_rad);
END_ENTITY;
```

```
ENTITY initial_dim_attributes SUBTYPE OF
(dim_predecessor);
dim_line_dir :  DIRECTION;
determinator :  POINT;
DERIVE
    pre_list  : LIST [1,#] OF dim_predecessor
         = CREATE_LIST (initial_dim_attributes);
    reference : LOGICAL = id (.T.);
    next_dir  : DIRECTION = id (dim_line_dir);
WHERE
    DEFINITION_SPACE (determinator) = DRAWING_SPACE;
    DEFINITION_SPACE (dim_line_dir) = DRAWING_SPACE;
END_ENTITY;
```

(* An initial_dim_attributes occurrence cannot refer to (or contain) a dim_predecessor occurrence. Therefore, it terminates a list of dimension descriptions. The relation between an element (n) and an element (m) of such a list, where n < m, is "is successor of". If no "embedded entity" is used, those lists are inverted in a physical file.

Both the logical value indicating whether the dimension or dimension chain that is described by such a list starts with an extension line and the real value designating the length of the extension line can be calculated easily. Formal specification of these derived attributes requires Express procedures, which shall not be specified at the moment. However, an informal specification shall be given in the following.

Let the initial_dim_attributes occurrence (i.e. the list terminator) be the last list element having the list index m (the list contains two elements at least). Assume element (i) being the element (MAX(n), 0 < n < m) that has a true reference_line flag or make i = 1, if no such element occurs. Then all elements from i to (m-1) represent

dimensions that are related to the determinator of the initial_dim_attributes occurrence (hence to the extension line in question, if one exists). Now we can state, that no extension line appears at this determinator position, if and only if all offsets of the elements i to (m-1) are zero or omitted. Otherwise an extension line appears and its length is determined as follows. Let S be the family of all subintervalls of [i,m-1] that is contained in the index set (i.e. all intervals [j,k] with i <= j <= k <= m-1 are elements of S). Then the extension line length is

$$MAX(|\Sigma_{n \epsilon s \epsilon S} \text{ offset}_n|)$$

where offset_n is the offset value of list element n. *)

1.7.1 Linear Dimension

(* lin_dim is the structure of descriptions of linear dimensions having no or two perpendicular extension lines. The dimension that can be described according to this structure may stand alone or be contained in a combined dimension graph: *)

```
ENTITY lin_dim SUBTYPE OF (dim_predecessor);
first_attribute    : dim_predecessor;
note               : NOTE;
terminator_inside  : LOGICAL;
offset             : OPTIONAL REAL;
determinator       : POINT;
reference          : LOGICAL;
DERIVE
    pre_list : LIST [1,*] OF dim_predecessor
       = EXTEND_LIST (first_attribute.pre_list, lin_dim);
    pre_ref  : dim_predecessor
       = LAST_TRUE_REFERENCE (first_attribute.pre_list);
    next_dir :  DIRECTION
       = id (pre_ref.next_dir);
WHERE
    DEFINITION_SPACE (determinator) = DRAWING_SPACE;
    determinator <> pre_ref.determinator;
END_ENTITY;
```

(* The dimension line of the dimension that is described by data according to this structure inherits its direction from **pre_ref** (one of its predecessors). Also its start_point is given with regard to a predecessor (see offset below). The dimension line terminates either at the **determinator** or at an extension line that is defined by determinator and is being perpendicular to the dimension line.

The flag **reference** is meaningful for succeeding dimension descriptions only. A dimension description is successor if it is in the relation "is successor of" to the lin_dim occurrence, see initial_dim_attributes. A true reference flag denotes either the extension line at the accompanying determinator (if there is an extension line) or (otherwise) the determinator itself. Dimension lines that are denoted by succeeding dimension descriptions are related to this determinator/extension line until another true reference flag occurs in the sequence (i.e. until pre_ref changes).

If the flag **terminator_inside** is true, both termination symbols of the dimension line are placed between and point to both the reference that is valid for this lin_dim occurrence and the possible new reference that is determined by determinator. If the flag terminator_inside is false, a standard elongation of the dimension line (proposal: 5 times the termination symbol length) beyond the part outlines and/or extension lines may be necessary. This depends on whether there is a preceding or a succeeding aligned dimension line. If two consecutive dimension descriptions have a false terminator_inside flag while the offset of the second is omitted or zero, the dimension lines have a circle instead of the arrow heads at their intersection (see example 7).

If the **offset** value is not equal to zero it follows that an extension line is given at pre_ref.determinator.

If pre_ref is an initial_dim_attributes occurrence, offset is the distance from the initial_dim_attributes.determinator in the direction "initial_dim_attributes.dim_line_dir plus 90°".

If an offset that is not equal to zero is given and pre_ref is a lin_dim, ang_dim, or arc_dim_rad occurrence, this offset represents the orthogonal distance (measured on the extension line) of the dimension line from the dimension line that is denoted by the immediate preceding dimension description. Same applies, if the offset is not equal to zero and the reference is given by an arc_dim_par which is not the immediate predecessor. If the offset is not equal to zero and the reference is given by an arc_dim_par which is the immediate predecessor the offset is measured from the

arc centre in a direction which is centrifugal and which points to the end point of the dimensioned arc.

A superimposed running dimension is given, if a) all dimensions that descriptions are linked are linear dimensions and b) all reference flags are false and c) all offsets are zero or omitted. In this case the first termination symbol is a circle (see example 6). *)

(* Propositions from the statement catalogue:

1. a linear dimension has a straight dimension line

2. a straight dimension line may be terminated by either 2 arrow heads or 2 circles or 1 arrow head and 1 circle

3. a straight dimension line may exceed the arrow heads, but not the circles

4. the length between the arrow heads or circles of a straight dimension line corresponds to the dimensional value with regard to the scale applied

5. where a dimension line has 2 arrow heads they are oriented against each other

6. an extension line contacts the dimension line at the arrow head or circle

7. a straight extension line is directed perpendicular to the straight dimension line

8. a dimension line has exactly 1 dimension note

9. dimension notes may be placed either at leaders or above dimension lines

10. the reading direction of dimension notes has to be oriented parallel to the dimension lines they are assigned to, except: If the dimension note is part of a superimposed running dimensioning, the dimension note may be placed near the arrow head, in line with the corresponding extension line.

*)

1.7.2 Angular Dimension

(* ang_dim is the structure for descriptions of angular dimensions having two dimension line terminations. The dimension that can be described according to this structure may stand alone or be contained in a combined dimension graph: *)

```
ENTITY ang_dim SUBTYPE OF (dim_predecessor);
first_attribute    : dim_predecessor;
note               : NOTE;
develop_dir        : LOGICAL;
terminator_inside  : LOGICAL;
offset             : OPTIONAL REAL;
determinator       : POINT;
reference          : LOGICAL;
next_dir           : DIRECTION;
DERIVE
    pre_list : LIST [1,*] OF dim_predecessor
       = EXTEND_LIST (first_attribute.pre_list, ang_dim);
    pre_ref  : dim_predecessor
       = LAST_TRUE_REFERENCE (first_attribute.pre_list);
WHERE
    DEFINITION_SPACE (determinator) = DRAWING_SPACE;
    DEFINITION_SPACE (  next_dir  ) = DRAWING_SPACE;
END_ENTITY;
```

(* The dimension line of the dimension that is described by data according to this structure terminates either at the determinator or at an extension line that is defined by determinator and is having the direction "next_dir plus 90o". The dimension line, which is an arc, is determined by its start point that is given by offset (see lin_dim), by the inherited dimension line direction which is a direction of the tangent at its start point, by the flag develop_dir (see below), and by the arc centre point that can be calculated because the direction next_dir is given. If the flag reference is true, next_dir becomes the dimension line direction that is hereditaried to a succeeding dimension description.

The flag develop_dir determines the side of the reference that the dimension is positioned at according to the current dimension line direction, that is inherited from the predecessor pre_ref.

terminator_inside, and reference are defined as for the lin_dim structure. *)

(* Propositions from the statement catalogue:

1. an angular dimension has a circular dimension line
2. a circular dimension line may be terminated by either 2 arrow heads or 2 circles or 1 arrow head and 1 circle
3. a circular dimension line may exceed the arrowheads, but not the circles
4. where a dimension line has 2 arrow heads they are oriented against each other
5. a dimension line has exactly 1 dimension note
6. an extension line contacts the dimension line at the arrow head or circle
7. dimension notes may be placed either at leaders or above dimension lines
8. the extension lines of an angular dimension are perpendicular to the tangents of the circular dimension line at its terminators

*)

1.7.3 Arc Dimension

1.7.3.1 Arc Dimension Parallel extension lines

```
(* arc_dim_par is the structure for descriptions of arc
dimensions having two parallel extension lines and two lines
(centre indication lines) that indicate the centre of the
dimensioned arc. The dimension that can be described
according to this structure may stand alone or be contained
in a combined dimension graph: *)


ENTITY arc_dim_par SUBTYPE OF (dim_predecessor);
first_attribute    : dim_predecessor;
note               : NOTE;
terminator_inside  : LOGICAL;
distance           : POSITIVE_REAL;
determinator       : POINT;
reference          : LOGICAL;
next_dir           : DIRECTION;
DERIVE
    pre_list : LIST [1,*] OF dim_predecessor
      = EXTEND_LIST (first_attribute.pre_list, arc_dim_par);
    pre_ref  : dim_predecessor
    = LAST_TRUE_REFERENCE (first_attribute.pre_list);
    determined_arc : ARC_SEGMENT
    = ARC_CALC (first_attribute, determinator, next_dir);
WHERE
    DEFINITION_SPACE (determinator) = DRAWING_SPACE;
    DEFINITION_SPACE (  next_dir  ) = DRAWING_SPACE;
    ARC_LENGTH        (determined_arc) <=
                         RADIUS (determined_arc) * Π/2;
    determinator <> pre_ref.determinator;
    next_dir <> pre_ref.next_dir
END_ENTITY;
```

(* The circular dimension line is an arc that is an offset curve with the determined_arc as basis curve and distance is the distance from the basis curve. The dimension line lies at the convex side of the determined_arc and is connected with it by two parallel extension lines. Within a combined dimension these two lines are not shared with any preceding or succeeding dimension. terminator_inside is defined as for the lin_dim structure.

The start point of determined_arc is the pre_ref.determinator. A centre indication line with this point as an end point has the directions "pre_ref.next_dir plus/minus 90°". The intersection of this line with the line that has the direction "next_dir plus/minus 90°" and that has determinator as an end point is the centre point of the determined_arc. Thus we have two centre indication lines (both with the centre point as end point) of which one is the relevant for succeeding dimensions.

A true reference flag designates the centre indication line through determinator as reference for successors. *)

1.7.3.2 Arc Dimension Radial extension lines

```
(* arc_dim_rad is the structure for descriptions of arc
dimensions having two extension lines through the centre of
the dimensioned arc. The dimension that can be described
according to this structure may stand alone or be contained
in a combined dimension graph: *)


ENTITY arc_dim_rad SUBTYPE OF (dim_predecessor);
first_attribute : dim_predecessor;
note            : NOTE;
offset          : OPTIONAL REAL;
determinator    : POINT;
next_dir        : DIRECTION;
DERIVE
    reference : LOGICAL = id (.T.);
    pre_list  : LIST [1,*] OF dim_predecessor
      = EXTEND_LIST (first_attribute.pre_list, arc_dim_rad);
    pre_ref   : dim_predecessor
      = LAST_TRUE_REFERENCE (first_attribute.pre_list);
    determined_arc : ARC_SEGMENT
      = ARC_CALC (first_attribute, determinator, next_dir);
WHERE
    DEFINITION_SPACE (determinator) = DRAWING_SPACE;
    DEFINITION_SPACE (  new_dir   ) = DRAWING_SPACE;
    ARC_LENGTH (determined_arc) >
                   RADIUS (determined arc) * Π/2
    determinator <> pre_ref.determinator;
END_ENTITY;


(*offset and determinator are defined as for the lin_dim
structure. next_dir becomes the dimension line direction
that is hereditaried to a succeeding dimension description
(because reference is always true). Also terminator_inside
is assumed to be always true and is therefore omitted. (For
reference and terminator_inside see lin_dim.)

The extension lines are elongated to the centre of the
determined_arc and serve therefore as centre indication
lines. A leader starts at the note and points to the
mid_point of the determined_arc. *)
```

(* Propositions from the statement catalogue:

1. an arc dimension has a circular dimension line

2. a circular dimension line may be terminated by either 2 arrow heads or 2 circles or 1 arrow head and 1 circle

3. a circular dimension line may exceed the arrowheads, but not the circles

4. where a dimension line has 2 arrow heads they are oriented against each other

5. a dimension line has exactly 1 dimension note

6. an extension line contacts the dimension line at the arrow head or circle

7. dimension notes may be placed either at leaders or above dimension lines

8. where the angle constituted by the terminating points of the circular dimension line and the centre remains under 90° the extension lines are parallel to the bisector of the angle

9. where the extension lines are parallel the length of the circular dimension line corresponds to the dimensional value

10. where the extension lines are not parallel, they are perpendicular to the tangents of the circular dimension line in its terminating points

11. where the extension lines are not parallel, there has to be a leader pointing from the dimension note to the dimensioned arc

*)

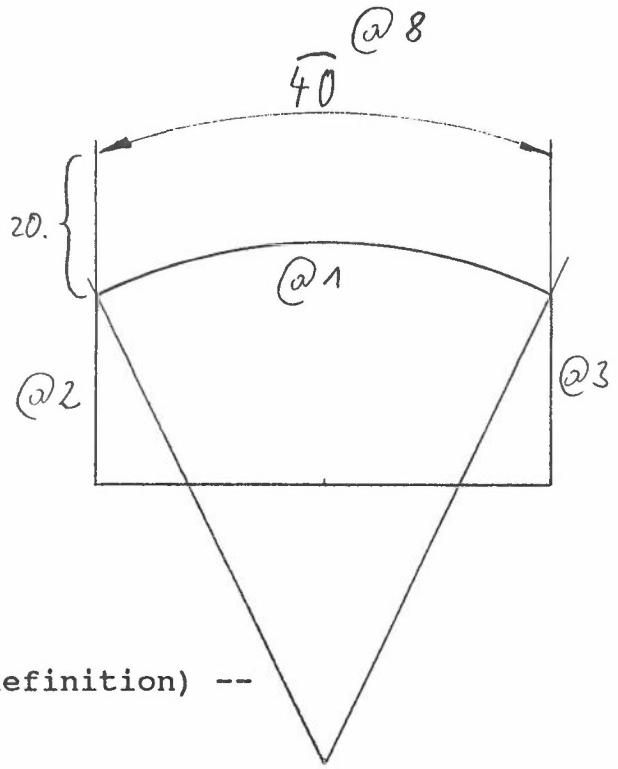
(* END_SUBSCHEMA dimensions *)

END_SCHEMA (* drafting *);

Neutral file examples

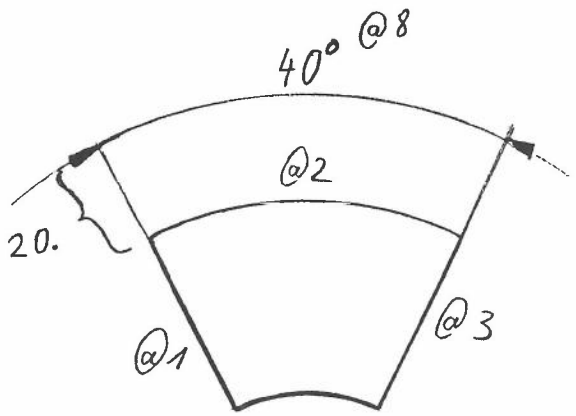
The preliminaries need not be given explicitly but can be given as projections, intersections, directions of any directed curve, etc.. Therefore the keywords that are used shall be considered as representatives of the respective supertypes.

Example 1:
Arc dimension with
parallel extension lines



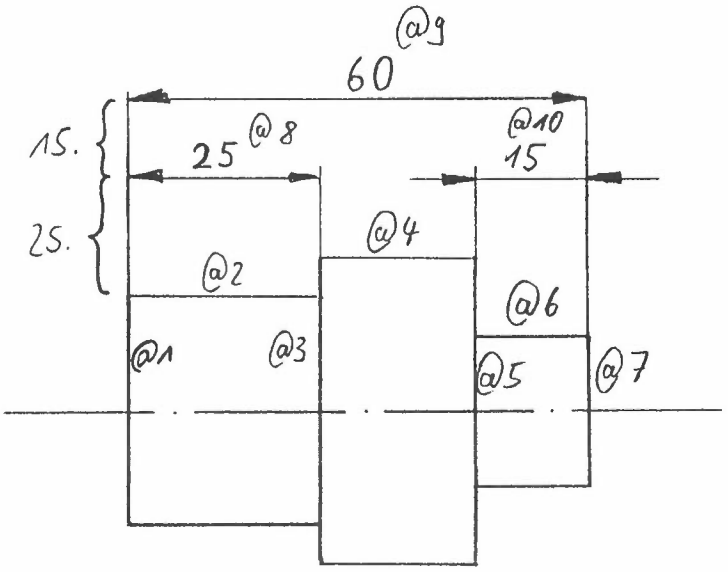

```
!*

-- preliminaries (see figure for definition) --

@1= ARC_SEGMENT
@2= LINE_SEGMENT
@3= LINE_SEGMENT
@4= CARTESIAN_TWO_COORDINATE (assumed is the intersection of
    #1 and #2)
@5= CARTESIAN_TWO_COORDINATE (assumed is the intersection of
    #1 and #3)
@6= TWO_SPACE_DIRECTION (assumed is the direction of the arc
    tangent at #4)
@7= TWO_SPACE_DIRECTION (assumed is the direction of the arc
    tangent at #5)

-- data according to the drafting schema --

@8= NOTE ( needs specification );

*!

@9= ARC_DIM_PAR ( INITIAL_DIM_ATTRIBUTES( #6, #4), #8, .T.,
    20., #5, .F., #7);
```

Example 2:
angular dimension



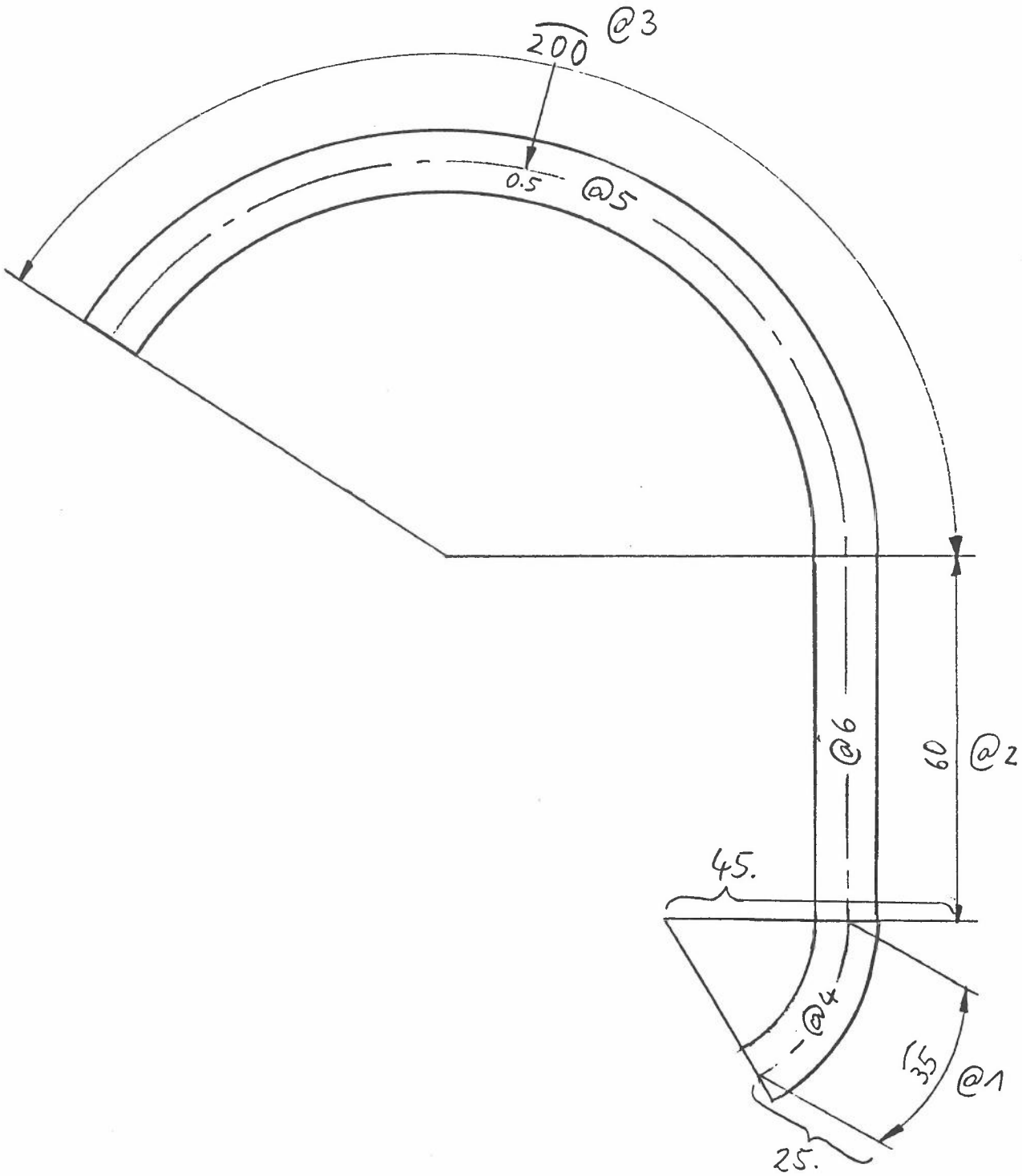

```
!*

-- preliminaries (see figure for definition) --

@1= LINE_SEGMENT
@2= ARC_SEGMENT
@3= LINE_SEGMENT
@4= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
    point of edges #1 and #2 );
@5= CARTESIAN_TWO_COORDINATE ( as #7 - assumed is the
    intersection point of edges #3 and #2 );
@6= TWO_SPACE_DIRECTION ( assumed is the direction of the
    arc tangent at #4 )
@7= TWO_SPACE_DIRECTION (  assumed is the direction of the
    arc tangent at #5 )

-- according to the drafting schema --

@8= NOTE ( needs specification );

*!

@9= ANG_DIM (INITIAL_DIM_ATTRIBUTES (#6, #4), #8, .T., .F.,
    20., #5, .F., #7);
```

Example 3:
chain dimension



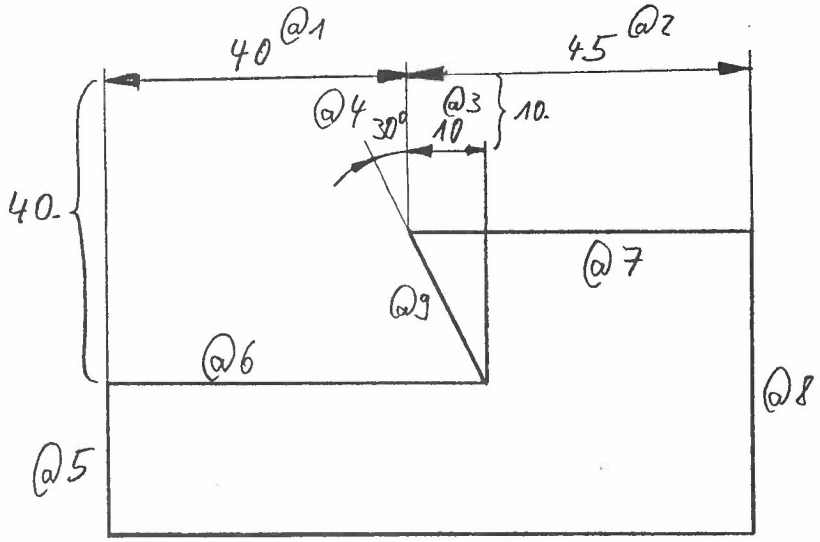

```
!*

-- preliminaries (see figure for definition) --

@1= LINE_SEGMENT       @2= LINE_SEGMENT       @3= LINE_SEGMENT
@4= LINE_SEGMENT       @5= LINE_SEGMENT       @6= LINE_SEGMENT
@7= LINE_SEGMENT
@11= TWO_SPACE_DIRECTION (assumed is the x- direction of
    edge #2 )
@12= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
    point of edges #1 and #2 );
@13= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
    point of edges #3 and #4 );
@14= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
    point of edges #4 and #5 );
@15= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
    point of edges #6 and #7 );

-- according to the drafting schema --

@8=  NOTE ( needs specification );
@9=  NOTE ( needs specification );
@10= NOTE ( needs specification );
*!

@16= LIN_DIM (LIN_DIM (LIN_DIM (INITIAL_DIM_ATTRIBUTES (#11,
    #12), #8, .T., 25., #13, .F.), #9, .T., 15. , #15, .T.),
    #10, .F.,  -15., #14, .F.);
```

Example 4:
chain dimension
used in piping



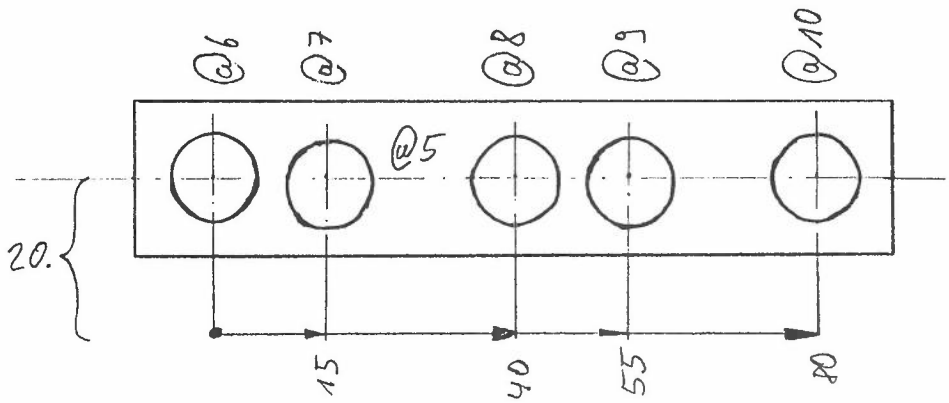
```
!*
-- preliminaries (see figure for definition) --

@4= ARC_SEGMENT
@5= ARC_SEGMENT
@6= LINE_SEGMENT
@7= CARTESIAN_TWO_COORDINATE ( assumed
    is the intersection point of edges
    #5 and #6 );
@8= CARTESIAN_TWO_COORDINATE ( assumed
    is the intersection point of edges
    #4 and #6 );
@9= CARTESIAN_TWO_COORDINATE ( assumed
    is the end point of arc #4 that is
    not #8 );
@10= CARTESIAN_TWO_COORDINATE ( assumed
    is the end point of arc #5 that is
    not #7 );
@11= TWO_SPACE_DIRECTION ( assumed is
    the direction of the arc tangent at
    #8 )
@12= TWO_SPACE_DIRECTION (  assumed is
    the direction of the arc tangent at
    #9 )
@13= TWO_SPACE_DIRECTION (  assumed is
    the direction of the arc tangent at
    #10 )

-- according to the drafting schema --

@1= NOTE (!* needs specification *!);

@2= NOTE (!* needs specification *!);

@3= NOTE (!* needs specification *!);


*!


@14= ARC_DIM_PAR (INITIAL_DIM_ATTRIBUTES( #12, #9 ), #1,

    .T., 25., #8, .T., #11 );


@15= LIN_DIM ( #14, #2, .T., 45., #7, .T. );


@16= ARC_DIM_RAD ( #15, #3,  , #10, #13 );
```

Example 5:
chain dimension



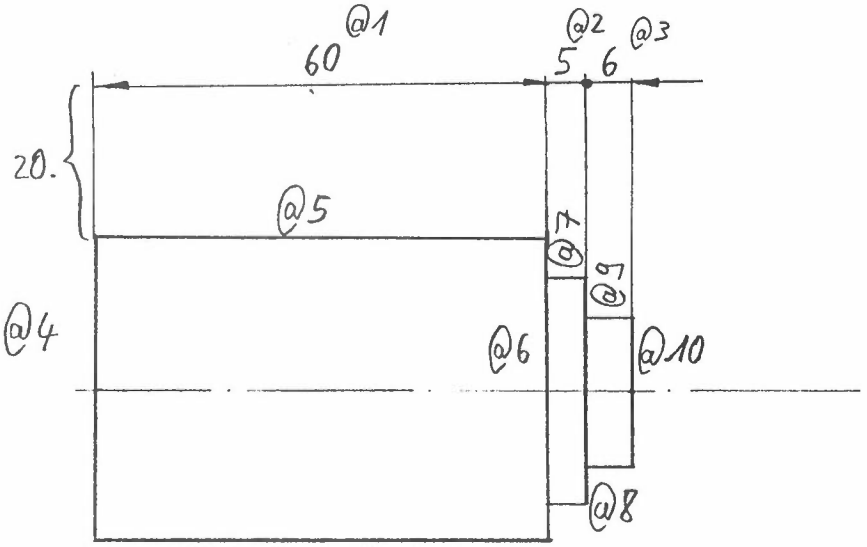

```
!*

-- preliminaries (see figure for definition) --

@5= LINE_SEGMENT    @6= LINE_SEGMENT    @7= LINE_SEGMENT
@8= LINE_SEGMENT    @9= LINE_SEGMENT
@10= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
     point of edges #5 and #6);
@11= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
     point of edges #6 and #9);
@12= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
     point of edges #9 and #7);
@13= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
     point of edges #7 and #8);
@14= TWO_SPACE_DIRECTION ( assumed is the x direction of
     edge #6);
@15= TWO_SPACE_DIRECTION ( as #14 - assumed is the direction
     of edge #9 minus 90o);

-- according to the drafting schema --

@1= NOTE ( needs specification );
@2= NOTE ( needs specification );
@3= NOTE ( needs specification );
@4= NOTE ( needs specification );

*!

@16= LIN_DIM (ANG_DIM (LIN_DIM (LIN_DIM
     (INITIAL_DIM_ATTRIBUTES (#14, #10), #1, .T., 40., #12,
     .T.), #2, .T.,  , #13, .F.), #4, .F., .F., -10., #12,
     .F., #15), #3, .T.,  , #11, .F.);
```

Example 6:
superimposed
running dimension




```
!*

-- preliminaries (see figure for definition) --

@5= LINE_SEGMENT
@6= CARTESIAN_TWO_COORDINATE    @7= CARTESIAN_TWO_COORDINATE
@8= CARTESIAN_TWO_COORDINATE    @9= CARTESIAN_TWO_COORDINATE
@10= CARTESIAN_TWO_COORDINATE
@11= TWO_SPACE_DIRECTION ( assumed is the x-direction of
     centerline #5);

-- according to the drafting schema --

@1= NOTE ( needs specification );
@2= NOTE ( needs specification );
@3= NOTE ( needs specification );
@4= NOTE ( needs specification );

*!

@12= LIN_DIM (LIN_DIM (LIN_DIM (LIN_DIM
     (INITIAL_DIM_ATTRIBUTES (#11, #6), #1, .T., 20., #7,
     .F.), #2, .T.,  , #8, .F.), #3,  .T.,  , #9, .F.), #4,
     .T.,  , #10, .F.);
```

Example 7:
chain dimension




```
!*

-- preliminaries (see figure for definition) --

@4= LINE_SEGMENT     @5= LINE_SEGMENT     @6= LINE_SEGMENT
@7= LINE_SEGMENT     @8= LINE_SEGMENT     @9= LINE_SEGMENT
@10= LINE_SEGMENT
@11= TWO_SPACE_DIRECTION ( assumed is the x-direction of
     edge #5);
@12= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
     point of edges #4 and #5);
@13= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
     point of edges #5 and #6);
@14= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
     point of edges #7 and #8);
@15= CARTESIAN_TWO_COORDINATE ( assumed is the intersection
     point of edges #9 and #10);

-- according to the drafting schema --

@1= NOTE ( needs specification );
@2= NOTE ( needs specification );
@3= NOTE ( needs specification );

*!

     @16= LIN_DIM (LIN_DIM (LIN_DIM (INITIAL_DIM_ATTRIBUTES
     (#11, #12), #1, .T., 20., #13, .T.), #2, .F.,  , #14,
     .T.), #3, .F.,  , #15, .T.);
```

1.8 The Neutral File Adapting System (NFAS). A Practical Approach to the Improvement of Data Exchange

1.8.1 Introduction

One great problem in transferring data between different CAD systems is the different representation of information in different systems. It is not sufficient to define a neutral file format which includes all representations of all systems, because most postprocessors do not interpret information represented in a form that does not match the form used in the receiving system. In figure 1 it can be seen that an extended standard improves the facilities of data exchange, but there will always be a loss of information.

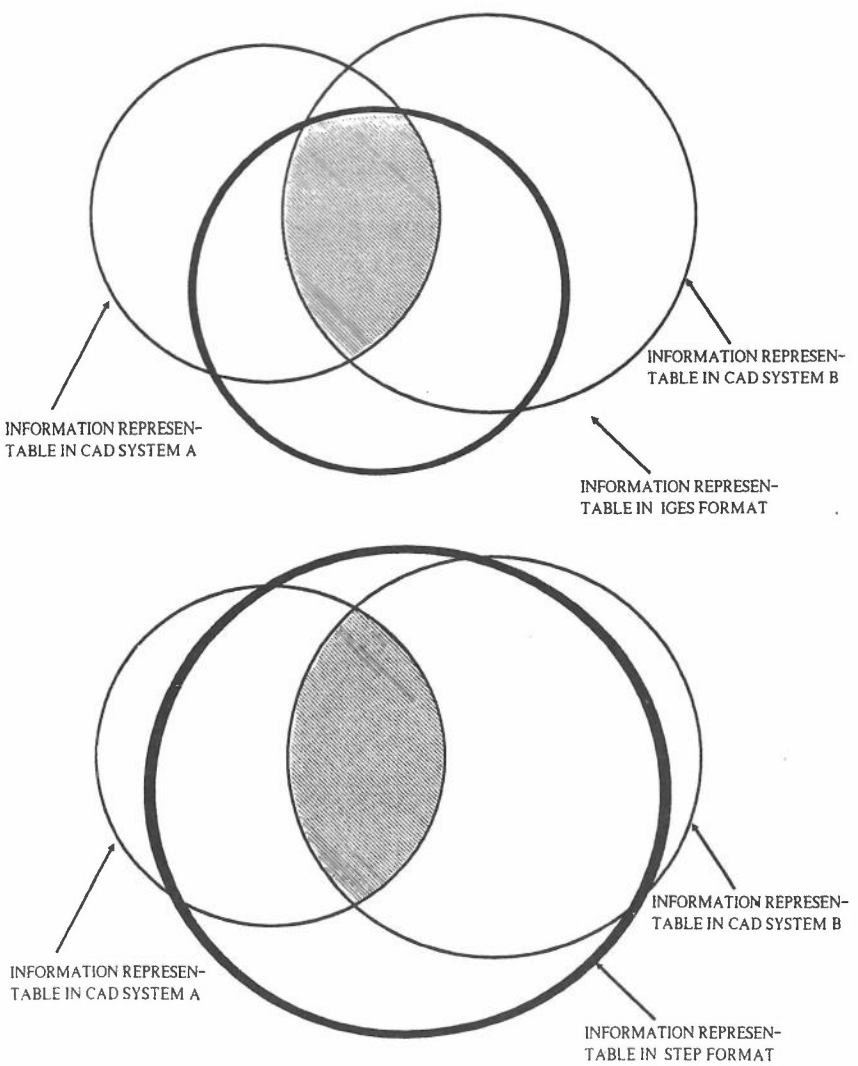


Fig. 1.8-1: Transferable information via CAD/CAM interfaces

The hatched area in figure 1.8-1 describes the transferable data with equal representation in CAD system A, CAD system B and the neutral format.

1.8.2 Adaption as a possible means of overcoming differences in data representation

The solution of the problem described above is the conversion from one representation of data into another. This can be done by a system dependent postprocessor or by a system independent adapting program, that works on the basis of a neutral format and can therefore be used for any combination of CAD/CAM systems which have the required interface processors.

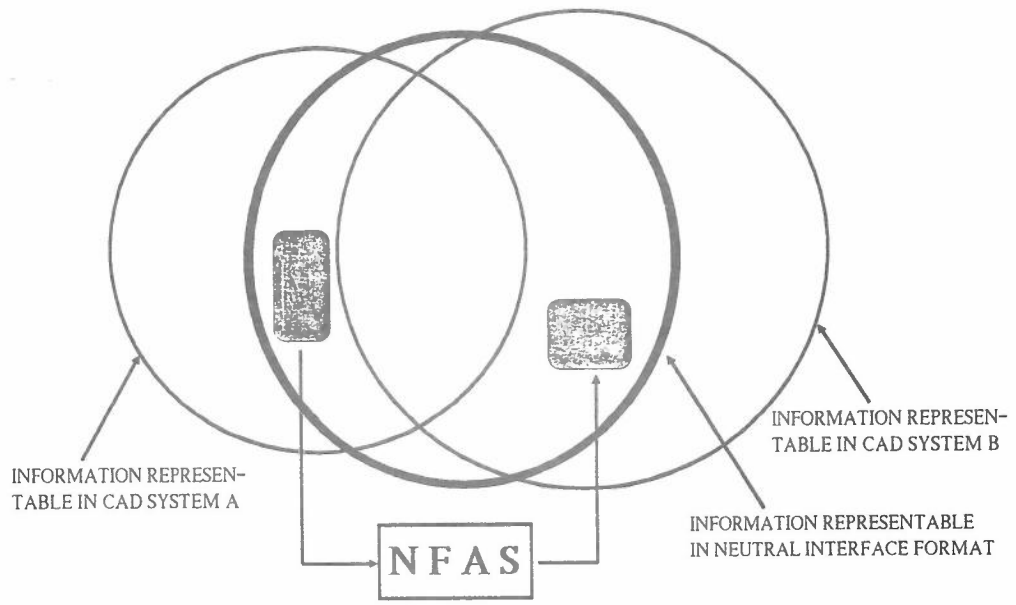


Figure 1.8-2: Necessity for neutral file adapting

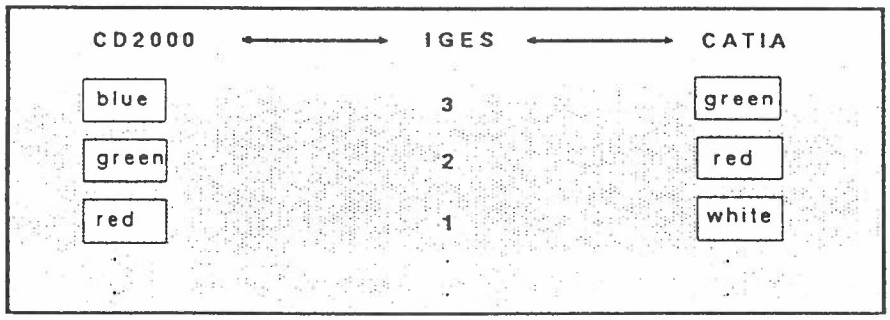
1.8.2.1 Examples for Practical Applications of a Neutral File Adapting System. Conversion of Attributes

NFAS perfomance

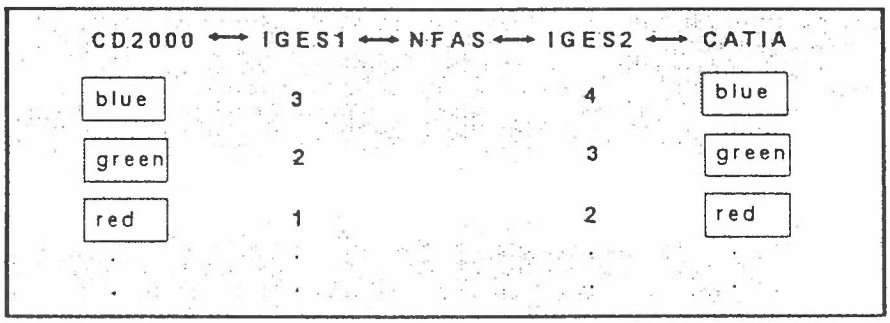
1. Conversion of attributes

Application : color conversion CD2000 / CATIA

without NFAS:



with NFAS:



Further possibilities:

Layer

Line font

Pen number

Visibility

Use

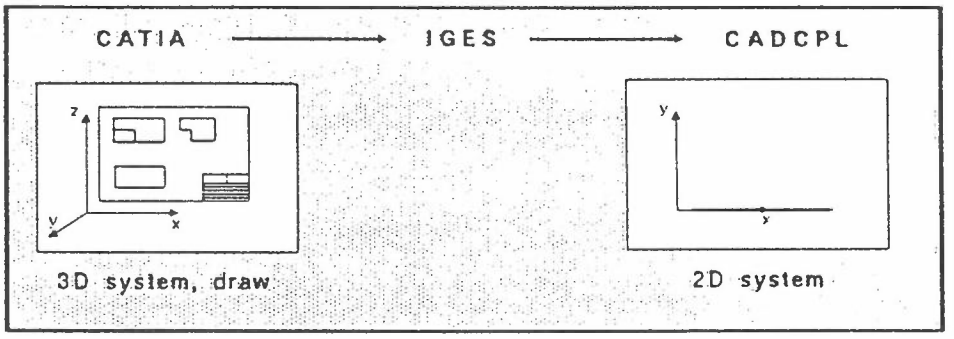
Example 1: Conversion of attributes

NFAS performance

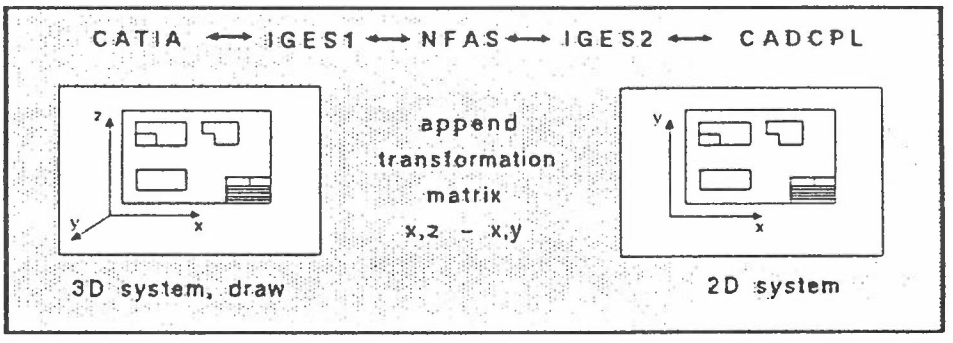
2. Geometry operations

Application : position alternation CATIA / CADCPL

without NFAS:



with NFAS:



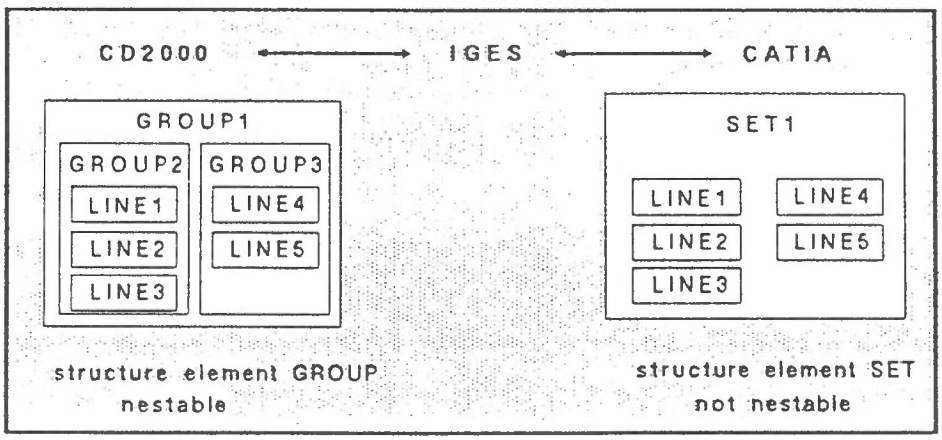
Example 2: Conversion of geometry

NFAS performance

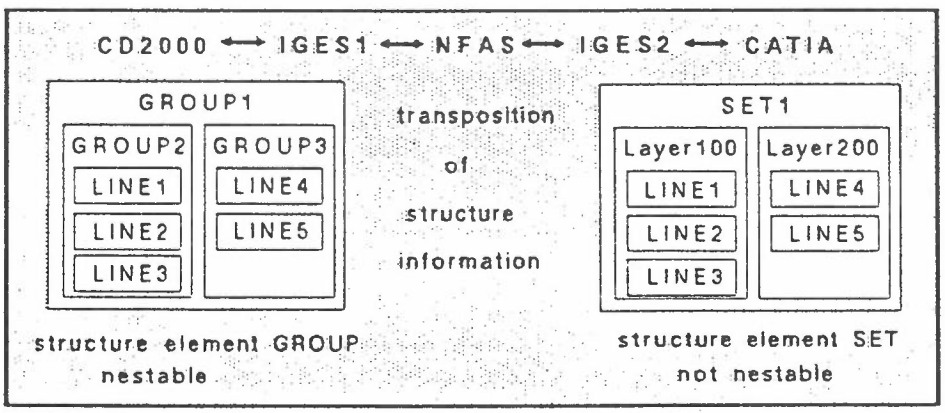
3. Structure operations

Application : structure handling CD2000 / CATIA

without NFAS:



with NFAS:

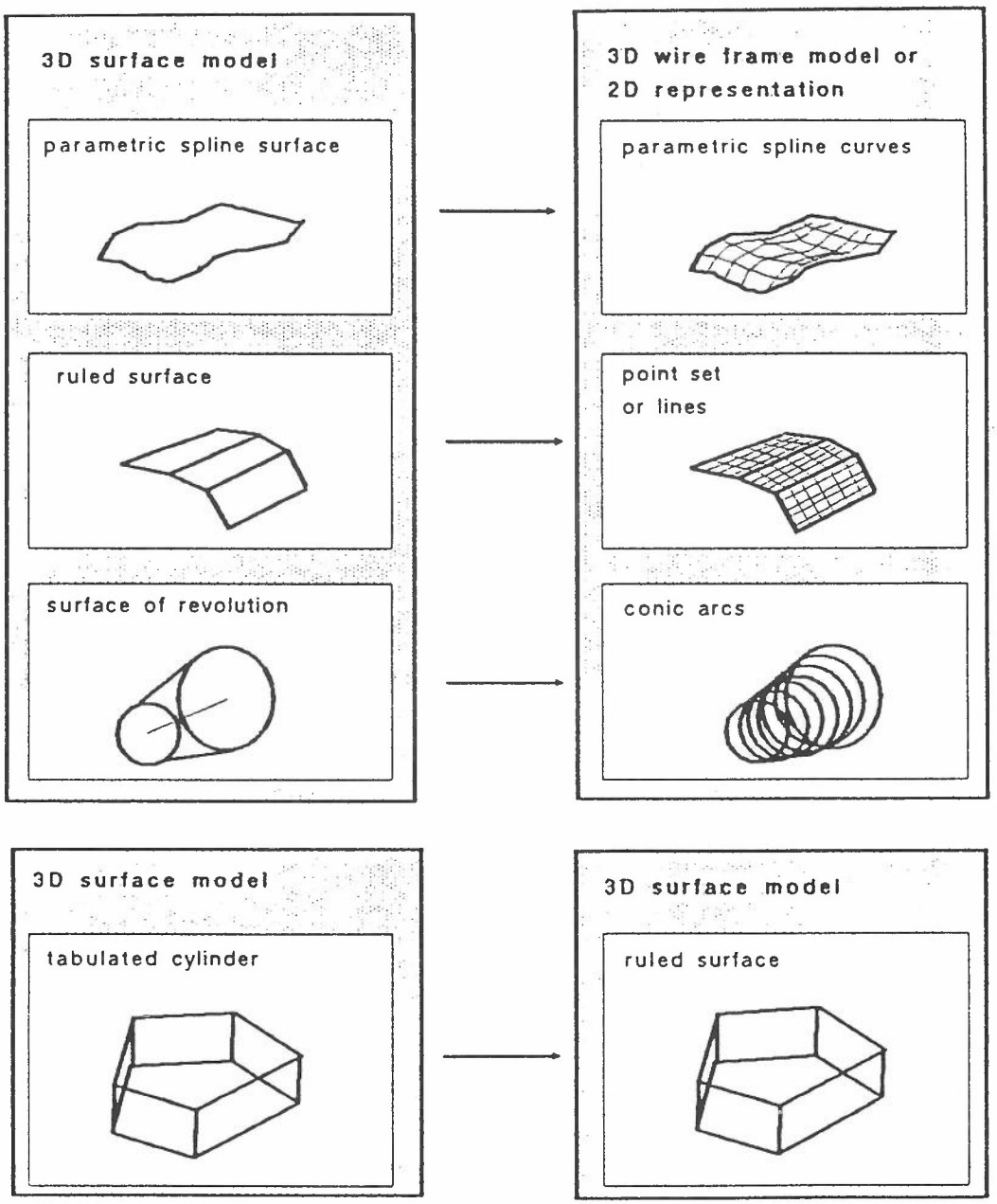


Further possibilities : attachment to
- colors
- line fonts
- pen numbers
- visibility

Example 3: Conversion of structures

NFAS performance

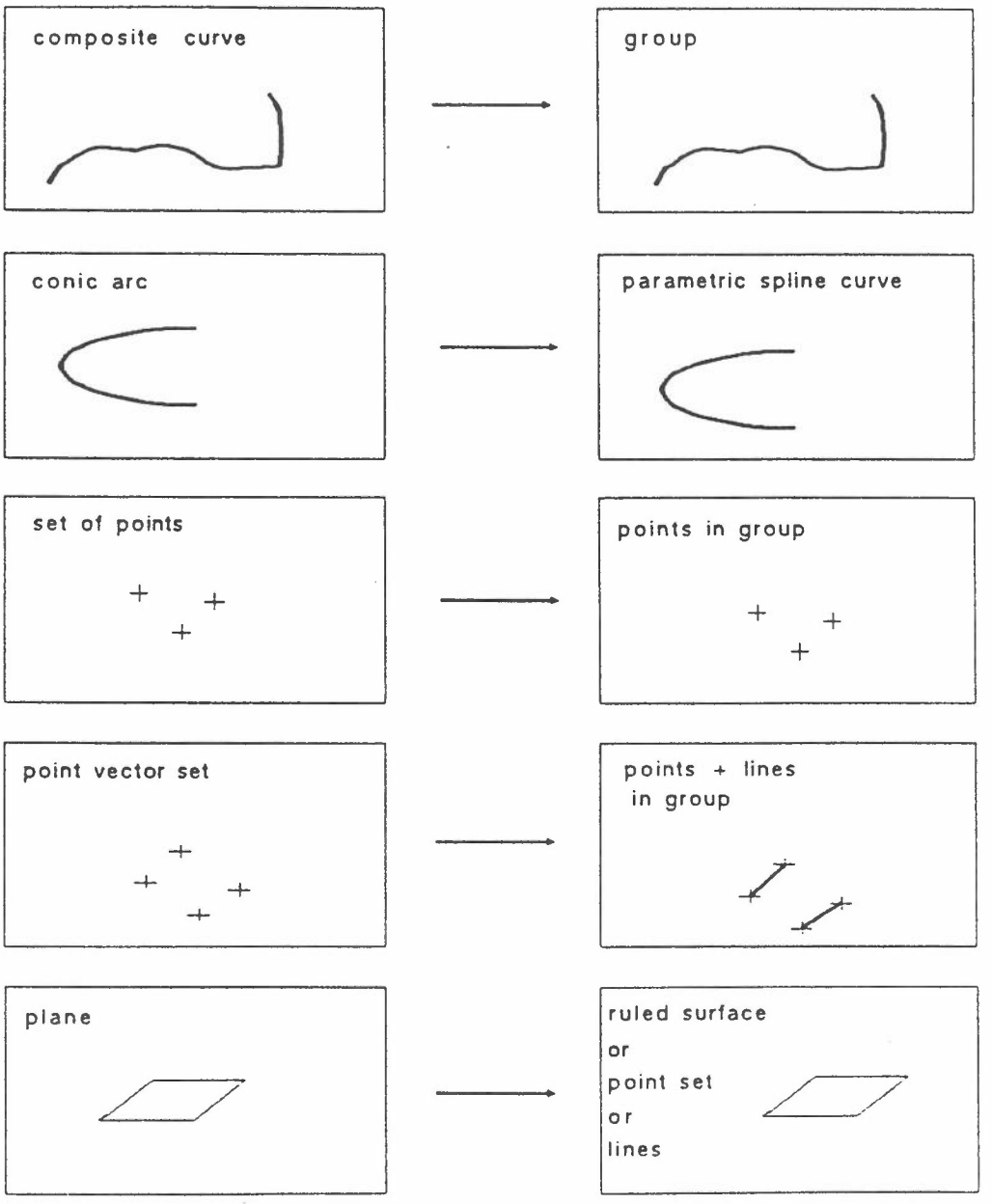
4. Type conversion



Examples for mapping of CAD model classes

NFAS performance

4. Type conversion



Examples for entity type conversion

1.8.2.2 Example of the Practical Application of a Neutral File Adapting System Based on the Format of IGES

In order to prove the possibilities of adapting neutral files, a system was developed that works on the basis of the IGES format.

System Structure

The system is divided into two subsystems:

- the application interface with control and processing routines for the adaption itself, so that this part can also be used for other neutral formats.
- and the call interface with information on the neutral file structure which allows a neutral file to be read and written.

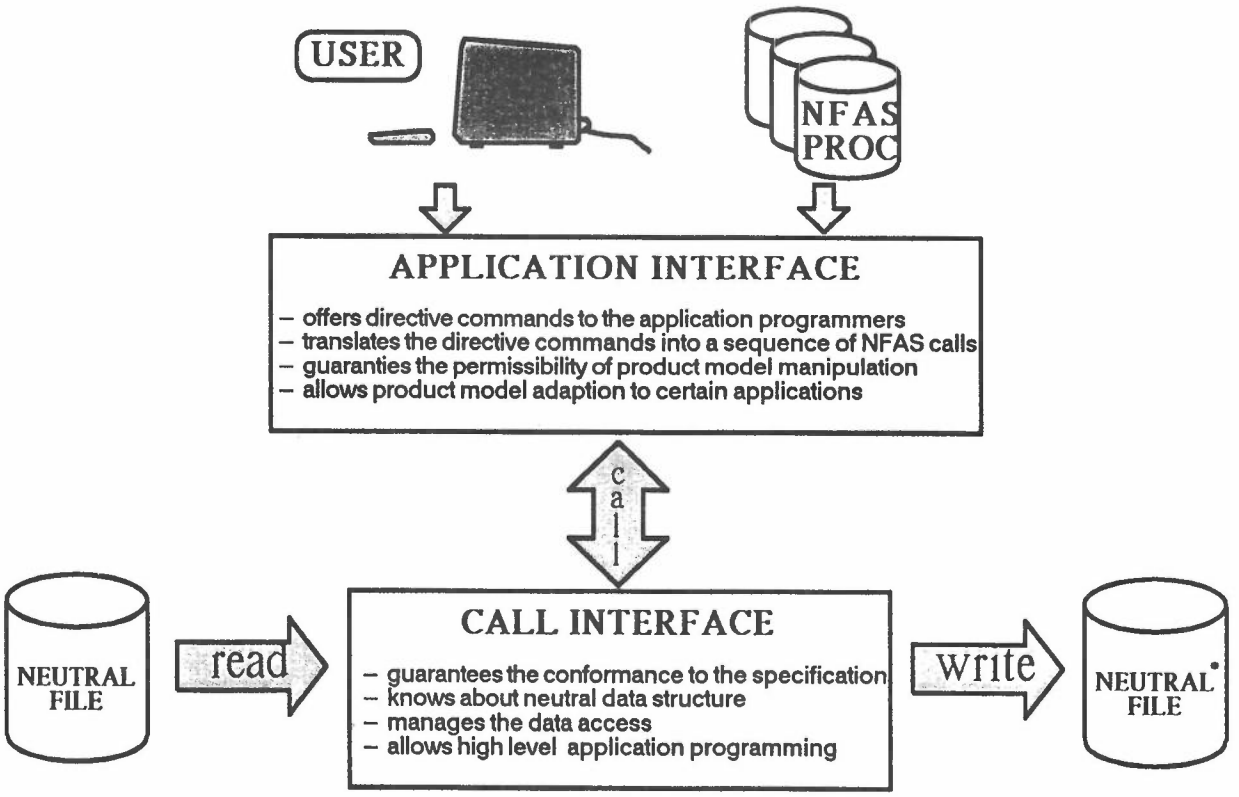


Figure 1.8-3: Conception of the Neutral File Adapting System

Application Interface

The application interface is a subsystem of NFAS including control and processing routines for the adaption. It is not dependent on a specific neutral format.

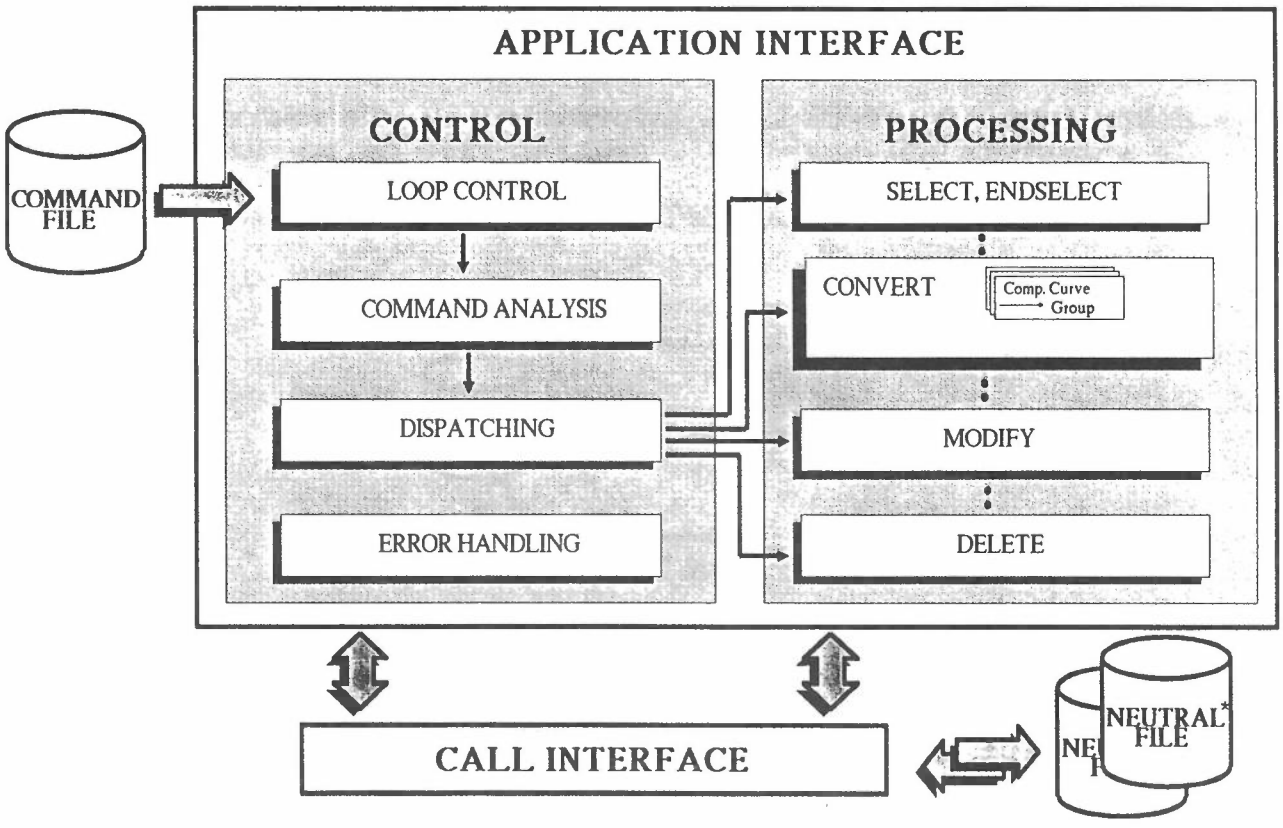


Figure 1.8-4: Neutral File Adapting System

The application interface works with a directive file as input. Therefore it is possible to run the adaption without interactive input in the background.

The aim of using directives was to define a user friendly simple language by a set of words and parameters. Figure 5 shows an overview of the possible combinations of keywords and parameters.

Command	Parameter	Description
PROTOCOLLTOLABEL	YES | NO	Generation of a protocoll
SELECT ENDSELECT	[TYPE EQ <enttyp>] | [<attrib> EQ <value>]	Definition of a target set
CONVERT	[TYPE:=<enttyp>]	Conversion into entity type
MODIFY	<modatt>[start]:=<value> | <modatt>:=<refatt>	Modification of attributes
NORMALIZECONICARC		Conversion of conics into normal form
DELETE		Deletion of entities
TREEDELETE		Deletion of entities inclusive all subordinates
MAKEMEMBERS	[<attrib>[start]:=<value>]	Generation of a group
MARKEMEMBERS	[<attrib>[start]:=<value>]	Mark of members in structures
MAKEDISJUNCT		Copy of multi referenced entities

Figure 1.8-5: Examples of directive commands

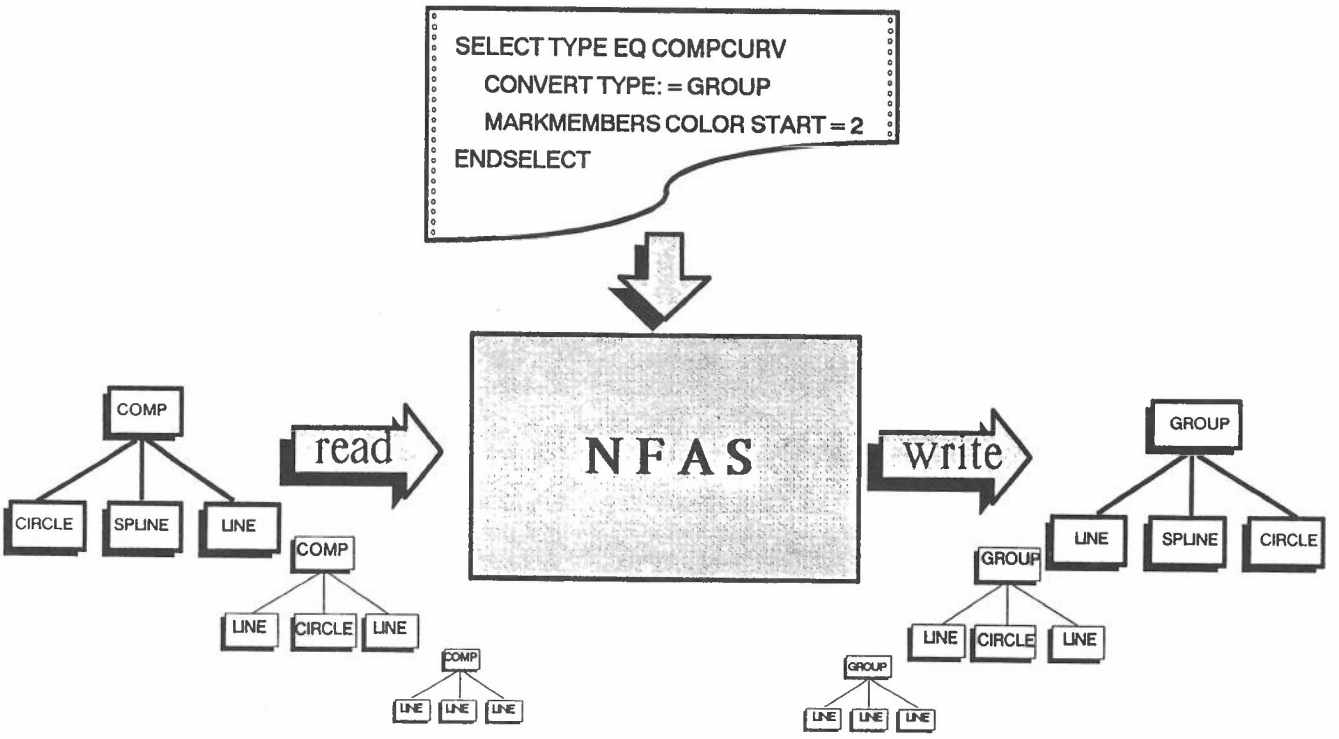


Figure 1.8-6: Example for Neutral File Adapting. Application with the corresponding directive file

1.8.2.3 Status of NFAS

The actual status of the developed system, working on the basis of an IGES call interface is as follows:

- modification of attributes
- modification of character strings in general note entities (212)
- type conversion between group (402) and subfigure (308/408) in both directions
- translation of geometry
- rotation of geometry round x-, y-, z-axis
- deletion of entities
- deletion of trees with dependent entities
- generation of groups (402)
- disjunction of groups

All these action functions are executed for the whole neutral file, or a selected subset of the neutral file. The possibilities of selection of subsets were also extended. The actual status is:

- selection by attributes as:
  type/form number or name, line font pattern, level number, blank status, entity use flag, line weight, colour and combi- nations

- selection of dependent entities by conditions (like above) of parent entities.

1.8.3 Future Development

After the development of this basic performance, the system will now be optimized concerning running time and storage conditions. First experiences in productive environment have shown, that the system is not only usable for compensation of system differences, but also for application dependent adaptions.

1.9 Experience Gained Using the Neutral File Adapting System (NFAS)

1.9.1 Introduction

CAD/CAM Systems are specialised for certain applications even if they are constituted by several specific moduls. Therefore problems occur while exchanging data between them to realise an industrial production process. A lot of these problems do neither depend on the documentation of the data interface nor on the implementations of the processors. But they are caused by the principle differences of CAD/CAM systems and the special requirements of the applications for which the data are intended.

Figure 1.9-1 shows the expected advantages of neutral file adapting.

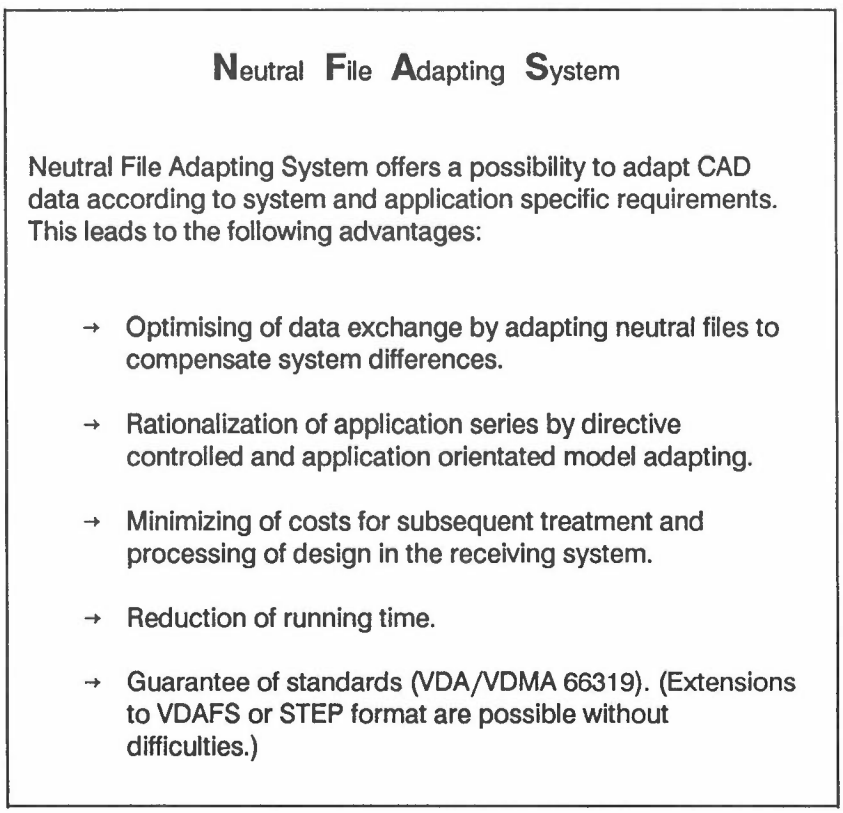


Fig. 1.9-1: Principle advantages of Neutral File Adapting

1.9.2 The Scope of NFAS

Firstly, NFAS has been intended to optimise the data exchange between CAD/CAM systems of different capabilities like:

- different geometric models
  - . 2D wireframe model
  - . 3D wireframe model
  - . surface model
  - . solid model

- use of synonymous data that cannot be easily mapped onto each other
  - . rational B-spline <-> polynomial curve
  - . group <-> set
- use of homonymous data that must be mapped to keep the original meaning. Especially this is the different semantic of used
  - . levels
  - . colours
  - . groups
  - . ...

Fig. 1.9-2 shows a specific use of NFAS for system adaption.

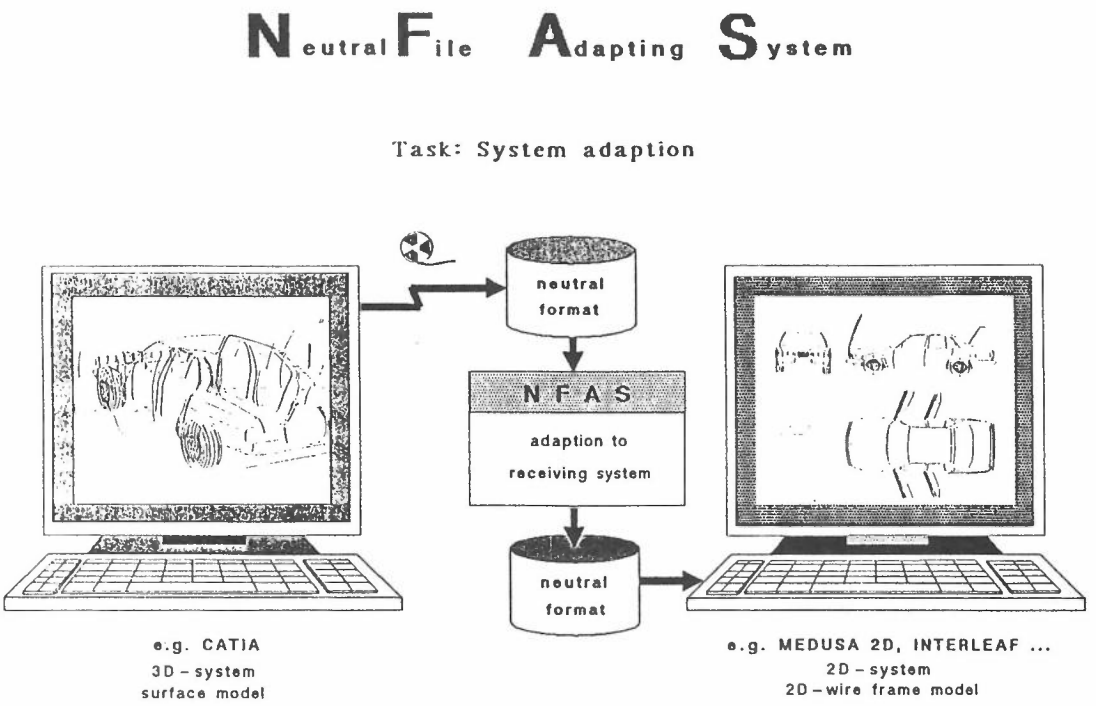


Fig. 1.9-2: Example for a system adaption by NFAS

Secondly, NFAS will be used to modify neutral files according to the requirements of the application receiving the data. First experiences show that this capability of NFAS offers a wide area of usage to it. Examples for such adaptions are:

- selection of relevant data by filtering the content of the neutral file
- manipulation of the data by applying operations such as projection, scaling, rotation, etc. to achieve an external representation that is adequate for the application.

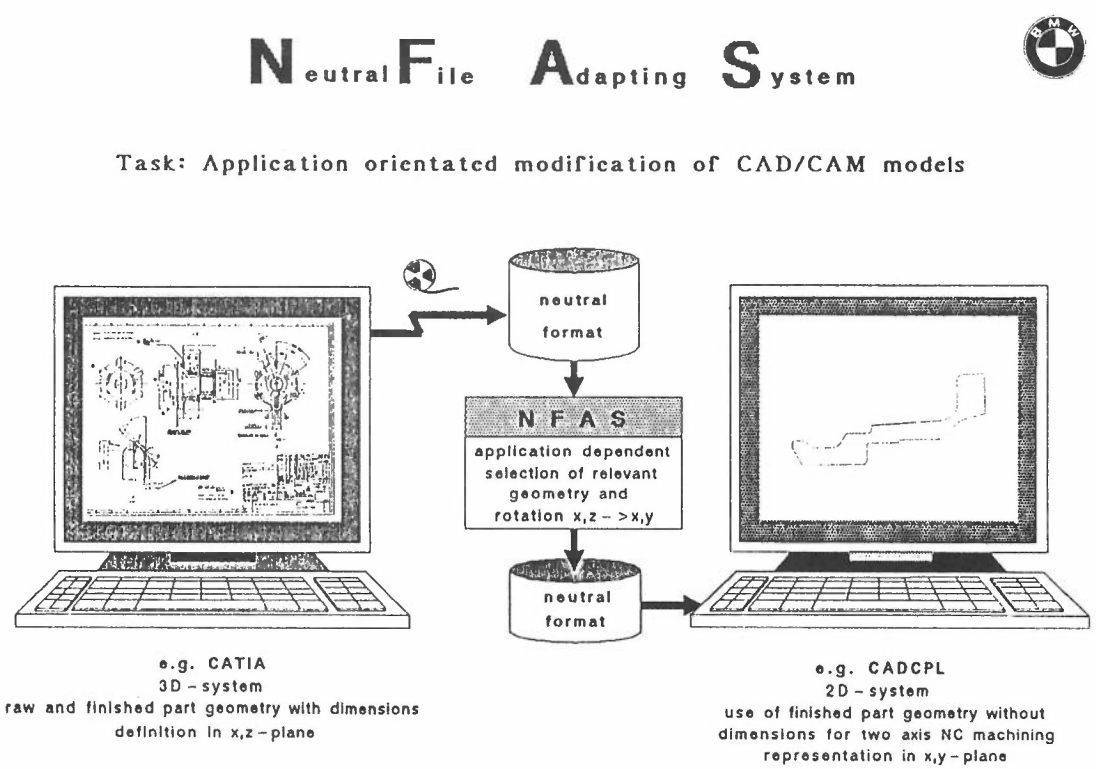


Fig. 1.9-3: Example for an application adaption by NFAS

1.9.3 Using NFAS

For a first test of NFAS under productive conditions the application chain: "part design -> NC-programming" was selected. To establish this link at BMW the 3D-CAD system CATIA and the 2½D CAM-system CADCPL (EXAPT) had to be connected via the IGES interface.

At this following points had to be considered:

- projection of 3D-geometry onto the x-y plane
- different implementation of the view mechanism in the CATIA IGES and the CADCPL-IGES-processors
- enormous size of CATIA-models which have to be transferred to CADCPL

Up to using NFAS, the realisation of this application chain (see Fig. 4) required the interactive creation of CATIA submodels to achieve the points mentioned above.

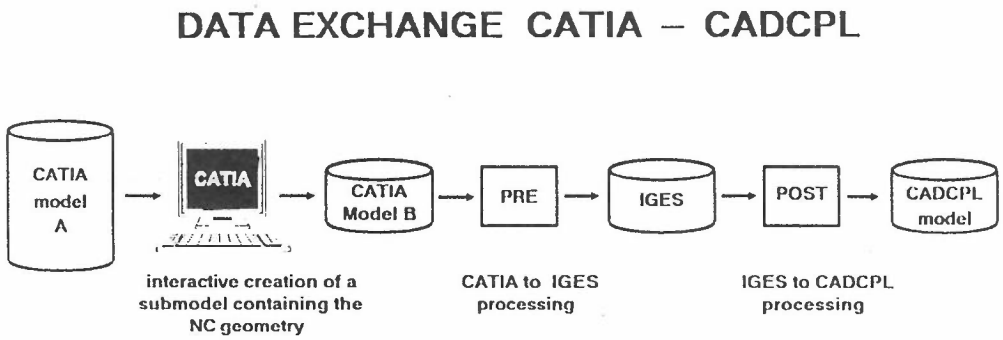


Fig. 1.9-4: CAD data flow without NFAS

This kind of proceeding did not provide a suitable solution. Main disadvantages were:

- effort for interactive submodel creation
- production of redundancy (in CATIA) that must be maintained
- incomplete adaption to the receiving system in the sending system

On the other hand, the adaption using NFAS proceeds as shown below and avoids the disadvantages mentioned above.

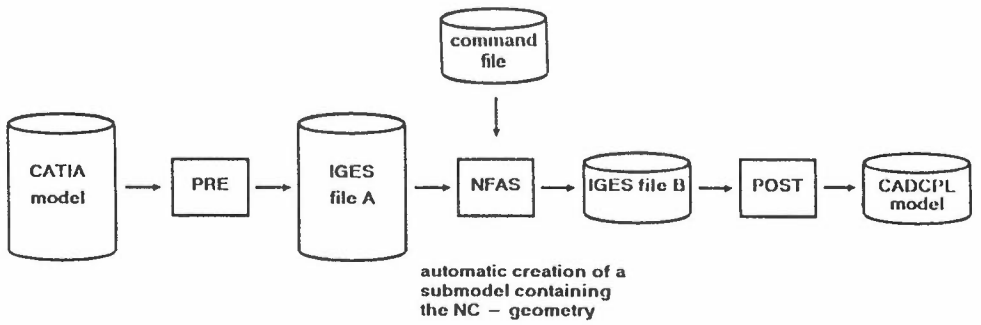


Fig. 1.9-5: CAD data flow using File Adaption

For instance, a specific application of NFAS shall be described by the following sequence of figures:

1) The original CAD model in CATIA containing the shape that is necessary for the NC-programming

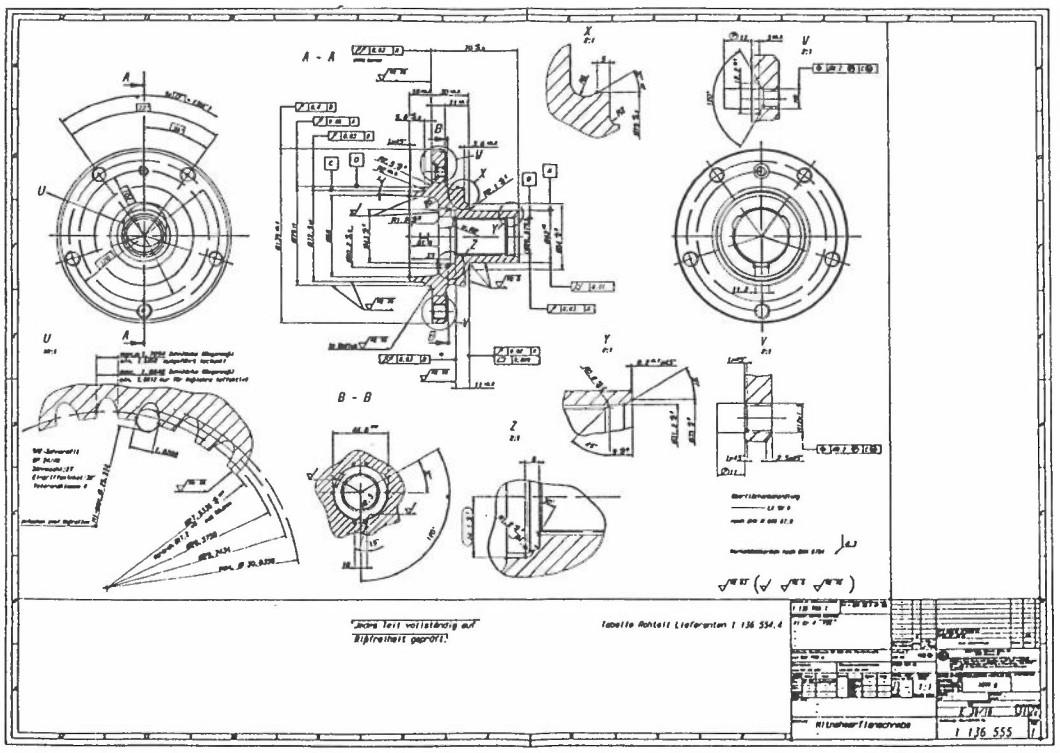


Fig. 1.9-6: Original CAD model

2) The command file that was edited for this application controls the file adaption

```
!*************************************************************!
! NFAS DIREKTIVE:    CADCPL                                   !
!*************************************************************!
!      Diese Direktive dient zur Selektion von Konturen für   !
!      die Drehbearbeitung aus komplexen CATIA Modellen zur   !
!      Weiterverarbeitung und NC-Programmierung mit CADCPL.   !
!                                                             !
!               CATIA V2R2 PTF4   Prozessor: V2R2 PTF4        !
!                                                             !
!               CADCPL V5.9       Prozessor: V2.0             !
!*************************************************************!
! Historie:                                                   !
!                                                             !
! erstellt von: H. Scheder FI-100           am: 06-10-1988    !
!                                                             !
!*************************************************************!
!                                                             !
!-------------------------------------------------------------
! Selektion und Löschen der nicht relevanten Layer
!-------------------------------------------------------------
  SELECT LEVEL.NE.20
    TREEDELETE
  ENDSELECT
!-------------------------------------------------------------
! Selektion und Löschen der Bemassung und Texte
!-------------------------------------------------------------
  SELECT TYPE.EQ.202.OR.TYPE.EQ.206.OR.TYPE.EQ.210.OR.TYPE.EQ.212
    TREEDELETE
  ENDSELECT
  SELECT TYPE.EQ.106.AND.FORM.EQ.31
    DELETE
  ENDSELECT
!
!-------------------------------------------------------------
! automatische Transformation der Geometrie nach x-y
!-------------------------------------------------------------
  SELECT
    TRANSFORM AUTO XY
  ENDSELECT
```

Table 1.9-1: NFAS command file

3) The result of this adaption allows the user to proceed with the NC-programming without subsequent treatment.

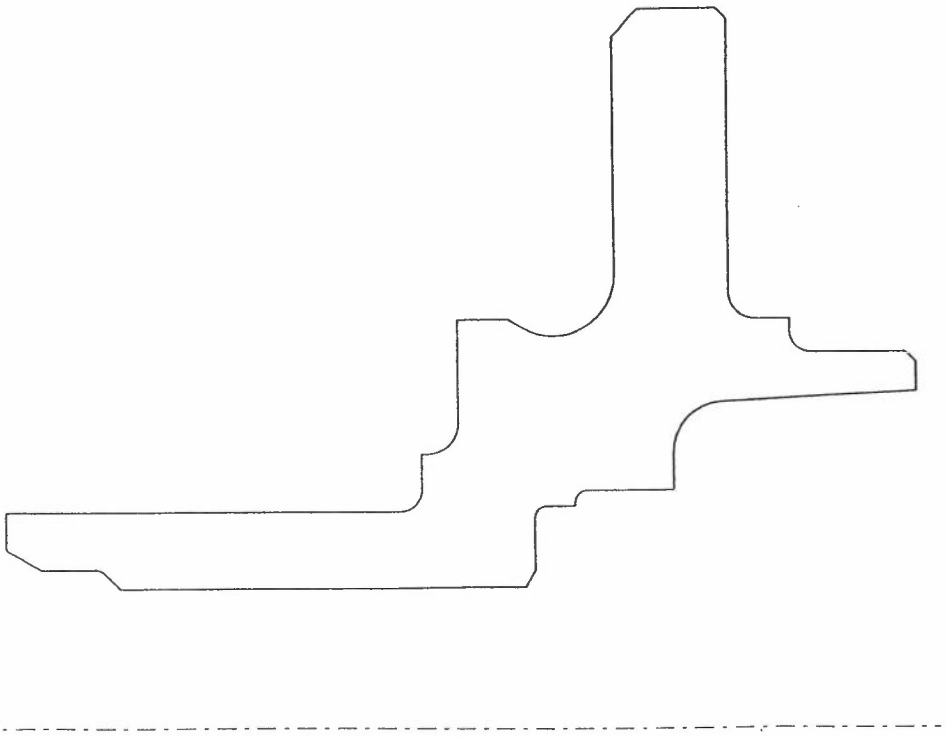


Fig. 1.9-7 Received CAD model after application adaption

1.9.4 Summary

The experience gained using NFAS shows the following advantages:

- avoidance of redundant CATIA models
- avoidance of pre- and subsequent treatments
- minimising of errors
- minimising of information losses.

It was confirmed that the existence of a detailed model structure is decisive for a good result of neutral file adapting.

1.9.5 Outlook

Due to the success of NFAS it is planned to use it for links between the various CAD/CAM systems of the suppliers and BMW. Also the CAD system ROBCAD shall be connected with CATIA via NFAS. The multi-purpose NFAS is expected to be used for optimising the data exchange of further applications.

More work has to be done in the area of CAD model structuring and product model definition as assumption for complete data transfer in the future.

1.9.6 References

/1/ E.G. Schlechtendahl, ed.
Specification of a CAD*I Neutral File for CAD Geometry. Wireframes, Surfaces, Solids. Version 3.3. Research Reports ESPRIT. 3rd, Revised Edition. Springer-Verlag, 1988.

/2/ German
Contribution to Drafting Model Standardization, ISO Document Number N 252, 1988

2. Working Group 2: Solids

One of the goals of the ESPRIT project CAD*I (CAD Interfaces) is to develop techniques for the exchange of CAD information between CAD systems. Working Group 2 (WG2) in the project is mainly concerned with Solid Modelling. As solid modelling is based upon curves and surface information in many respects, WG2 also has the responsibility to integrate wireframe and surface representation where appropriate.

This part of the Status Report documents the results achieved by CAD*I Working Group 2 (Solids) in the fourth year of the project. The principal achievements of the first three years have been documented elsewhere /1,2,3/ and are summarized here briefly.

- The development of an overall metafile format which has been accepted by the CAD*I project as a whole.

- A file format for descriptive text (called "letters") which is to be transmitted together with CAD data on the same file. Provisions have been made to allow for all European national characters in such letters.

- Identification of the possible methods of interfacing CAD systems to a neutral file.

- Textual description of the overall structure of pre- and post-processors.

- The development of a formal language for defining CAD system data structures (HDSL = high level data specification language).

- Formalised specification of all syntax elements on the neutral file (alphabet, alphabet extensions, tokens, statements, and entity/property representations).

- Documentation and publication of complete and consistent CAD*I specifications for geometry.

- Development of pre- and post-processors according to these specifications.

- Performance of the cycle tests and intersystem tests with solid models of CSG, B-rep, and polyhedron type on a routine basis.

- Influencing the international standardization activities in order to make the future standards conform with essential elements of the CAD*I specification.

The essential results of the fourth project year are:

- Conversion of the processors to version 3.2 of the specification.

- Development and publication of version 3.3 of the specification.

- Enhancement of all processors to version 3.3.

- Begin and completion of a CAD*I pre-processor for Catia (polyhedron models and sweep primitives).

- Development of a formal specification of the semantics of CSG models and CSG-based systems using the methods of algebraic data types.

- Completion of the processor development for elementary solid models for most systems. Emphasis has shifted to advanced capabilities like parametric modelling, instancing, and external references. The processors for all but one system (Geomod) have reached the state of fully operational prototypes. This was the project goal with respect to processor development. Documentation has begun.

- Beginning exploitation of the techniques and the processors developed outside the CAD*I project.

2.1 Specification

The specification version 3.2 /4/ of a neutral file for CAD geometry had been published in year 3 of the project. During the first months of project year four, a number of enhancements and corrections were introduced as a consequence of feedback from processor development. However, there were also three more important changes necessary in the syntax resulting from the first transfer of solid model data into the finite element domain. The FEM neutral file syntax is more restrictive than the one allowed for geometry files. These changes required a new version of the specification (version 3.3) to be defined. The resulting specification version 3.3 /5/ is now published and constitutes the basis for all processor developments.

2.2 Semantics of the CAD*I data structures

Several semantic levels have to be distinguished:

1) The semantics of the neutral file which is to define the reconstruction of a data structure that conforms to the CAD*I reference schema.

- This level of semantics is defined in terms of a finite state machine in /5/.

2) The semantics of the constituent elements of the specification language HDSL, which is to define the effect of low level operations like

   - enter scope
   - leave scope
   - create entity, property
   - identify entity, property
   - pass entity, property to some application function
   - inquire attributes in the context of the HDSL terminology
   - ENTITY
   - PROPERTY
   - ATTRIBUTE
   - SCOPE

   This level of semantics is also specified in /5/. A part of that specification has become an integral portion of the STEP standard document on the physical file structure.

3) The semantic meaning of the data-structures on the information level which is to define what the data actually means and what can be done with that information.

   This level of semantics is formally spelled out in the CAD*I specification as far as curves and surfaces are concerned. The method uses common mathematical notation for vectors and parametric representations.

   For solids, however, there is no equivalent formal representation. In order to develop an unambiguous formal representation for solids various specification techniques were investigated. The most promising technique available in computer science today is based on "algebraic data structures". This method is now being utilized for specifying the semantics of CSG models and is outlined briefly in the following chapters.

2.3 Algebraic specification of CSG

In the past, it has become evident that the specification of CAD-modellers /6/ and thus the specification of CAD-entities using the natural language leads to misunderstandings. In order to avoid these misunderstandings formal specification methods have to be used. One of these methods - the algebraic specification - allows a unique specification of CAD-modellers (CAD-entities) and abstracts from the implementation of these objects. The basic idea of the algebraic specification is, that formally a CAD-modeller is a many-sorted algebra and is specified by a hierarchical abstract type. In the CAD*I project investigations have showed, that a CAD-modeller can be specified by a special hierarchical type: hierarchical object specification /6/. The idea is, to separate the operator sets of a hierarchical abstract type T in constructors, attributes, relations and modificators and the sort-set of S in entities and values. Nevertheless, one has to make some restrictions to a hierarchical object-specification: unique, sufficiently complete, consistent, based on special Horn-clauses as axioms, regular, and primitive-recursive /7/.

The following system of object type specifications describes CSG solids. For the sake of simplicity, the set of primitives consists only of boxes, spheres, cylinders and cones. The abstract type STANDARD specifies an incomplete but (for this example) sufficient modelling concept for the attribute values of CSG solids. Besides, values of sort real, values of sort bool and values of sort string are models of the standard modeltypes REAL, BOOL, and STRING. The specification language is an extension of CIP-L. REAL and BOOL terms are written in infix notation (i.e. a+b stands for +(a,b)).

```
value STANDARD

    sort real, bool, string

    funct false   : bool,
    funct true    :  bool,
    funct and     : (bool,bool) bool,
    funct or      : (bool,bool) bool,
    funct not     : (bool) bool,
    funct +       : (real,real) real,
    funct –       : (real,real) real,
    funct ×       : (real,real) real,
```

funct ÷ : (real,real) real,
funct Sqrt : (real) real,
funct Qrt : (real) real,
funct = : (real,real) bool,
funct ≤ : (real,real) bool,
funct $=_w$: (string,string) bool

end of value

For b≡TRUE and b≡FALSE we write only b and not(b). The operator sqrt denotes the real square root function, Qrt the real square function and ÷ the real division. For sqrt(x) we write $\sqrt{x}$, for qrt(x) we write x^2.

The Euclid Space

The euclid space is a 3-dimensional vector space with the euclid norm $\|\langle x,y,z\rangle\| = \sqrt{(x^2+y^2+z^2)}$. The following object specification is not complete but for this example sufficient.

object EUCLID

based on VALUE

sort euclid

cons P : (real,real,real) euclid,
modi Add : (euclid,euclid) euclid,
modi Sub : (euclid,euclid) euclid,
modi Inv : (euclid) euclid,
attr DX : (euclid) real,
attr DY : (euclid) real,
attr DZ : (euclid) real,
attr Abs : (euclid) real,
rela Le : (euclid,euclid) bool

axioms all (p_1, p_2, p_3, r_1, r_2, r_3:**real**, pk_1 ,pk_2:**euclid**)

$\mathrm{Add}(\mathrm{P}(p_1,p_2,p_3),\mathrm{P}(r_1,r_2,r_3)) \equiv \mathrm{P}(p_1+r_1,p_2+r_2,p_3+r_3)$,

$\mathrm{Sub}(\mathrm{P}(p_1,p_2,p_3),\mathrm{P}(r_1,r_2,r_3)) \equiv \mathrm{P}(p_1-r_1,p_2-r_2,p_3-r_3)$,

$Inv(P(p_1,p_2,p_3)) \equiv P(-p_1,-p_2,-p_3)$,

$DX(P(p_1,p_2,p_3)) \equiv p_1$,

$DY(P(p_1,p_2,p_3)) \equiv p_2$,

$DZ(P(p_1,p_2,p_3)) \equiv p_3$,

$Abs(P(p_1,p_2,p_3)) \equiv \sqrt{(p_1)^2 + (p_2)^2 + (p_3)^2}$,

$Le(P(p_1,p_2,p_3),P(r_1,r_2,r_3)) \equiv (p_1 \leq r_1$ and $p_2 \leq r_2$ and $p_3 \leq r_3)$

end of object

One can show, that the axiom set of EUCLID consists only of primitive-recursive equations. So EUCLID is sufficiently complete, consistent and semantically unique. Therefore the existence of initial and final models of EUCLID is guaranteed /8/.

Algebraic Specification of CSG Primitives

For this example the primitives are boxes, that are bounded by the xy-plane, yz-plane and the xz-plane of the euclid space, spheres (with arbitrary midpoint), cylinders, and cones.

object PRIMITIVE

based on VALUE, EUCLID

sort prim,
cons Kugel : (euclid,real) prim,
cons Quader : (euclid) prim,
cons Zylind : (real,real) prim,
cons Kegel : (real,real) prim,
rela Defined : (prim) bool,
rela Prim_El : (euclid,prim) bool

axioms all (r,s:**real**, d,x,s:**euclid**, p,q:**prim**)

Prim_El(x,Kugel(m,r)) ≡ **if** Defined(Kugel(m,r)) **then**
Abs(Sub(m,x))≤r **else** false,

Prim_El(x,Quader(d)) ≡ **if** Defined(Quader(d) **then**
(Le(P(0,0,0),x) and Le(x,d)) **else** false,

Prim_El(x,Zylind(r,s)) ≡ **if** Defined(Zylinder(r,s) **then**
($DX^2(x) + DY^2(x) \leq$ r sup 2 and DZ(x)≤s) **else** false,

Prim_El(x,Kegel(r,s)) ≡ **if** Defined(Kegel(r,s) **then**
($DX^2(x) + DY^2(x) \leq (r - (\frac{r}{s} \times DZ(x)))^2$ and DZ(x)≤s) **else** false,

Defined(Kugel(m,r)) ≡ (r > 0),

Defined(Quader(d)) ≡ Le(P(0,0,0),d),

Defined(Zylind(r,s)) ≡ (r > 0 and s≠0),

Defined(Kegel(r,s)) ≡ (r > 0 and s≠0)

end of object

One can show that the axiom set of PRIMITIVE is a primitive-recursive set of equations. So PRIMITIVE is sufficiently complete, consistent and semantically unique. The relation Defined was only introduced to get total constructors for PRIMITIVE.

The terms (r>0) and Le(P(0,0,0),d) of sort bool are the context conditions of PRIMITIVE and are called static semantic. The sufficient completeness of PRIMITIVE guarantees that the static semantic of PRIMITIVE is decidable. For the processor development, this means that the static semantic of PRIMITIVE can be evaluated and checked by the CAD processors.

Algebraic Specification of CSG Solids

In this case, a CSG solid consists of objects of type PRIMITIVE. The following object specification is self-embedded, and so objects of type CSG can not be declared in a object-oriented database.

object CSG

based on VALUE,EUCLID,PRIMITIVE

sort csg
cons Empty : csg,
cons Constant : (prim) csg,
cons Union : (csg,csg) csg,
cons Differ : (csg,csg) csg,

cons Inters : (csg,csg) csg,
cons Transl : (euclid,csg) csg,
rela Csg_el : (euclid,csg) bool

axioms all (x,y:**euclid**, c,c_1,c_2:**csg**, p:**prim**)

Csg_el(x,Empty) ≡ false,

Csg_el(x,Constant(p)) ≡ Prim_el(x,p),

Csg_el(x,Union(c_1,c_2)) ≡ (Csg_el(x,c_1) or Csg_el(x,c_2)),

Csg_el(x,Inters(c_1,c_2)) ≡ (Csg_el(x,c_1) and Csg_el(x,c_2)),

Csg_el(x,Differ(c_1,c_2)) ≡ (Csg_el(x,c_1) and ¬Csg_el(x,c_2),

Csg_el(x,Transl(y,c)) ≡ Csg_el(Sub(x,y),c)

end of object

The set of axioms of CSG is primitive recursive. So CSG is sufficiently complete, consistent and semantically unique. CSG is not syntactically unique.

Algebraic Specification of CSG Environments

Since the type CSG is self embedded, it is not possible to declare the type CSG in the object-oriented database R^2D^2. Therefore, we specify the types CSG_TERM, CSG_EXPRESSION, CSG_ENTITY and CSG_ENVIRONMENT.

object CSG_TERM

based on VALUE,EUCLID

sort csg_term
cons Prim_const : (prim) csg_term,
cons Identifier : (string) csg_term,
cons Plus : (string,string) csg_term,
cons Minus : (string,string) csg_term,
cons Mult : (string,string) csg_term,
attr Term_el : (euclid,csg_term) bool

axioms all (p:**prim**, x:**euclid**, i,j:**string**)

```
        Term_el(x,Prim_const(p)) ≡ Prim_el(x,p),

        Term_el(x,Identifier(i)) ≡ false,

        Term_el(x,Plus(i,j)) ≡ false,

        Term_el(x,Minus(i,j)) ≡ false,

        Term_el(x,Mult(i,j)) ≡ false

    end of type
```

One can show, that the axiomset of CSG_TERM is primitive recursive. Therefore CSG_TERM is sufficiently complete, consistent and semantically unique. Furthermore, all constructors of CSG_TERM are regular. So CSG_TERM can be declared in R^2D^2 /9/.

```
    object CSG_ENTITY

        based on STANDARD,CSG_TERM

        sort db_entity

        cons Assign   : (string,csg_term) csg_entity,
        attr Ident    : (csg_entity) string,
        rela Get      : (csg_entity) csg_term

        axioms all (i:string, t:csg_term)

         Ident(assign(i,t)) ≡ i,

         Get(assign(i,t)) ≡ t

    end of object
```

Since the constructor Assign is not self embedded, it is possible to declare the type CSG_ENTITY in a R^2D^2 database system. The axiom set of CSG_ENTITY is primitive-recursive. So CSG_ENTITY is sufficiently complete, consistent, and semantically unique.

object CSG_ENVIRONMENT

based on STANDARD,CSG_EXPR,CSG_ENTITY

sort csg_enviro

cons Empty : csg_enviro,
cons Insert : (csg_entity,csg_enviro) csg_enviro,
attr Contains : (csg_enviro,csg_entity) bool

axioms all (x:**euclid**, ce,ce_1, ce_2:**csg_entity**, env:**csg_enviro**)

contains(empty,ce) ≡ false,

contains(insert(ce_1,env),ce_2) ≡ (ident(ce_1)=$_w$ident(ce_2) or contains(env,ce_2)),

contains(ce,env) ⇒ (insert(ce,env)≡env)

end of object

The object type CSG_ENVIRONMENT is right-recursive. So it is possible to declare CSG_ENVIRONMENT in a R^2D^2 database system. Furthermore one can show, that the axiomset of CSG_ENVIRONMENT is primitive recursive. This implies that the type CSG_ENVIRONMENT is sufficiently complete, consistent and semantically unique.

A model of type CSG_ENVIRONMENT corresponds to a CAD database. In general one has to specify not only Insert but also database operations like Delete, Modify, etc..

object CSG_EXPRESSION

based on VALUE,EUCLID,CSG_TERM,CSG_ENTITY,CSG_ENVIRO

sort csg_expr

cons Base : (csg_term) csg_expr,
cons ⊕ : (csg_expr,csg_expr) csg_expr,
cons ⊖ : (csg_expr,csg_expr) csg_expr,
cons ⊗ : (csg_expr,csg_expr) csg_expr,
modi Subst : (csg_expr,csg_expr,string) csg_expr,
modi Value : (csg_expr,csg_enviro) csg_expr,
rela Expr_el : (euclid,csg_expr) bool

axioms all (i:**string**, e, e_1, e_2 :**csg_expr**, t:**csg_term**, ce:**csg_entity**, env:**csg_enviro**)

Expr_el(x,Base(t)) ≡ Term_el(x,t),

Expr_el(x,($e_1 \oplus e_2$)) ≡ (Expr_el(x,e_1) or Expr_el(x,e_2)),

Expr_el(x,($e_1 \ominus e_2$)) ≡ (Expr_el(x,e_1) and ¬Expr_el(x,e_2)),

Expr_el(x,($e_1 \otimes e_2$)) ≡ (Expr_el(x,e_1) and Expr_el(x,e_2)),

Subst(Base(Prim_const(p)),e,i) ≡ Base(Prim_const(p)),

Subst(Base(Identifier(j)),e,i) ≡ **if** (Eq(i,j) **then** e **else** Base(Identifier(j)),

Subst(Base(Plus(j,k)),e,i) ≡ (Subst(Base(j),e,i) ⊕ Subst(Base(k),e,i)),

Subst(Base(Minus(j,k)),e,i) ≡ (Subst(Base(j),e,i) ⊖ Subst(Base(k),e,i)),

Subst(Base(Mult(j,k)),e,i) ≡ (Subst(Base(j),e,i) ⊗ Subst(Base(k),e,i)),

Subst(($e_1 \oplus e_2$),e,i) ≡ (Subst(e_1,e,i) cplus Subst(e_2,e,i)),

Subst(($e_1 \ominus e_2$),e,i) ≡ (Subst(e_1,e,i) cminus Subst(e_2,e,i)),

Subst(($e_1 \otimes e_2$),e,i) ≡ (Subst(e_1,e,i) ctimes Subst(e_2,e,i)),

Value(e,Empty) ≡ e,

Value(e,Insert(ce,env)) ≡ Subst(Value(e,env),Base(Get(ce)),Ident(ce))

end of object

One can show, that the axiomset of CSG_EXPRESSION is primitive recursive. So CSG_EXPRESSION is sufficiently complete, consistent and semantical unique. One notices that CSG_EXPRESSION is self-embedded, i.e. CAD models of type CSG_EXPRESSION cannot be stored in a R^2D^2 database system.

object REGULAR_CSG

based on VALUE,EUCLID,CSG_ENTITY,CSG_ENVIRO

sort reg_csg

cons Mk_csg : (csg_entity,csg_enviro) reg_csg,
rela Reg_csg_el : (euclid,reg_csg) bool

axioms all (i:**string**, t:**csg_term**, env:**csg_enviro**)

Reg_csg_el(x,Mk_csg(assign(i,t),env)) ≡ Expr_el(x,Value(Base(t),env))

end of object

One can show, that the axiomset of REGULAR_CSG is primitive recursive. One notices that REGULAR_CSG is regular so that all models of type REGULAR_CSG can be stored in a R^2D^2 data base.

2.4 Scanner and parser

As a consequence of the changes in the specification from 2.1 via 3.2 to 3.3, the parser tables had to be regenerated a few times. As an additional capability a type checking mechanism was introduced in the parser. The parser can now test whether a reference to an entity refers in fact to an entity of the proper type.

New tables were also required for interpreting STEP files according to the STEP specification, St. Louis version. All these changes could be accommodated easily with the parser generator which, thus, has proven to be a very useful software tool. In order to be able to read immediately the syntax specification for STEP files, which will be in WSN syntax form rather than BNF, a new version of the parser table generator was developed that accepts this form directly.

2.5 Processor development for version 3.3

Processors for solid model transfer according to the specification have been developed for the following commercial systems:

-	Bravo	(Applicon)	at KfK
-	Catia	(Dassault)	by KfK at JET
-	Euclid	(Matra Datavision)	at KfK
-	Geomod	(SDRC)	at UKA
-	Icem	(Control Data)	at NEH
-	Proren	(Isykon)	at KfK
-	Romulus	(Shape Data)	at CIT
-	Technovision	(Norsk Data)	at DTH

Systems (Solid Modelling) Overview
BRep capabilities (CAD*I-V3.3)
= = > Released on: November 14, 1988

system / CAD*I entity	CATIA	GEOMOD		PROREN		ROMOLUS		TECH3D	
	POS	PRE	POS	PRE	POS	PRE	POS	PRE	POS
Point	d	d	d	M	M	M	M	M	M
Direction	d	?	?	D	D	M	M	D	D
Line	d	d	d	D	D	M	M	D	D
Circle	d	d	d	D	D	M	M	D	D
Ellipse	d	?	?	D	D	m	m	-	A
Parabola	d	?	?	D	D	d	a	-	A
Hyperbola	d	?	?	D	D	d	a	-	A
Polygon	d	d	d	D	D	d	d	M	M
B-Spline-Cv.	d	?	?	a	a	a	?	-	?
Planar-Sf.	d	d	d	D	D	M	M	D	D
Conical-Sf.	d	?	?	D	D	M	M	D	D
Cylind.-Sf.	d	d	d	D	D	M	M	D	D
Spherical-Sf.	a	d	d	D	D	M	M	D	D
Torus- Sf.	a	?	?	-	-	M	m	D(1)	D(1)
B-Spline-Sf.	d	?	?	a	a	?	?	?	?
Sf. of Rev.	d	d	d	-	-	?	?	?	?
Sf. of Trans.	?	d	d	-	-	?	?	?	?
Vertex	d	?	?	D	D	M	M	M	M
Edge	d	?	?	D	D	M	M	D	D
Loop	a	?	?	D	D	M	M	D	D
Vertex_loop	a	?	?	d	d	M	m	d	d
Face	d	?	?	D	D	M	M	D	D
Shell	d	?	?	D	D	M	M	D	D
BRep-result	d	?	?	D	D	m	m	D	D
Assembly	?	?	?	a	a	M	m	D	d

Table 1

Systems (Solid Modelling) Overview
CSG capabilities (CAD*I-V3.3)
= = > Released on: November 14, 1988

system / CAD*I entities	BRAVO3 PRE	BRAVO3 POS	CATIA POST	EUCLID PRE	EUCLID POS	ICEM PRE	ICEM POS	PROREN POST	GDS PRE	GDS POS
Assembly (la=3)	M	M	M	-	-	M	M	M	M	M
Assembly (la=8)	-	M	-	M	M	-	M	a	M	M
Instancing	-	M	m	m	m	-	m	?	M	M
Component	M	-	M	D	D	M	M	M	D	D
Construct	M	M	M	M	M	M	M	M	M	M
ROT MATRIX	-	m	m	D	D	M	m	?	D	D
ROT GLOBAL	-	m	-	d		M	m	?	M	M
ROT AXIS	-	m	-	d	d	M	m	?	d	d
Operands										
DIFFERENCE	M	M	-	M	M	M	M	M(2)	M	M
UNION	M	M	-	M	M	M	M	M(2)	M	M
INTERSECT.	M	M	-	M	M	M	M	M(2)	M	M
Primitives										
BOX	M	M	M	A	A	M	M	M	M	M
S. SPHERE	M	M	M	A	A	M	M	M	M	M
S. CYLINDER	M	M	M	M	M	M	M	M	M	M
TR. CONE	M	M	M	M	M	M	M	M	M	M
TR. PYRAMID	-	?	a	-	-	-	-	M	D	D
REG. PRISM	-	?	A	a	a	-	M	M	D	D
TORUS	M	M	M	a	a	M	M	A	M	M

Table 2

Systems (Solid Modelling) Overview
CSG (sweeps) capabilities (CAD*I-V3.3)
= = > Released on: November 14, 1988

CAD*I entities	BRAVO3 PRE	BRAVO3 POS	CATIA POST	EUCLID PRE	EUCLID POS	ICEM PRE	ICEM POS	PROREN POST	GDS PRE	GDS POS
ROTATIONAL & LINEAR SWEEP										
contour element	M	M	m	-	M	m	m	m	m	m
point	M	M	m	-	M	m	m	m	d	d
line	M	M	m	-	d	m	m	m	d	d
line segment	M	M	m	-	M	m	m	m	d	d
circle	M	M	m	-	M	m	m	m	d	d
ellipse	m	m	-	-	-	-	-	m	-	-
parabola	m	m	-	-	-	m	m	?	-	-
hyperbola	m	m	-	-	-	-	-	?	-	-
poly curve	-	?	-	-	-	-	-	?	-	-
B-spline curve	m degree 3	?	-	-	-	-	-	-	-	-

Table 3

Systems (Solid Modelling) Overview
Polyhedron capabilities (CAD*I-V3.3)
= = > Released on: November 14, 1988

systems CAD*I entities	CATIA PRE POS	EUCLID PRE POS	GEOMOD PRE POS	ICEM PRE	PROREN POS	TECH3D PRE
Assembly (la=3)	M -	M M	? ?	M	M	M
Assembly (la=8)	M -	M M	? ?	-	-	-
Instancing	M -	m m	? ?	m	-	m
POLYHEDRON	M -	M M	d m	M	D	D
POLY_SHELL	D -	D D	d m	M	D	D
POLY_FACE	M -	M M	d m	M	D	D
POLY_LOOP	M -	M M	d m	M	D	D

Table 4

The given Solid Modeling capabilities (CSG, B-Rep and Polyhedron) are available under the System Version/Release:

	BRAVO3	CATIA	EUCLID	GEOMOD	ICEM	PROREN	ROMULUS	TECH3D
V	1.4/							
R	2.0	2.2	4.2			87.1	6.0/J	H00

Legend:
- : not mappable
? : not known yet
d : exactly derivable, not implemented
D : exactly derived, implemented
m : exactly mappable, not implemented
M : exactly mappable
a : can be approximated, not implemented
A : approximated, implemented
(1): only partial open torus can be implemented
(2): only realized for penetrating bodies with no tangent surfaces

In addition processors for parametric CSG models are being developed for the R&D system

- GDS (DTH) at DTH

The table at the end of this chapter gives an overview on the capabilities of the CAD*I processors for solid model transfer as of October 1988.

In addition to these processors which are related to individual CAD systems, a pair of converters has been developed between the CAD*I format and the STEP format. The St.Louis version of the STEP specification was used as a reference. The similarity between the CAD*I approach and the STEP approach is perhaps best illustrated by the fact that it took no more than two months to implement a pair of converters for a subset of CSG and B-rep capabilities to the extent that cycle tests could be performed successfully. The converters were tested with the CSG models shown in Fig. 2.5-1 and the B-rep models shown in Fig. 2.5-2. During 1988, the STEP specification developed even closer towards the CAD*I specification so that CAD*I has decided not to support the converter approach any longer, but rather to convert its processors to conform with the STEP file format. This conversion will be performed as soon as the STEP specification becomes sufficiently stable which is expected to happen at the ISO/TC184/SC4 meeting in Tokyo, early in December 1988.

2.5.1 Status of processors for Bravo3 (Applicon)

The CAD*I processors for the CAD system Bravo3 were enhanced from specification version 2.1 via version 3.2 to the present state which corresponds to version 3.3. CSG geometry and assembly structures can be transmitted from and received by Bravo3. Emphasis was then put into implementing more advanced capabilities of the specification:

- instancing and
- external references.

The post-processor can accept instances and reconstruct the corresponding data structures in Bravo3. The pre-processor does not provide for this feature as the information is already lost when the solid model is written to the Synthavision file (which serves as input to the pre-processor). All instances (in the data base) are replaced by copies of the instanced entity on the Synthavision file. The external referencing mechanism has been analyzed and implemented in the post-processor.

Tests are presently being prepared with CAD*I files from DTH containing external references.

Fig. 2.5-3 shows the result of a test in which a geometric model consisting of a block with holes and a doubly instantiated bolt was transmitted from the system GDS at DTH to Bravo3. The left-hand part of the figure shows the result immediately after the transmission. then, the bolt (which exists only once in the data base) was modified by slotting its head. This immediately changes both instantiations. This test proves that the CAD*I approach to solid model transfer not only maintains the proper shape of solid models in a static sense, but also maintains the intended behaviour of models which contain instances of other models.

Bravo3 was used as the test system for performing the cycle tests from a CAD system via a CAD*I file onto a STEP file and back via CAD*I into the system. Test results are shown in Fig. 2.5-1.

The Bravo3 processors are now complete with respect to the project goals except for the testing of the external referencing mechanism. The documentation of the processors has begun.

2.5.2 Status of processors for Catia (Dassault/IBM)

The CAD system Catia

Catia, distributed by IBM, is a two- and three-dimensional CAD system developed by Dassault Systems, France. Catia runs only on the IBM mainframes (e.g. IBM 3090) and on the IBM6150 workstation. It is written in Fortran 77 and Assembler and operates currently under Version 3, Release 1.

Catia works on the following three layers:

* The Base layer with data and communication management, library functions, and data base access routines (i.e. CATGEO and CATMSP);

* The Geometry modelling layer comprising drafting/drawing, 3D design, advanced surfaces, and solid modelling capabilities;

* The Application dependant layer with Catia modules for different applications like numerical control, robotics, kinematics, and building design.

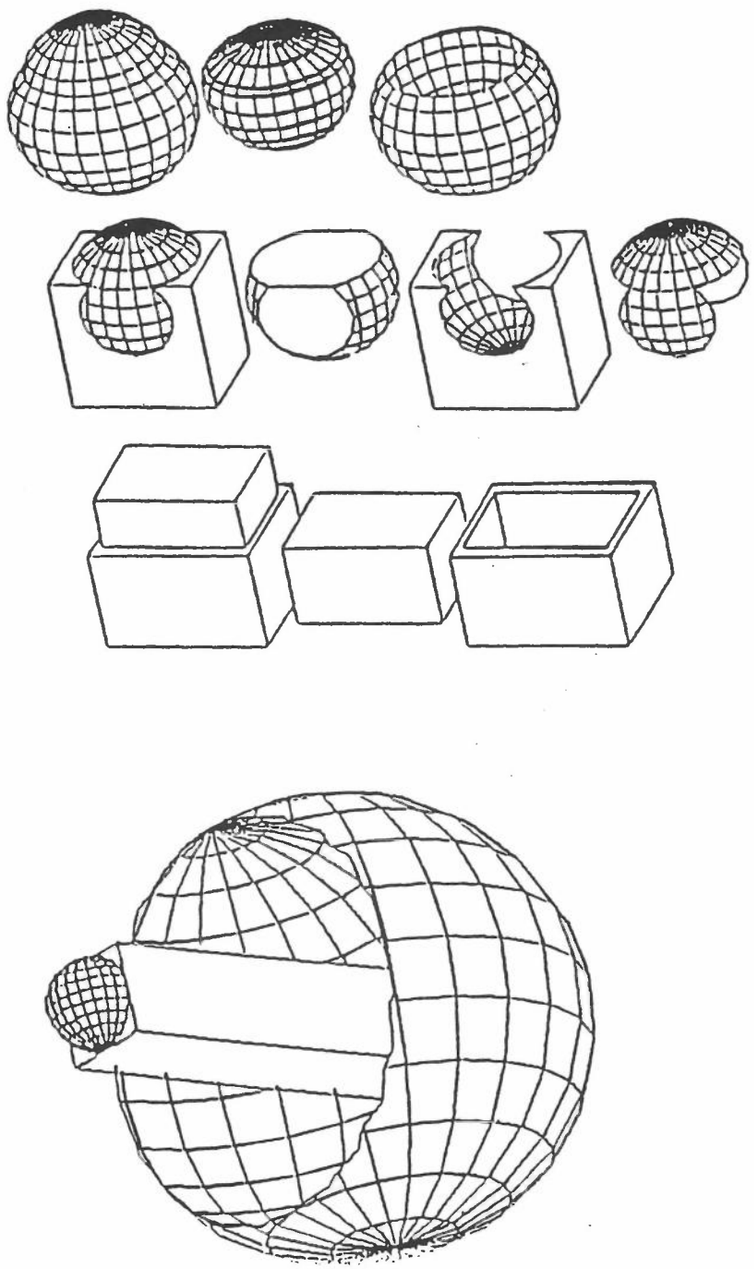


Fig. 2.5-1:
CSG models used for STEP/CAD*I conversion tests

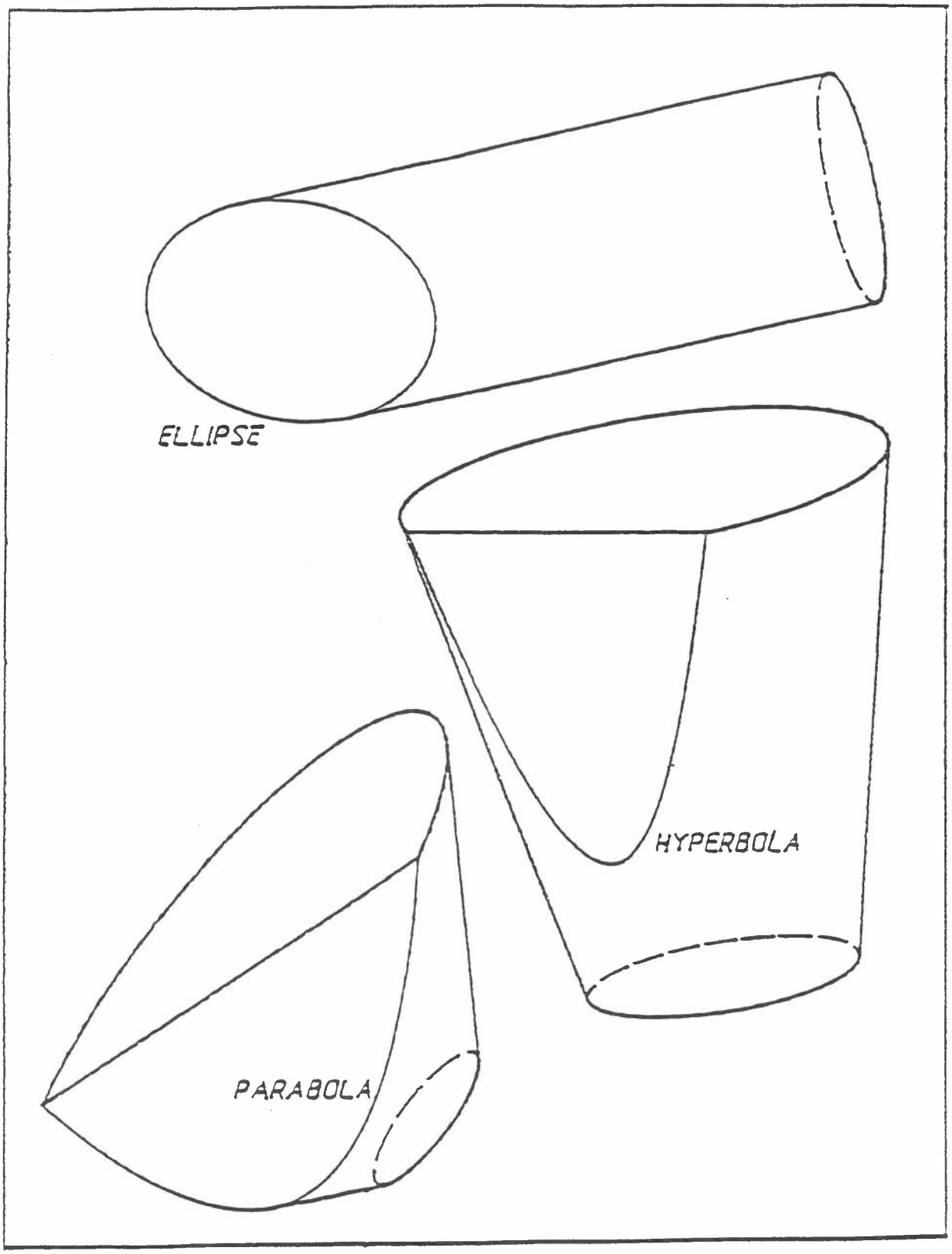


Fig. 2.5-2:
B-rep models used for STEP/CAD*I conversion tests

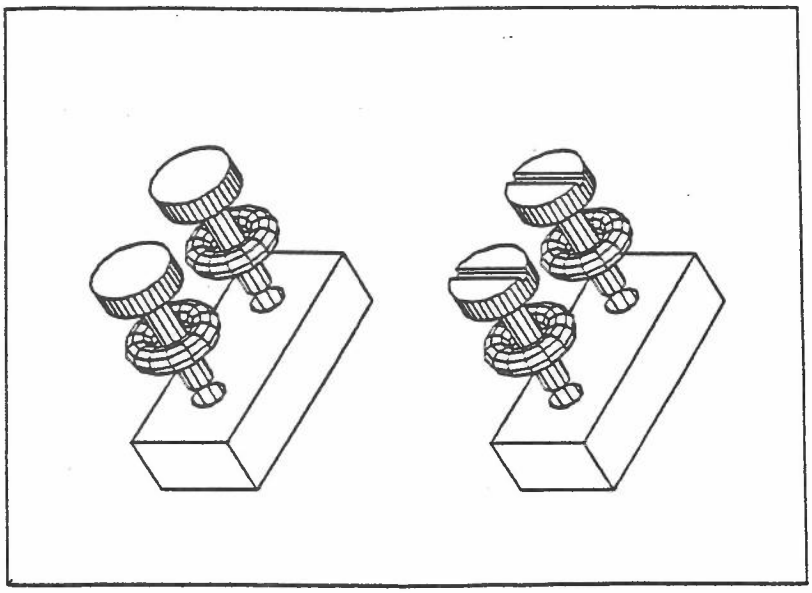


Fig. 2.5-3:
A CSG model in which a bolt is instantiated twice.
left-hand side: immediately after transfer from GDS (DTH)
Right-hand side: after modifying the (only one) bolt model.

The Catia Solid Geometry Module (SGM) is a planar-facet approximation type of solid modeller. Additionally the SGM provides the exact representation (Catia: 'canonical form') of basic primitives and the corresponding boolean operators. Currently Catia (SGM) provides the following CSG-primitives:

Box, Sphere, Cylinder, Cone, Torus, Prism, Pyramid, Pipe, and Solid of Revolution.

The corresponding boolean operations are:

Union, Subtraction, and Intersection.

Besides the possibilities creating solids by Boolean operands and operations, surfaces may be converted to solids by associating a thickness to the surface or projecting the surface onto a plane. Also volumes (created by complex surfaces in the Catia surface modeller) may be converted into a solid on condition that each surface element has at most 3 or 4 edges.

Catia provides an assembly mechanism (within Catia called 'SET') with up to 124 components. Nesting of assemblies is not possible.

Instancing of already existing assemblies (instancing of up to one assembly) comprising one or more solids is possible as internal 'DETAIL' (within a Catia model) and as an external 'DETAIL' brought in from an external library.

Interfacing Catia

The geometrical data is stored in a binary database form, a text file form (i.e. journal file) is not available. Catia has different geometry interface concepts and is thus able to acquire data originating from other CAD systems as well as providing data for different applications, e.g. FEM systems. The Catia system interfaces are:

* CATGEO - allows access to the Catia database by a Fortran subroutine library in batch mode;

* CATMSP - means Catia Mathematical Subroutine Package, which is available with Version 3.1 to evaluate and calculate mathematical properties of curve and surface elements;

* IUA - an interactive Catia interface concept by means of IUA-specific procedural language elements. IUA

allows the interactive user access to the functionalities of CATGEO and user specific Fortran routines.

Both CATGEO and IUA allow the user to create, modify, and interrogate solid model data. The current version (V3R1) allows creating and reading of the following solid models:

* CSG - models comprising a number of Boolean operands and operations as mentioned before;

* the facetted boundary representation, also known as polyhedron;

* the exact boundary representation by using the surface capabilities of Catia, this has to be converted internally (Catia) into a so-called polyhedron solid.

In order to communicate with other Cxx-systems Catia already provides pre- and post-processors, related to IGES 3.0, VDAFS 1.0, and SET 1.1. It is not possible to transfer solid model information, by these interface concepts.

Status of CAD*I processor development

The Catia/CAD*I processor development comprises pre- and post-processors, according to the CAD*I neutral file specification version 3.3 and Catia version 2.2. The following processors have been implemented:

- Polyhedron pre-processor, and
- CSG post-processor.

The software is written in Fortran 77 and has been implemented on an IBM 3090XA under MVS environment. Both the pre- and post-processor interrogate the Catia database by CATGEO (V2.2) subroutine calls. The processors have been developed to enable the data transfer of solid models between Catia and the real-time robotic simulation system KISMET. The actual data transfer comprises:

* manipulator geometry parts (geometry especially created in Catia for the robotic simulation system), and

* 'static' geometry parts of the robotic environment.

Furthermore, the processors are in common use for the data transfer between Catia and other CAD systems (e.g. Euclid) and vice-versa.

Pre-processor

The main purpose of the Catia - pre-processor is to map the Catia data structure (polyhedron elements) onto the CAD*I neutral file, which is realized with some exceptions (calculation of inner shells and loop orientation) straightforward.

The pre-processor handles currently (in CAD*I terms):

- ASSEMBLY (assembly level la=8: the hierarchy is world, assembly, subassembly, and component);
- INSTANCE (instances of assemblies);
- INDEX_ENTRY (user defined names of assemblies, instances, and polyhedron);
- RENDER_POLYLINE (colour attributes of the polyhedron entity, an extension to the CAD*I schema);
- POLYHEDRON (POLY_SHELL, POLY_FACE, POLY_LOOP, and POINT);

In all cases where the evaluated polyhedron model on the Catia database is the result of a Boolean operation on non-intersecting primitives (Boolean operations of primitives with contacting surfaces) the pre-processor writes onto the CAD*I neutral file a polyhedron model with all edges and faces of all previously defined primitives. This Catia system fault constrains the user to check each Catia solid model for non-intersecting solids before running the pre-processor.

The next step of the pre-processor development will be a revised implementation of the current pre-processor (Version 1.0) according to CATGEO Version 3.1.

Post-processor

The post-processor is based on the limited capabilities of CATGEO Version 2.2, which allow only the creation of a subset of available CSG operands (within Catia). Furthermore it is not possible to map the Boolean operations union, subtraction, and intersection (available on the CAD*I-NF) onto the Catia database.

The following organizational data can be mapped onto the CAD*I neutral files:

- ASSEMBLY (assembly structure with three levels, no nested assemblies);

- INDEX_ENTRY (user-defined name of an assembly and construct);

The Catia post-processor handles up to now the following subset of CAD*I CSG operands:

- BOX;
- SOLID_SPHERE;
- SOLID_CYLINDER;
- TRUNCATED_CONE;
- REGULAR_PRISM;
- SOLID_TORUS.

The next steps will be:

- implementation of the current post-processor capabilities under CATGEO V3.1.

- the implementation of the Boolean operations (union,subtraction, and intersection), using CATGEO V3.1,

Performed tests

A number of intersystem data exchange tests have been performed in order to check the above mentioned pre- and post-processor capabilities. The testparts which have been used were either provided by industrial partners or especially developed test parts for Catia (some correct, and some faulty, such as empty ones, models without any solid geometry). The following solid modelling systems were involved in intersystem tests:

- the data transfer of polyhedron models (CAD*I neutral file version 3.3) from Catia to Euclid and to Proren, an example of an industrial testpart is given in Fig. 2.5-4;

- the data exchange of polyhedron models (especially for robotic simulation purposes created geometry) from Catia to KISMET as shown in Figure 2.5-5;

- data transfer of CSG models from Euclid as well as from Bravo3 to Catia, refer to Fig. 2.5-6 for an example.

Within Catia the test parts were either created in 'SOLID' mode by Boolean operands and operations or in 'SURF' mode by using analytical as well as free- form curve and surface elements. The latter case implied the

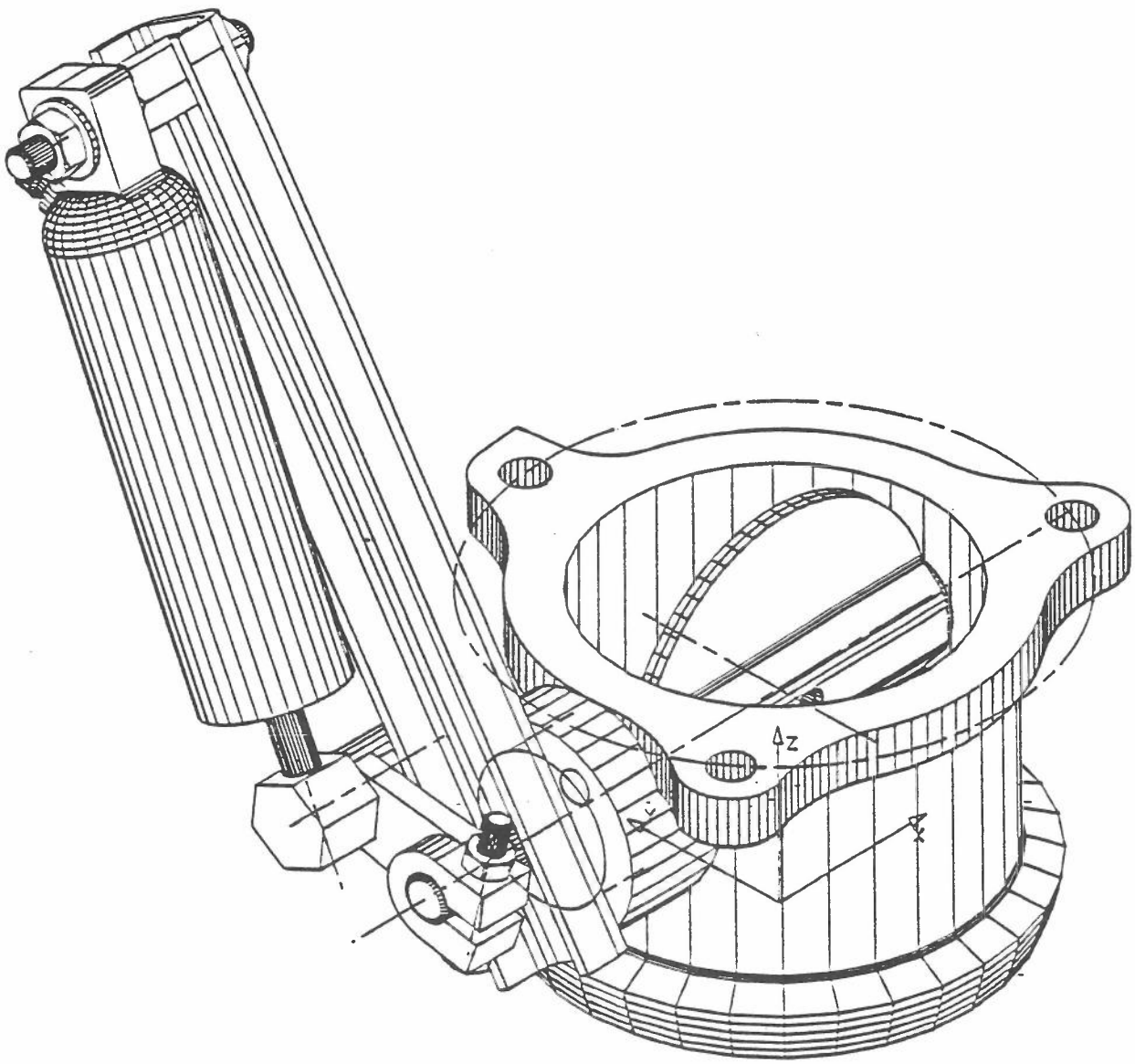


Fig. 2.5-4:
A polyhedron test model (the carburetor) as transferred from CATIA to EUCLID and PROREN.

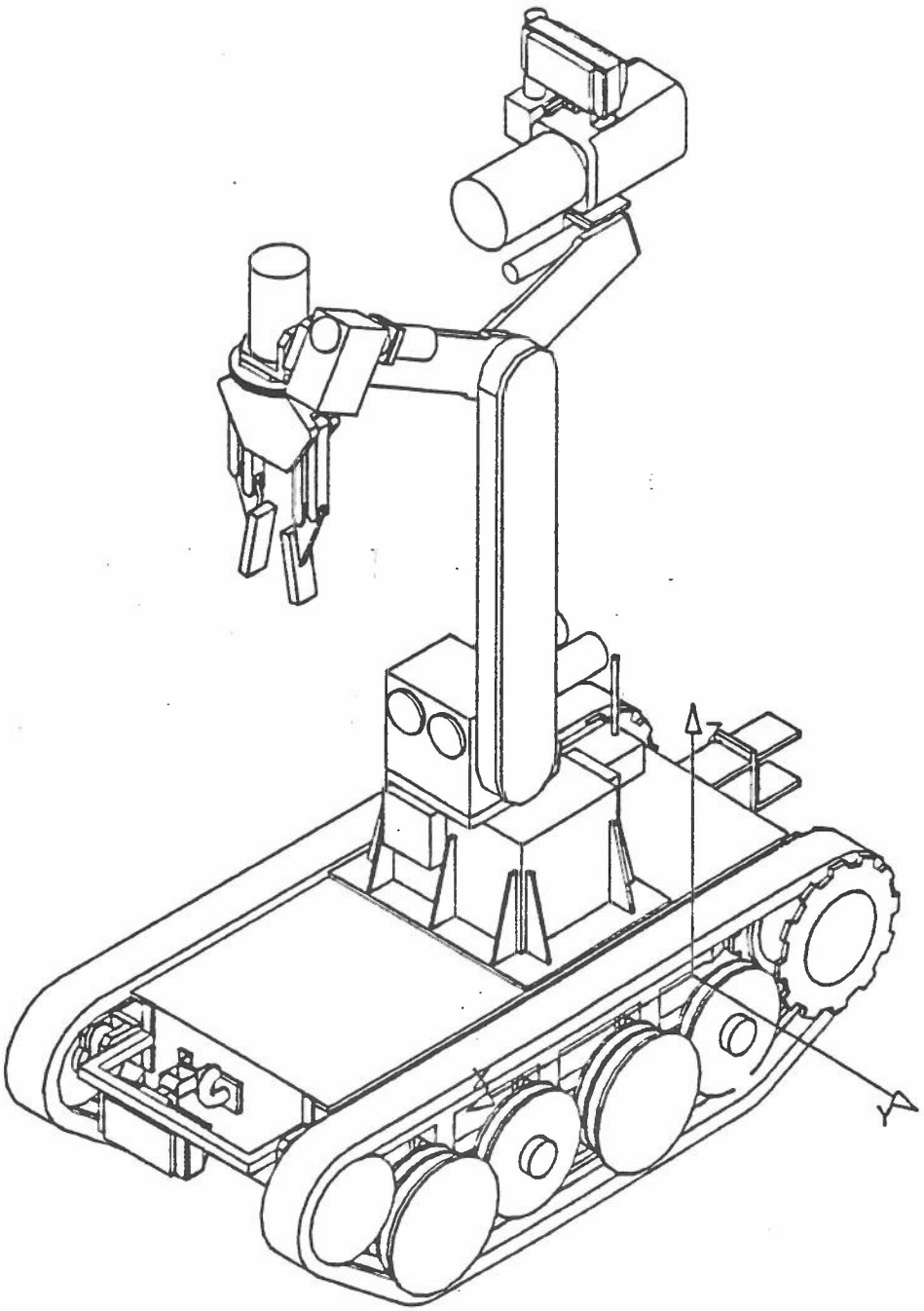


Fig. 2.5-5:
A polyhedron test model as transferred from CATIA fo the robotic simulator KISMET.

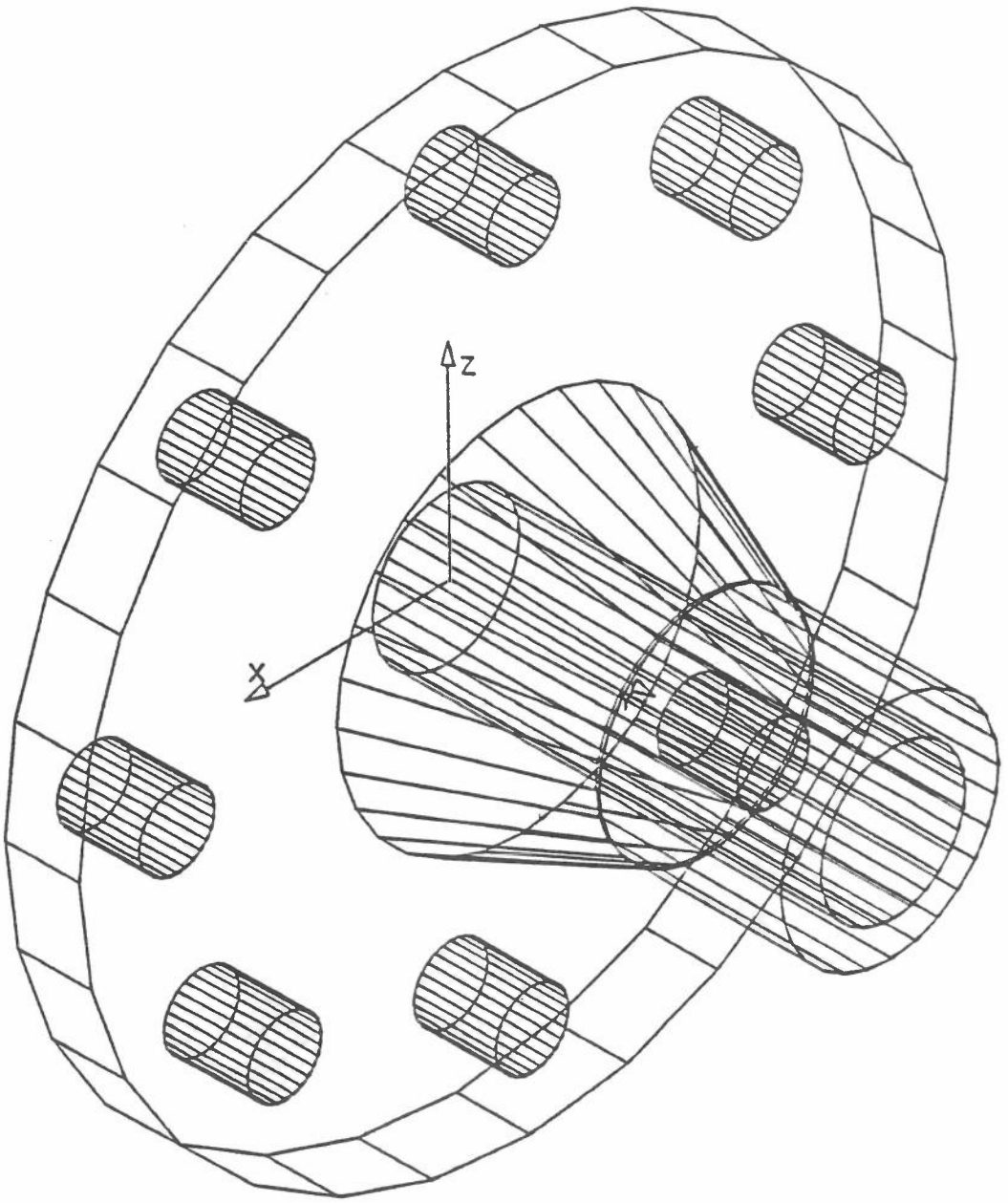


Fig. 2.5-6:
A polyhedron test model as transferred from EUCLID to CATIA.

transformation 'TRANSFORM' of curve and surface elements into a volume and finally into a solid.

The Catia processor development is complete with respect to the project goal. A conversion of the processors from CATGEO version 2.2 to CATGEO version 3.1 is scheduled for spring 1989.

2.5.3 Current Status of Processors for Euclid (Matra DataVision)

Two major work packages have been completed:

- o Linear and Rotational Sweep entities have been implemented in the post-processor. This allowed the successful transfer of the "COIL" testpart from Applicon Bravo3 to Euclid. The respective capabilities have been implemented in the pre-processor.

- o The POLYHEDRON pre-processor has been redesigned completely in order to overcome the flaw of the last version which consisted in its inability to handle multiple shells properly. This problem has been solved by calculating the shells from the face information. Successful transfer of "Multi Shell Objects" has been performed with the POLYHEDRON post-processor of Proren.

In the second half year the polyhedron processors were tested thoroughly with cycle tests, with models sent to Proren, and with models received from Catia. Fig. 2.5-4 shows a test part received from Catia.

The EUCLID processors are now complete with respect to the project goal. Documentation will begin soon.

2.5.4 Status of processors for Geomod (SDRC)

Work has begun with the development of a post-processor for the conversion of a CAD*I boundary representation (B-rep), which is one possible representation of solid models, and the Geomod B-rep representation. The concepts and status are documented in this report.

2.5.4.1 General Concept

As all the other CAD*I post-processors, the post-processor for Geomod will be realised by a sequence of

program components that have special tasks. The sequence consists of following modules:

- scanner
- parser
- semantics analysis and data structure generation.

The Geomod post-processor uses for scanning and parsing the CAD*I SCANNER-PARSER which produces a parse-tree file. The scanner/parser analyzes the structure of the CAD*I statements and stores the "correct" statements in the parse-tree file. This file is made available for the following module by READ and GET statements. This module, called SEMGEN (Semantics and data structure generation for CAD*I files) converts the information into the format of the receiving database.

Fig. 2.5-7 shows this concept of RPK's Geomod post-processor; Fig. 2.5-8 shows a flow chart of a subroutine which evaluates a part of the B-rep data structure with the help of READ/GET and other routines.

2.5.4.2 Introduction to IDEAS

GEOMOD is one package out of a larger software family called I-DEAS.

IDEAS Software Families

IDEAS (Integrated design engineering analysis software) offers a comprehensive package for mechanical design engineers. This package provides capabilities for:

- solid modelling (GEOMOD)
- system assembly
- kinematics
- finite element pre/post processing
- finite element solution
- system dynamics
- drafting
- test data analysis
- project relational database

These capabilities are packaged as families of software. IDEAS consists of the following families:

- solid modelling family, which provides modelling of objects,assemblies and mechanics

- engineering analysis family, which provides mesh generation and output display for FEM applications, analysis and system dynamics

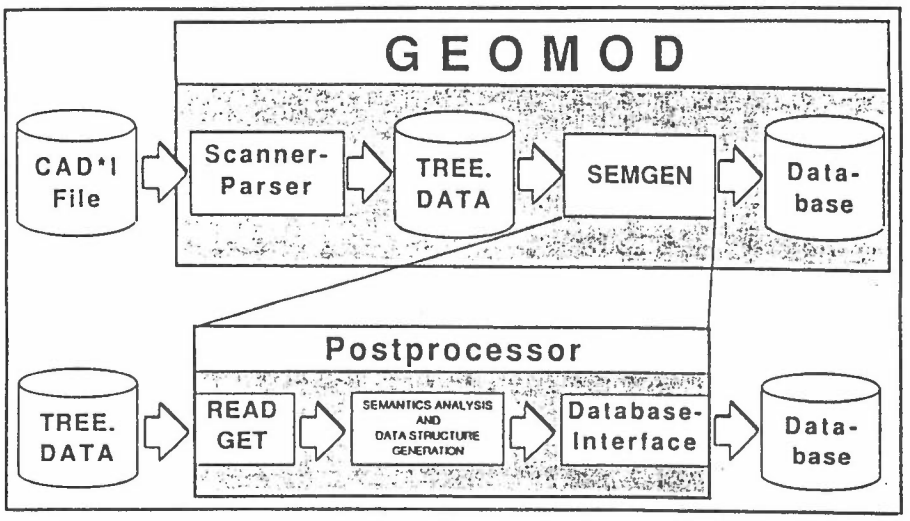


Fig. 2.5-7: Software structure for the Geomod post-processor

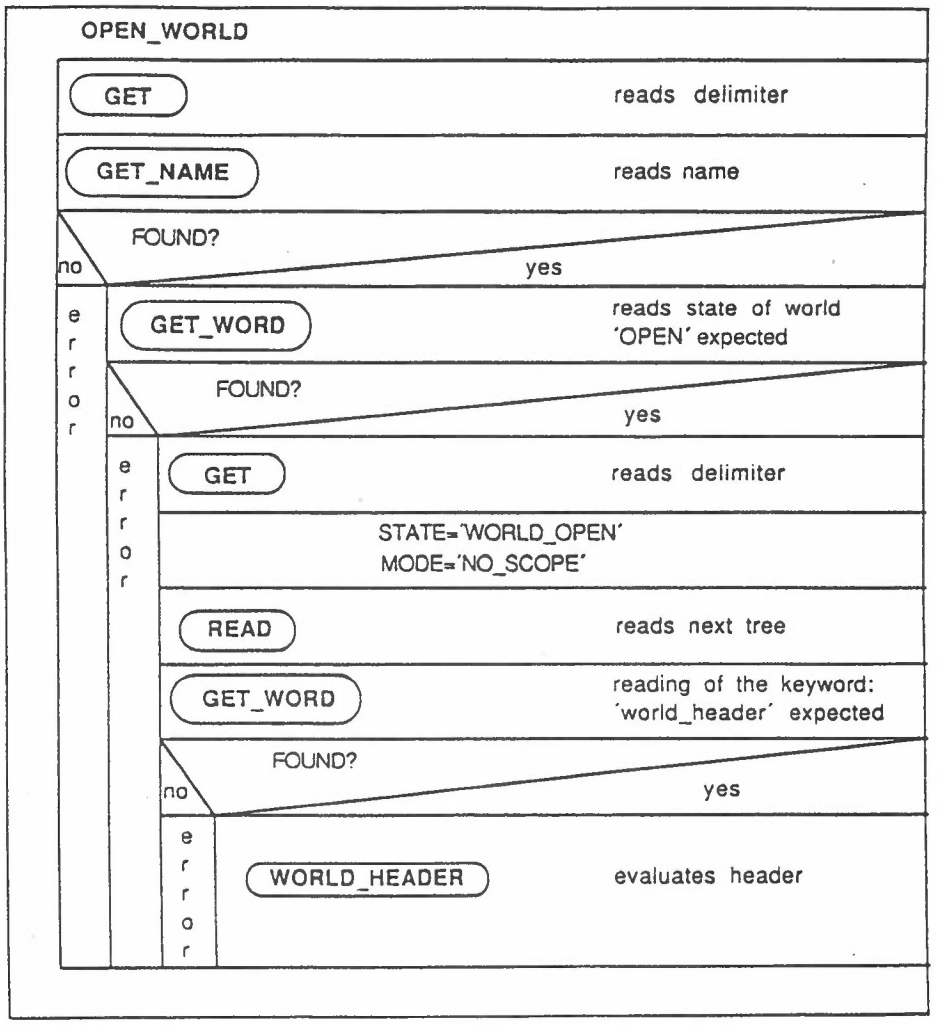


Fig.2.5-8: Flow chart of a B-rep evaluation routine

- drafting family for dimensioning of objects and graphical output

- test data analysis family, which provides different methods for testing and statistics

The communication between these families is realized by a so-called CAE Application Database. For further applications in the area of CAD/CAM there is the UNIVERSAL FILE and the PROJECT DATABASE and the PEARL DATABASE /10, pp.1-23/.

2.5.4.3 IDEAS Database System

UNIVERSAL FILE

The UNIVERSAL FILE format is in human readable form and can be used to interface with the special purpose in programs. Each UNIVERSAL FILE is a sequential formatted file with records that have a maximum of 8o characters /11, pp.25-1/.

Project Database

This project database appears as a system of tables to provide a clear, easily manipulated view of information, that have to be manipulated by the applications. Special user applications must handle these tables with the Pearl Data Manager (IDEAS Pearl) that allows to manipulate and analyse tables and individual items that form the IDEAS database. Within the project database all data are stored in rectangular tables. Commands can cooperate on several tables, some commands produce new tables with the result of the operation. The number of tables in each database and the number of rows in each table is not fixed or limited while the length of one row is fixed to 1015 computer words Data types of the table fields are INTEGER, REAL, DOUBLE PRECISION, or CHARACTER.

There are functions and interfaces that allow information handling, graphics and statistics with user-written programs. Functions that are available are:

- open and close a database
- create, copy, rename and delete tables
- modify elements of a table
- relate tables to one another
- analyse columns for specific data
- display tables as columns of values or as xy-plots
- write/read a table to/from a formatted file

There are some commands that will be important in the CAD*I project:

- for data comparison the project data manager allows to define a region of tolerance. So the user is able to create statements of conditions;

- to write tables to user defined formatted files there exists a translate command;

- to extract data from the pearl database there are the commands READ_VALUE and READ_ROW. The extract data is stored in IDEAS language variables.

Main steps in using Pearl are:

1) Enter pearl either by utility facilities or by the task-menu
2) Open the database
3) Create and manipulate tables
4) Display the table or part of the table
5) Modify the table
6) Relate the tables (subsets based on specific criteria, sorting in specified order, combining tables according to specified constraints)
7) Analyse the tables (maximum, minimum, sum ...)
8) Extract data into variables
9) Translate tables to or from external files
10) Managing of tables (store, retrieve, delete...)
11) Close/save the database

To realize the user dependant programs, there exists a subroutine interface to write own translators. There are some tools available to realize file translators:

1) IDEAS language is the major tool for implementation. This language provides facilities for interaction, for defining global symbols, assigning values to variables and calculation

2) Project Data Manager allows to access to the project database, that is organized in tables. The manager allows to combine tables or to create new tables or add data to existing tables.

3) Formatting commands allow to the formatting of tables for output. /11, pp.29-1/.

Pearl Database

A Pearl database is merely a collection of tables. Each table is composed of one or more columns , each representing an attribute of data in the table, and one

or more rows, each representing an instance of the attributes. The description of each column includes a data type, a number of values or characters in each row, and a unit type. The maximum length of a row is 1015 computer words. As shown in the last sections this is not very comfortable. The very new version of I-DEAS offers a more comfortable way for the user specific application to manipulate the data. This is a subroutine library that is available to access and convert geometry from the PEARL database. /11. pp.43-1 to 43-5/. Fig. 2.5-9 shows the I-DEAS families and databases.

2.5.4.4 Library architecture of PEARL-Interface

The major parts of the subroutine architecture are:

- access routines
- conversion routines
- in-core data management routines

where

- the access routines retrieve, store and delete data

- the conversion routines convert between non uniform rational B_SPLINE (NURBS) form and other representations of geometry

- in-core data management routines exist to insulate you from a specific implementation of the data structure associated with the NURBS representation /11, pp.32-1 to 32-4/.

2.5.4.5 Geomod's Data Structure

The geometric data structure of objects can be divided in object geometry and context free geometry. The object geometry consists of the topological entities surfaces, curves and points, that are represented in a consistent mathematical way, where curves and surfaces are of non uniform B-Splines type (NURBS). Context free geometry is represented in the same mathematical way, but grouped by special entities of type: SET_OF... Fig. 2.5-10 shows this structure of Geomod geometry.

Nurbs Curve Representation

The NURBS CURVE REPRESENTATION consists of two sets of data:

- NCUR (integer data)
- CUR (real data)

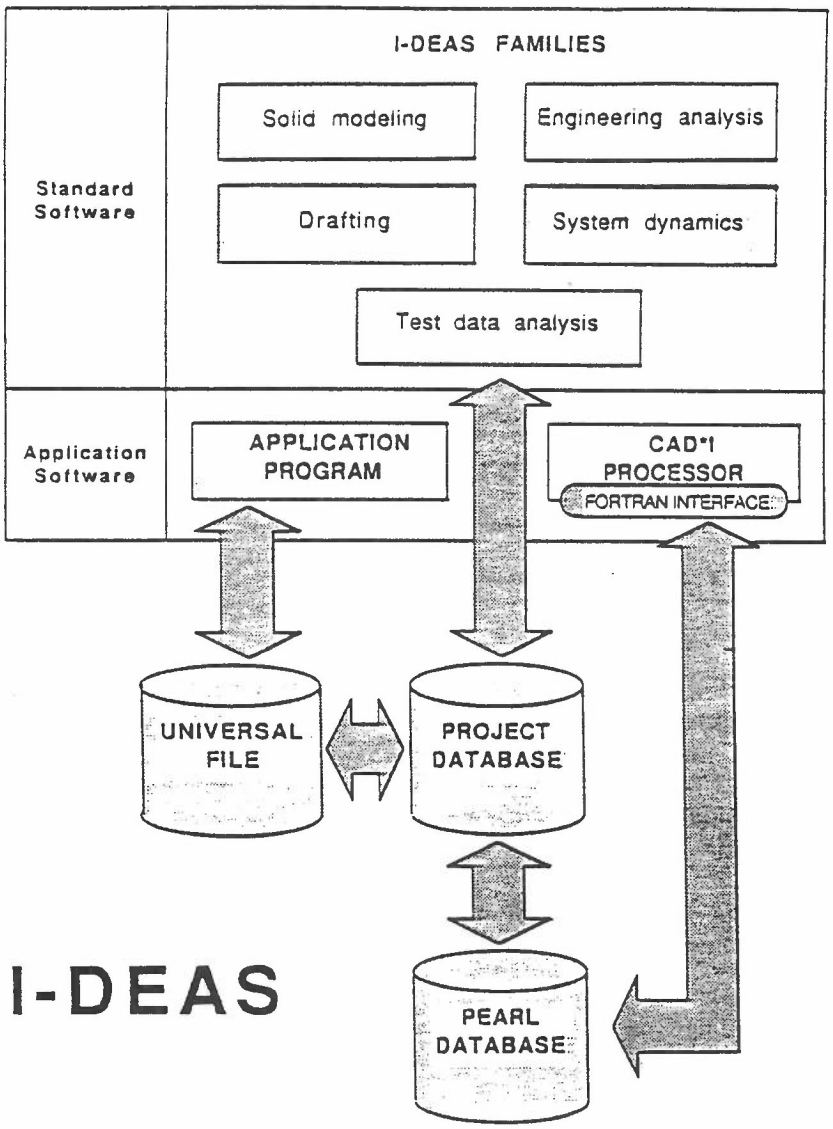


Fig. 2.5-9: I_DEAS software families and data bases

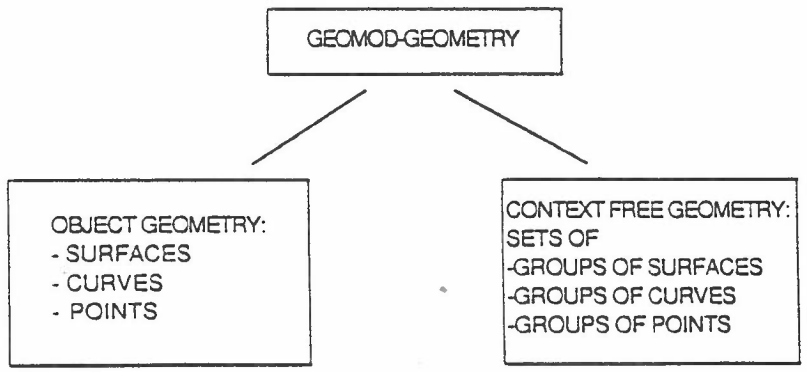


Fig.2.5-10:The Geomod geometry model

The required integer data consists of a number of points, knots, flags, form for line, circular arc, rational spline,... .The required real data holds the real portion of the NURBS REPRESENTATION.

Curve Groups

This data structure is needed by loops or composite curves for trimmed surfaces

- NCURHP: NCUR structure for each component and pointers to CUR structure and next NCUR structure
- CUR: curve data for each curve

Surfaces

The surface representation consists of two structures of data

- integer data NSUR (ISUR)
- real data SUR

The integer data structure consists of the number of points, knots, flags, surface type, etc. The real data structure contains the knot vectors, control points, etc. /11, pp.40-1 to 40-7/.

Geomod B-rep structure

The B_rep consists of topological entities, which describe how the geometry is related and connected, and geometric data, which describe the shape and the object /11, pp.40-1 to 40-7/. Fig. 2.5-11 shows the Geomod B_rep structure.

2.5.5 Current Status of Processors for Icem (Control Data)

Work has concentrated on updating of the Icem Solid Modeler neutral file processors according to specifications version 3.2 and subsequently to version 3.3. The processors have been successfully cycle tested using the NEH FREEZER test object and CSG test objects from KFK (Bravo & Euclid). The KFK test objects have been post-processed for Icem Solid Modeler and then pre-processed to neutral files. The resulting neutral files have been returned to KFK for further testing. Fig. 2.5 - 12 shows the FREEZER test part as it was received by Bravo3 from ICEM.

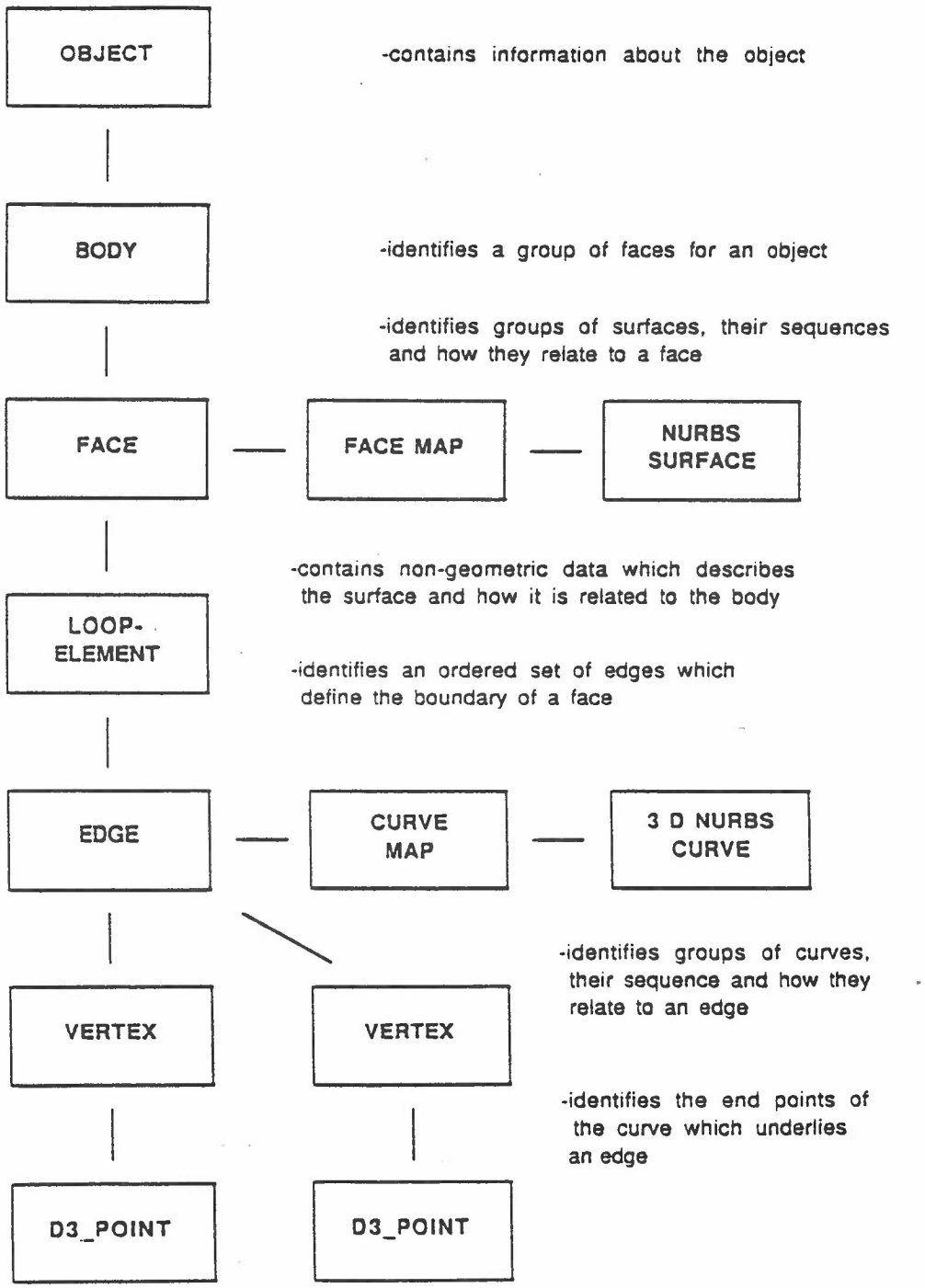


Fig. 2.5-11: The Geomod B-rep model

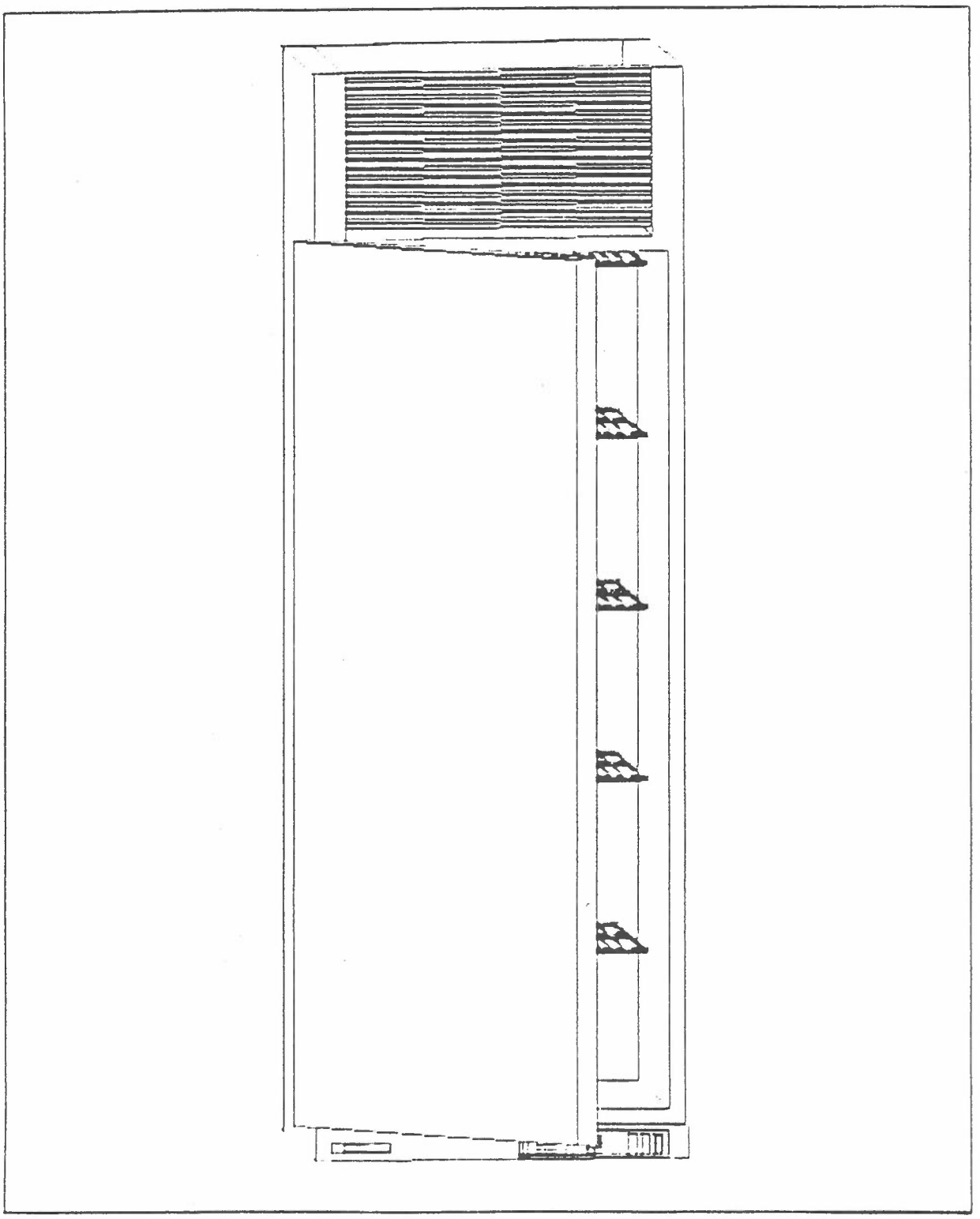


Fig. 2.5-12:
The FREEZER test part after transfer from ICEM to BRAVO3

WG6 test objects have been modelled in Icem and version 3.2 neutral files produced and distributed to the WG6 partners for use in testing of the WG6 processors.

A description of the Icem Solid Modeler version 3.2 processor capabilities have been distributed to the WG2 partners. The possibilities of using instancing and external references in ICEM Solid Modeler have been investigated.

The processors for ICEM are considered complete with respect to the project goal. Documentation has begun.

2.5.6 Current Status of Processors for Proren (Isykon)

The following processors, according to version 3.3 of the CAD*I specification and Proren2 version 87.1-VAX, are currently implemented:

- o CSG post-processor
- o B-rep pre-processor
- o B-rep post-processor
- o Polyhedron post-processor.

The processors are able to work both in interactive or in batch-mode.

CSG post-processor

The CSG post-processor handles currently :

- o Boolean tree operands
  - Box
  - SolidCylinder
  - SolidSphere
  - TruncatedCone
  - Torus (only approximated)
  - RegularPrism
  - Truncated Pyramid
- o Boolean operations
  - Difference
  - Intersection
  - Union
- o Sweeps (both rotational and linear) the geometric entities available in the sweeps are
  - line, line segment
  - polygon
  - circle

o structuring entities

- Assembly
- Construct
- Component
- Geometry Association.

The CSG post-processor produces a Command file which contains the corresponding Proren2 interface commands to build up a model.

B-rep pre-processor

The Proren2 B-rep pre-processor handles currently:

o Curve geometry

- Line
- Circle
- Ellipse
- Hyperbola
- Parabola
- Polygon

o Surface geometry

- PlanarSurface
- CylindricalSurface
- ConicalSurface
- SphericalSurface

o Topological information

- Vertex
- Edge
- EdgeLoop
- Face
- Shell

The processor works on a Proren2 native database and produces, besides the neutral file, some additional files containing statistical and runtime information.

B-rep post-processor

The Proren2 post-processor was developed in parallel to the pre-processor, therefore it can handle the same features than the pre-processor. As input of the post-processor is used the parse tree file of the CAD*I Parser and the output consists of a Proren2 native database and some additional files containing statistical or runtime information.

Polyhedron post-processor

This post-processor is able to post-process a neutral file containing a polyhedron model. Transferring the information into Proren, an automatic conversion into a (Proren-specific) B-rep-structure is done. Input to the post-processor is the parse tree file of the CAD*I Parser and the output consists of a Proren2 native database and some additional files containing statistical or runtime information. Fig. 2.5-4 shows a polyhedron test part as received from Catia.

The Proren processors are complete with respect to the project goal. The documentation of the processors is almost completed.

2.5.7 Current Status of Processors for Romulus (Shape Data)

The relationship between Romulus and CAD*I of assembly, instancing and external referencing mechanisms was analyzed and reported. This activity follows the working group agreement to concentrate on these particular issues regarding features which are more difficult to handle than mere geometry transfer. These features have not been tested so far, but as a consequence of the CAD*I participation in the ISO working group TC184/SC4/WG1 have recently been implemented in STEP in accordance with the CAD*I proposal.

The Romulus pre-processor development was completed according to specification 3.3 (excluding assembly structures). The pre-processor produces a polygon for any curve in the Romulus B-rep model other than line, circle, or ellipse. Although Romulus treats these curves as intersection curves the intersection_curve entity is not used on the file. All geometric entities produced are explicitely defined (embedded in other entities) rather than defined separately and then referenced.

Four approaches for post-processing in the Romulus environment have been investigated in parallel:

i) Edge-based construction via traversal of the vertex-edge graph, using a very limited set of general geometric/topological construction procedures (two only);
ii) Face-based construction via traversal of the face-edge graph, using boolean operations on sheet objects;

iii) "Direct" construction of nodes in the data base via a low-level interface;
iv) Generation of a sequence of Euler operations based on the neutral file structure.

The post-processor can handle most of the basic curve and surface entities for B-reps. However, polygon, parabola, and hyperbola entities can cause some problems, because these are represented as intersection curves, and not explicitely in Romulus. Selecting and orienting the correct curve from the general intersection of two surfaces can lead to inconsistencies.

Testing of the processors continued. Romulus models were successfully transferred to both Technovision and Proren. Partial success was achieved in receiving solid models from these systems.

The Romulus processor development is complete with respect to the project goal. Documentation has begun.

2.5.8 Status of Processors for Technovision (Norsk Data) and GDS

By the end of November 1988 the processors to and from Technovision were upgraded to cover the range of boundary representation geometry in the CAD*I specification version 3.2. Cycle-tests were performed without any errors.

Recent changes in the implemented version of Technovision have caused some major changes in the interface possibilities to the system. The former EXAPT-like input language has now disappeared and is no longer valid for Technovision. This has the effect that the post-processor reading CSG Neutral Files is no longer in operation.

On the other hand, the processors were upgraded to be able to handle polyhedron representations of solids. Intersystem tests with Proren, Romulus, and Catia were successfully accomplished.

The so called KOMPIN/KOMPUT interface format has been changed. This caused some changes in the reading part of the pre-processor and in the writing part of the post-processor. A positive change in the KOMPIN/KOMPUT interface is that tori and spheres now can be handled by the system.

The processors for GDS (Geometrical Design System) are able to read and write CAD*I Neutral Files version 3.3.

Cycle-tests and intersystem tests with Bravo3 have been performed successfully (see Fig. 2.5-3). Models which utilize the advanced capabilities of external referencing, instancing, and parametric modeling were produced and tested in cycle tests and (as far as other systems could receive them) in intersystem tests. Those are areas not handled by STEP.

In December 1987, the 2nd revised version of the CAD*I test model library was distributed to all WG2 partners. As new items the library now also contained models with conical curves and surfaces in the boundary representation.

Also in December 1987, a converter program taking CAD*I neutral files version 3.2 (B-rep) as inputs and producing STEP files (according to the St. Louis edition) was prepared and distributed as well as a revised and upgraded version of the file handler program.

In February 1988, two models were generated in Technovision and neutral files were produced for the use in WG6. Later on, it was possible for GfS to recover those models in a FE mesh generator and make actual FE calculations on them.

Work related to assessment of file exchange was started. In March 1988, initial tests checking the correctness of recovered models were performed.

2.6 Results of solid model transfer tests during the 3rd International CAD*I Workshop in Copenhagen

2.6.1. Purpose

In parallel session A of the 3rd International Workshop on CAD Interfaces a demonstration of solid model transfer was presented by connecting Technovision, Proren and Romulus.

At the workshop location, the Technovision system was implemented on a work station including a ND 5400 mini-computer, and via an IBM PS-80 communicating to Proren at KfK and Romulus at CIT via X.25 Kermit protocol.

The purpose of the demonstration was to illustrate that modelling could be continued in the receiving system after a file transfer.

2.6.2. The Demonstration

A model (see Fig. 2.6.2-13) was created in Technovision and integral properties (i.e. surface area, volume, and centre of gravity) were calculated. The model was then preprocessed and the resulting neutral file was sent to both Proren and Romulus. A letter (a letter is a special neutral file defined by CAD*I and embedded with other neutral files in a single CAD*I metafile) attached to the neutral file explained the tasks, that the person sitting at the receiving site had to perform.

The first task was to calculate the integral properties in the receiving system. Fig. 2.6.2-14 shows a table comparing the results obtained in the three systems.

As can be seen from the table in Fig. 2.6.2-14, no significant difference was found in the calculation of the integral properties in the model recovered at KfK and CIT.

Next, two cylindrical holes were asked to be inserted in the received model, parallel to the Y-axis and with the axes going through (130, 0, 130) and (-130, 0, -130) respectively. The integral properties of this new model were calculated in Technovision after recovering the model from Proren. The result is shown in Fig. 2.6.2-15.

Finally, the original model was required to be separated into two parts and the one part moved 20 mm along the x-axis. The result was preprocessed and returned to DTH. Here, it was postprocessed and recovered in Technovision (see Fig. 2.6.2-16).

A model operation was issued in Technovision on the recovered modified model sent by ROMULUS for moving the one part back again and then glue the two parts together. The integrated properties were once more calculated and compared with the original calculations. The result is shown in Fig. 2.6.2-17.

2.6.3. Conclusions

The demonstration was, as illustrated in this document, very successful and it looks very promising for future implementations of processors for solid model transfer based on the methodology developed and used within the CAD*I project. It also shows that processor programs on both the sending and receiving ends as well as the transmission procedure are very stable by now.

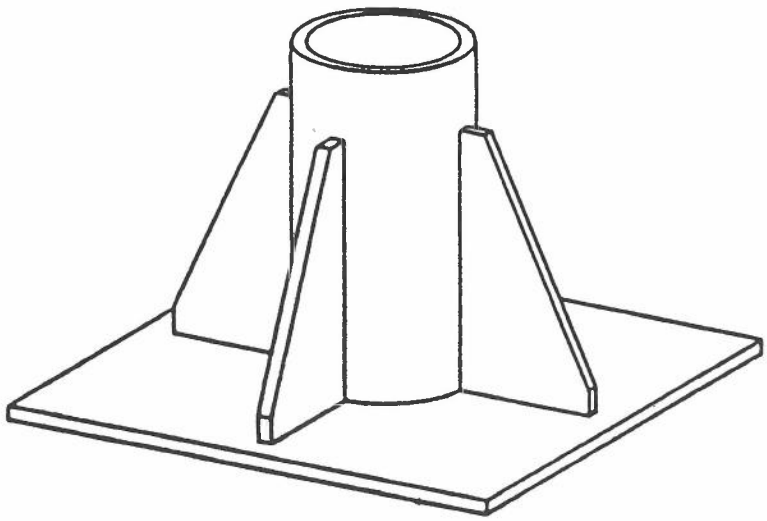


Fig. 2.6.2-13: Test model used for solid model transfer tests during the 3rd International CAD*I Workshop

Integral properties	TECHNOVISION	ROMULUS	PROREN
Surface Area	3.57116E+05	357116.211883	-
Volume	1.38755E+06	1.38755E+06	1.3875496E+06
Center of gravity			
XC	0.	0.	0.
YC	0.	0.000043	0.6E-5
ZC	0.	0.	0.

Fig. 2.6.2-14: Comparison of integral properties of test model

Integral properties	ROMULUS	TECHNOVISION
Surface Area	356864.884471	3.56865E+05
Volume	1.38252E+06	1.38252E+06
Center of gravity		
XC	0.	0.
YC	0.15292	0.15294
ZC	0.	0.

Fig. 2.6-15. Integral properties of modified model.

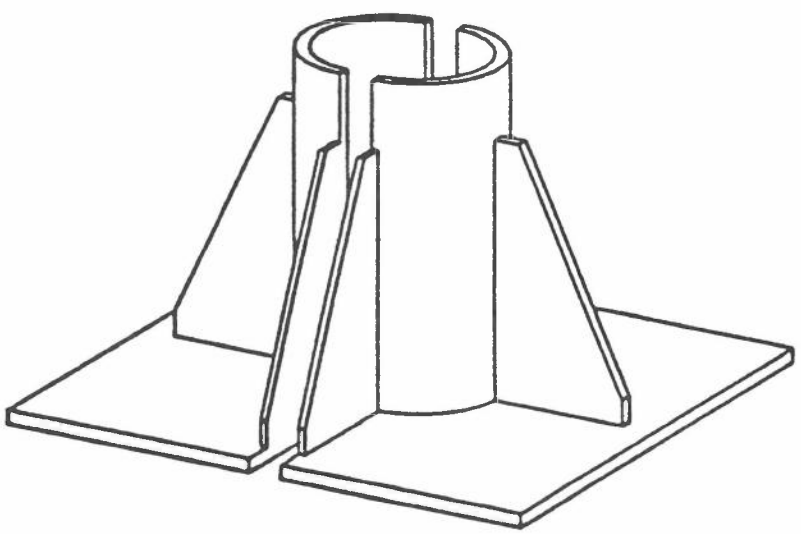


Fig. 2.6-16: Cut in the original model

Integral properties	Original Model	Assembled Model
Surface Area	3.57116E+05	3.57116E+05
Volume	1.38755E+06	1.38855E+06
Center of gravity		
XC	0.	- 1.18E-7
YC	0.	6.57E+5
ZC	0.	7.75E-7

Fig. 2.6-17: Comparing integral properties of original and assembled model.

2.7 Coordination of European efforts regarding STEP

In order to improve the cooperation of European CAD data exchange experts on the ISO level, a second meeting was held between CAD*I and SET. The meeting took place in Paris in December 1987. As a result of the meeting common conclusions regarding STEP were established. These conclusions concerned the following issues:

1) the need for processor implementation,
2) the lack of a block structure on the file for representing complex entities,
3) the lack of a sub-bloc (SET) or property (CAD*I) construct to reflect information that is attached to entities rather than an independently existing entity itself,
4) referencing rules on the file,
5) external references and libraries.

In the meantime, due to the European influence on the STEP development, the STEP specification has been enhanced such that satisfactory solutions were obtained for issues 2, 4, and 5.

2.8 Exploitation

NEH has been in contact with Control Data A/S, Denmark. The purpose of the contact is to try to establish an agreement on sale of our prototype processors and NEH consultancies to CDC and/or CDC Icem Solid Modeler users.

Furthermore NEH uses the experiences and results from their CAD*I participation as background information in NEH marketing and consultancies.

KfK is in negotiation with two vendors of the Proren system regarding a common approach to further development and enhancement of CAD*I processors as a step towards the availability of commercially available STEP processors for Proren.

2.9 References

/1/ I. Bey, J.Leuridan, ed.
CAD*I CAD Interfaces, Status Report 2, KfK-PFT 121, March 1986

/2/ I. Bey, J.Leuridan, ed.
CAD*I CAD Interfaces, Status Report 3, KfK-PFT 132, March 1987

/3/ I. Bey, J. Leuridan, ed.
CAD*I CAD Interfaces, Status Report 4, KfK-PFT 139, March 1988

/4/ E.G. Schlechtendahl, ed.
Specification of a CAD*I Neutral File for CAD Geometry. Wireframes, Surfaces, Solids. Version 3.2. Research Reports ESPRIT. 2nd Revised and Enlarged Edition. Springer-Verlag, 1987.

/5/ E.G. Schlechtendahl, ed.
Specification of a CAD*I Neutral File for CAD Geometry. Wireframes, Surfaces, Solids. Version 3.3. Research Reports ESPRIT. 3rd, Revised Edition. Springer-Verlag, 1988.

/6/ N. Brändli: Korrekter CAD-Datenaustausch. Kongress Infina 89,Informatik für die industrielle Automation, Karlsruhe, February 15-17, 1989,to be published.

/7/ M. Wirsing et al.: On Hierarchy of Abstract Data Types / Acta Informatica, Vol. 20 (1983), S. 1-33

/8/ H.A. Klären: Algebraische Spezifikation - eine Einführung. Springer Verlag -Berlin Heidelberg New York 1983

/9/ B. Möller, A. Tarlecki, M. Wirsing: Algebraic Specification with Built-in Domain Constructions / DRAFT VERSION

/10/ A. J. Korenjak, J. E. Hopcroft: Simple Deterministic Languages /7'th Symposium on Switching and Automata, Berkeley (1966) S. 36-46

/11/ I-DEAS Technical Overview, 1986

/12/ I-DEAS User Guide Volume 2

3. Working Group 3: Surfaces

3.1 Introduction

The past half year has been a period of consolidation with most of the effort being concentrated upon software development and testing rather than on new concepts. Opportunities have been taken whenever they have arisen to publicise our achievements to a wider audience, and during this period we have made five separate presentations at conferences and meetings. On each occasion the relevance of our conversion and approximation work to surface data exchange problems in the wider context of IGES and STEP as well as the CAD*I neutral file has been highlighted. First steps have also been taken towards commercial exploitation of our results.

Some further developments have been made to the concept of orthogonal approximation, these are in the areas of efficient computation of the basis functions and in applying tangent constraints to surfaces as well as curves. Further details of these are given in the enclosed technical note.

We have identified two remaing problem areas in approximation theory relevant to practical surface data exchange. We hope to make some initial investigations of these in the next half year. The first problem involves the use of rational functions as approximations to parametric curves and surfaces, this could be useful when transmitting data from a high degree rational or polynomial system to a low degree rational B-spline based system. The second problem arises when bounded surface data consisting of a surface and bounding curves is to be communicated to a system lacking the bounding capability. This could be solved by approximating the bounded region by a simple parametric surface.

3.2 Results

During this period a paper 'Some Comparative Results in Conversion and Approximation' was prepared showing the results obtained with our prototype software for B-spline - Bezier - Polynomial conversions and some comparisions between the Chebyshev polynomial approximation method and an alternative method installed at BMW. Due to the late date of the previous review meeting this paper was included with the previous interim report. Since that time some effort has been expended in preparing this software for commerical exploitation. A special version of the B-spline - Bezier conversion software has been delivered to a British software company. A demonstration version of software for conversion from VDA-FS to IGES B-spline format has been delivered to BMW. This software consists of two subroutines, one for curves, the other for surfaces, for conversion from explicit polynomial

coefficients in VDA-FS type format to an IGES B-spline type format representing a piece-wise Bezier curve or surface. In each case a 0-1 parameterisation of each segment or patch is assumed, and, because of computational problems in confirming higher order continuity from VDA-FS data, only C0 continuity is assumed. The subroutine specifications are included as appendix A to this report.

Work on the orthogonal approximation method has been concentrated on methods to improve the computational efficiency and on extending the CI continuity preserving approximations from curves to surfaces. When implemented this will enable an automatic strategy for surface sub-division to be developed. Details of this work is included as appendix B.

3.3 WG3 Standardisation Activities

The close involvement of WG3 personnel in national and international standardisation activities has continued. At the national level Cranfield staff have participated in all BSI AMT/4 meetings during the period and have helped to maintain a high level of awareness of the CAD*I project amongst those in Britain concerned with product data exchange standards. BMW staff have continued to be active participants in the DIN activites and have represented DIN at ISO meetings.

Internationally we have been represented at a CEN/CENELEC meeting to coordinate European activities in product data exchange standards and have been involved in all the TC 184 SC4 WG1 meetings which have occurred during the period. In view of the intended publication of the ISO Draft Proposal for STEP version 1.0 in December of this year the pace of the STEP development has accelerated considerably during the period. As the integration process has progressed many new application areas have been incorporated in the document and some significant changes have been made to existing sections, as a result the amount of material to be reviewed has increased dramatically. An extra WG1 SG1 meeting, for which I was chairman, was held in Leeds during September to prepare a corrected set of geometric entities and to prepare drafting and presentation application models. During this time a significant development has been the recognition that the IPIM in the logical layer of STEP is a pure information model and that some consideration must be given to the preparation of a modified version of this for efficient representation on the physical data file. The implication of this for the CAD*I project is that although in places the IPIM is not identical to the CAD*I specification our continued involvement in the ISO activity should help to ensure that when mapped to the physical file the correspondence is much closer.

3.4 Outlook

For the next year WG3 expects to be making significant contributions to the CAD*I project books which will make our results and achievements known to a much wider audience. We plan to produce a significant paper on rational approximations and to continue to develop and exploit our existing conversion and approximation methods. A particular problem which remains to be addressed concerns the approximation of bounded regions on existing surfaces.

It is now clear that the STEP development will not cease with the publication of the first Draft Proposal and we hope to be actively involved in improving this specification and extending its scope. Most of the work we have done on conversion and approximation of surface data is just as relevant to STEP as it is to the CAD*I proposal.

APPENDIX A:

Details of subroutines:

Curves: SUBROUTINE CURCON(NS,A,MS,NC,NK,B,C)

The arrays are dimensioned to accept a maximum degree (= order -1) of 25.

CALLS: SUBROUTINE ELEVAT, FUNCTION COMB

Input:

NS: INTEGER = number of segments in multi-piece curve

A: DOUBLE PRECISION ARRAY (1 D) containing the following information:

A(1) = order of first segment (ord1)

A(2) to A(3*iord1+1) = x-coeffs of first seg
y-coeffs of first seg
z-coeffs of first seg

A(3*iord1+2) = order of second segment (ord2)

A(3*iord1+3) to A(3*iord1+3*iord2+2) = x-coeffs of second seg
y-coeffs of second seg
z-coeffs of second seg

A(3*iord1+3*iord2+3) = order of third segment
.
.
and so on

where iord2 = INT(ord1) etc.

Output:

MS: INTEGER = degree of B-spline curve

NC: INTEGER = number of control points

NK: INTEGER = number of knots

B: DOUBLE PRECISION ARRAY (length 3*NC) containing the B-spline control points.

B(1) = x-coeff of first control point
B(2) = y-coeff of first control point
B(3) = z-coeff of first control point
B(4) = x-coeff of second control point
.
.
and so on

C: DOUBLE PRECISION ARRAY (length NK) containing the knots.

Surfaces:

SUBROUTINE SURCON(NS,NT,A,MS,MT,NC1,NC2,NK1,NK2,B,C1,C2)

The arrays are dimensioned to accept a maximum degree of 25 by 25 for any one patch.

CALLS: SUBROUTINE ELELVAT, FUNCTION COMB

Input:

NS: INTEGER = number of segments (patches) in u

NT: INTEGER = number of segments (patches) in v

A: DOUBLE PRECISION ARRAY containing the following information:

```
A(1) = order of patch (1,1) in u (ordu1)
A(2) = order of patch (1,1) in v (ordv1)

A(3) to A(3*iordu1*iordv1+2) = x-coeffs of patch (1,1)
                               y-coeffs of patch (1,1)
                               z-coeffs of patch (1,1)

A(3*iordu1*iordv1+3) = order of patch (1,2) in u (ordu2)
A(3*iordu1*iordv1+4) = order of patch (1,2) in v (ordv2)

A(3*iordu1*iordv1+5) to A(3*iordu1*iordv1+3*iordu2*iordv2+4)=
                                  x-coeffs of patch (1,2)
                                  y-coeffs of patch (1,2)
                                  z-coeffs of patch (1,2)
.
.
.
and so on (i.e patch order (1,1),(1,2),....,(2,1),(2,2)..)

where iordu1 = INT(ordu1), iordv1 = INT(ordv1).
```

Output:

MS: INTEGER = degree of B-spline surface in u

MT: INTEGER = degree of B-spline surface in v

NC1: INTEGER = number of control points in u

NC2: INTEGER = number of control points in v

NK1: INTEGER = number of knots in u

NK2: INTEGER = number of knots in v

B: DOUBLE PRECISION ARRAY (length 3*NC1*NC2) containing the control points:

```
B(1) = x-coeff of first control point
B(2) = y-coeff of first control point
B(3) = z-coeff of first control point
B(4) = x-coeff of second control point
.
.
 and so on.
```

(here the order is with v increasing most rapidly, then u)

C1: DOUBLE PRECISION ARRAY (length NK1) containing the u knot set.

C2: DOUBLE PRECISION ARRAY (length NK2) containing the v knot set.

APPENDIX B

Technical Note: Some Further Results on Orthogonal Approximation

Orthogonal Basis Functions

An earlier paper described the use of constrained orthogonal polynomials in curve and surface approximation. The polynomials used in that paper were subject to the simple constraint $f(0) = f(1) = 0$. When appropriately applied they enabled piecewise curves to be approximated with C_o continuity from segment to segment and patched surfaces to be approximated with similar conditions. The basis polynomials in use at that time were restricted to a maximum degree of 4 and were derived explicitly by the Gram-Schmidt process. For many applications, particularly when subdivision is needed to meet tolerance requirements a higher degree of geometric continuity is required. When using higher order basis functions their explicit formulation in terms of their coefficients requires a lot of computer storage and is not computationally efficient for their evaluation. This paper suggests a simple recursive algorithm which can be used both for the definition and evaluation of these basis functions. It can be used to produce orthogonal polynomials satisfying all types of constraint at the end points.

If we assume the constraints to be satisfied are that $f(t)$ and all its first $r-1$ derivatives are zero at $t=0$, and that $f(t)$ and all its first $s-1$ derivatives are zero at $t=1$ then define $v_0(t) = t^r(t-1)^s$

$v_0(t)$ is then normalised to produce $u_0(t)$ such that

$$\int_0^1 (u_0(t))^2 \, dt = 1.$$

Let $b_0 = 0$ and for $n>0$ define

$$v_{n+1}(t) = t\,u_n(t) - a_n u_n(t) - b_n\,u_{n-1}(t)$$

If we assume inductively that $u_n(t)$ is orthogonal to all constrained polynomials of degree $<n-r-s$ then $v_{n+1}(t)$ will have a similar property if a_n and b_n are chosen such that

$$\int_0^1 v_{n+1}(t)\,u_n(t)\,dt = 0 \quad \text{and} \quad \int_0^1 v_{n+1}(t)u_{n-1}(t)dt = 0$$

These conditions are satisfied if

$$a_n = \int_0^1 t(u_n(t))^2 \, dt \qquad (1)$$

$$\text{and } b_n = \int_0^1 t\, u_n(t)\, u_{n-1}(t)\, dt$$

But, since, $v_n(t) = t\, u_{n-1}(t) - a_{n-1}u_{n-1}(t) - b_{n-1}\, u_{n-2}(t)$,

$$b_n = \int_0^1 t\, u_{n-1}(t)\, u_n(t)\, d\, t = \int_0^1 v_n(t)\, u_n(t)\, dt =$$

$$\left|\int_0^1 (v_n(t)^2\, dt)\right|^{\frac{1}{2}}$$

since u_n is defined as $\dfrac{v_n}{(\int_0^1 (v_n)^2\, dt))^{\frac{1}{2}}}$.

Thus if we define $a_n = \int_0^1 t(u_n(t)^2\, dt$, $b_n^2 = (\int_0^1 (v_n(t)^2 dt)$

(2)

and save all values of a_n and b_n; all subsequent basis functions can be defined from the recursive formulae:

$$v_{n+1}(t) = t\, u_n(t) - a_n\, u_n(t) - b_n\, u_{n-1}(t) \qquad (3)$$

$$u_{n+1}(t) = v_{n+1}(t)/b_{n+1} \qquad (4)$$

In practice the evaluation of the orthogonal function coefficients in any particular approximation involves numerical integration which requires the evaluation of the basis functions at a number of discrete points. Using formulae (3) and (4) for this evaluation rather than explicit formulae can significantly reduce computer storage requirements and the number of floating point operations required to evaluate the full set of approximation coefficients. If the end constraints are symmetrical a further saving can be made since in these cases a_n is independent of n.

Curve Approximations With C_1 Continuity

Let $u_0(t)_1$, $u_1(t)_1$, ... , $u_n(t)$ be the set of

orthogonal polynomials satisfying $u_i(0) = u'_i(0) = u_i(1) =$

$u'_i(1)$ and $\int_0^1 u_i(t)\, u_j(t)\, dt = \delta_{ij}$

Suppose $\underline{r}(t)$ is the curve to be approximated.

Let $g_0(t) = 1 - 3t^2 + 2t^3$, $g_1(t) = 3t^2 - {}_2t^3$

$h_0(t) = t - 3t^2 + t^3$, $h_1(t) = t^3 - t^2$ (5)

be the Hermite interpolation functions, then, if we define

$$\underline{r}_0(t) = \underline{r}(t) - g_0(t)\,\underline{r}(0) - g_1(t)\underline{r}(1) - h_0(t)\underline{r}'(0) - h_1(t)r'(1)$$

we obtain a curve which represents the difference between $\underline{r}(t)$ and its Hermite interpolation and has the properties

$$\underline{r}_0(0) = \underline{r}_0'(0) = \underline{r}_0(1) = \underline{r}_0'(1).$$

Following the usual least squares approximation process a best polynomial approximation to $\underline{r}_0(t)$ is given by

$$\sum_{i=0}^{n} \underline{c}_i\, u_i(t) \text{ with } \underline{c}_i = \int_0^1 \underline{r}_0(t)u_i(t)\,dt \qquad (6)$$

and mean square error

$$= \int_0^1 (\underline{r}_0(t)^2 dt - \sum_{i=0}^{n} \underline{c}_i{}^2 \qquad (7)$$

The best approximation to $\underline{r}(t)$ with the same error is then

$$\underline{S}(t) = \underline{r}_0(t) + g_o(t)\,\underline{r}(0) + g_1(t)\,\underline{r}(1) + h_0(t)\,\underline{r'}(0)$$

$$+ h_1(t)\,\underline{r}'(1)$$

$\underline{S}(t)$ will maintain any existing C_1 continuity conditions for segments adjacent to $\underline{r}(t)$.

Surface Approximations with C_1 Continuity

The approach of the earlier paper to first constructing approximations to the boundary curves and then constructing a Coons patch is followed once more but higher order blending functions are required to define this patch.

Following the notation of the earlier paper, let

$\underline{a}_0(u)$, $\underline{a}_1(u)$, $\underline{b}_0(v)$, $\underline{b}_1(v)$ be the best approximations

of the required degree to the boundaries

$\underline{r}(u,0)$, $\underline{r}(u,1)$, $\underline{r}(0,v)$, $\underline{r}(1,v)$ of the surface to be approximated. These curves can be obtained by the C_1 approximation method defined in the previous section.

In order to maintain C_1 continuity around the boundaries the cross-boundary derivatives must also be considered. If, for example, $\underline{r}_v(u,0)$ denotes the cross boundary derivative function $\frac{d\underline{r}}{dv}$ along the boundary v=0 the curve approximation technique can be applied to obtain an approximation $\underline{a}_{ov}(u)$ to this vector valued function as a polynomial in u. Extending this notation and the process to the other boundaries we can then define the Coons patch $\underline{C}(u,v)$ using the Hermite blending functions (5)

$$\underline{C}(u,v) = g_o(u)\ \underline{b}_o(v) + g_1(u)\ b_1(v) + h_o(u)\ \underline{b}_{ou}(v)$$

$$+ g_o(v)\ \underline{a}_o(u) + g_1(v)\ \underline{a}_1(u) + h_o(v)\ \underline{a}_{ov}(u) + h_1(v)\ \underline{a}_{1v}(u)$$

$$-(g_0(u),g_1(u),h_0(u),h_1(u))\ x$$

$$\begin{vmatrix} \underline{r}(0,0) & \underline{r}(0,1) & \underline{r}_v(0,0) & \underline{r}_v(0,1) \\ \underline{r}(1,0) & \underline{r}(1,1) & \underline{r}_v(1,0) & \underline{r}_v(1,1) \\ \underline{r}_u(0,0) & \underline{r}_u(0,1) & \underline{r}_{uv}(0,0) & \underline{r}_{uv}(0,1) \\ \underline{r}_u(1,0) & \underline{r}_u(1,1) & \underline{r}_{uv}(1,0) & \underline{r}_{uv}(1,1) \end{vmatrix} \begin{vmatrix} g_0(v) \\ g_1(v) \\ h_0(v) \\ h_1(v) \end{vmatrix}$$

(8)

$\underline{C}(u,v)$ is then a surface bounded by the approximate boundary curves to $\underline{r}(u,v)$ and coinciding with $\underline{r}(u,v)$ in position and tangent at the four corners.

As before the orthogonal basis functions can be used to approximate the difference

$$\underline{r}_0(u,v) = \underline{r}(u,v) - \underline{C}(u,v)$$

to obtain the coefficients

$$\underline{a}_{ij} = \int_0^1 \int_0^1 \underline{r}_0(u,v)\ u_i(u)\ u_j(v)\,du\,dv.$$

The final surface approximation is then

$$\underline{S}(u,v) = \underline{C}(u,v) + \sum_{i=0}^{m} \sum_{j=0}^{n} \underline{a}_{ij}\ u_i(u)\ u_j(v).$$

The properties of the constrained orthogonal basis functions ensure that $\underline{S}(u,v)$ will coincide with $\underline{r}(u,v)$ at the four corners and that along each boundary, the boundary curve of $\underline{S}(u,v)$ and the cross boundary derivative is defined only form the corresponding boundary data for $\underline{r}(u,v)$. Any adjacent surface patch with C_1 continuity with $\underline{r}(u,v)$ will generate an approximation with C_1 continuity to $\underline{S}(u,v)$.

4.1 Working Group 4: Data Base

4.1.0 General introduction

Companies produce various types of data (administrative, technological...) which are more and more often stored in a database. This enables any authorized user:

- To know what data produced in the company is available to him
- To benefit of the results of the other users.
- To be informed immediately of any modification done by any user.
- To communicate better with users from other departments
- Etc...

However, these functionalities cannot be developed in the CAD area because of the difficulty to manage easily the isolated islands that are the CAD data stored in independent and unsufficiently flexible "FILES".

It is a nearly impossible job to create and maintain relations between pieces of information (entities) of one MODEL stored in one "FILE" and pieces of information (entities) of an other MODEL stored in an other "FILE" .

The aim of WG4 is to demonstrate that a DATA BASE SYSTEM is the appropriate tool to create, maintain and modify these complex relations.

In that way, WG4 specified and implemented a standardized communication interface with a DATABASE SYSTEM.

This communication interface consists of a set of standard subroutines for Fortran applications to WRITE, to READ, to MODIFY, to DELETE... CAD data in a CAD*I DATABASE.

Since users use DBMS to store the definition of complex products, and since administrative information (in particular access rights and management of versions) have a major importance for complex products WG4 considered it is of higher importance to include in the CAD*I data scheme the most commonly administrative data types used in industry .

Therefore, one of the aims of WG4 is to define the main administrative and access control data to be included in the existing CAD*I data scheme (specified by WG1, WG2 and

WG3) and to specify and implement the standard subroutines which operate on these data.

4.1.1 PROBLEMS COMMON TO CAD USERS AND CAD DEVELOPERS

4.1.1.1 Problems met by CAD users when developing programs using CAD data

More and more CAD users (automobile, aerospace industry,...) develop application programs specific to their industry, company, or department, based on CAD data. These application programs represent for these users both a considerable effort of development and an increasing part of their know-how stored in the computer.

Most of the time, these application programs are dependent on a specific CAD system. Indeed, they access (write, read...) CAD data via a library of subroutines provided by their CAD vendor. Since the specifications of these subroutines are not standardized, the application programs are strongly dependent on this library.

Furthermore, it is very difficult for these users to obtain the source code of this library from their CAD vendor (it represents their know-how), and when they can buy it, most of the time it is computer-dependent (for CPU efficiency). This situation makes these CAD users strongly bound to:

- a CAD system
- AND a computer.

4.1.1.2 Problems met by CAD developers when developing CAD interfaces

CAD data schemes proposed by standardization organizations are becoming more and more complex. The range of applications is permanently increasing (solids, surfaces, drawings, finite elements, numerical control, piping, schematic...) and the capabilities in each application are getting more and more sophisticated (new concepts, parameterization...). Therefore, developing an interface software which:

- Has good response time,
- Uses Resources needed (CPU, IO) pertinently,
- Covers the whole domain of information defined by the standard,
- Is flexible enough to integrate new specifications of the standard,
- Is able to make efficient consistency checks on neutral data,

- Includes user options to communicate better with simpler CAD systems

...requires an increasing investment from CAD suppliers.

It is clear that if some parts of software needed by all CAD developers to develop their interfaces could be specified and developed only once, it would represent a great benefit both for CAD vendors and CAD users.

In particular, a library of subroutines to access (write, read...) CAD data stored according to the neutral format would be very helpful for:

- easily accessing data,
- checking data consistency.

4.1.1.3 What is standardized today in the CAD field ?

Today, standardization organizations propose:

- Conceptual level specifications of a CAD data scheme.
- Physical level specifications of this CAD data scheme in a sequential ASCII file.

On the contrary, at present, there is no standardization of:

- the specifications of a library of subroutines to access the CAD DATA stored according to the neutral format.

4.1.2 The proposal of WG4

4.1.2.1 WG4 proposes to take one more step in the standardization by specifying and standardizing a library of subroutines to access CAD data stored according to the CAD*I NEUTRAL FORMAT.

The specifications of standard subroutines are independent of the physical format of data stored. Therefore the subroutines can be implemented equally on:

- A SEQUENTIAL ASCII FILE,
- A HIERARCHICAL DATABASE,
- A RELATIONAL DATABASE,
- ETC . . .

4.1.2.2 WG4 decided to implement a SUBSET of this package on a RELATIONAL DATABASE SYSTEM supporting the standard SQL language.

Such a choice is closely related to the capability:

- Of the standard SQL language to describe flexible and powerful functionalities.
- Of the relational DBMS to execute these functionalities with a high level of performance.

Indeed:

WG4 wants to offer a library of flexible subroutines able to manage complex objects (complex relations + large volume of data) efficiently (CPU).

Thanks to its parameterization functionalities, the CAD*I data scheme has a high capability to define complex relations between objects.

For example: Shape constraints can be defined between different

objects (common radius, common boundary defined by reference to a same curve...)

In order to manage with the same efficiency:

- Models containing more and more complex relations,
- Models defined by a larger and larger volume of data,

CAD developers need to develop a software package to access CAD data which is getting more and more complex. If they continue in this direction they will develop a CAD data access software package that has a complexity more and more comparable with the complexity of a DBMS.

The volume of information managed by actual CAD systems is relatively small. Indeed one must keep in mind that when one sees a picture like Figure 4.1-2, only the skin of the car is defined in the CAD model.

To this end, the implementation of the standard library must be done with a modern and powerful TOOL able to take into account efficiently the increasing complexity of relations and the larger volume of data created by CAD users: A DATA BASE MANAGEMENT SYSTEM.

To perform the necessary tests, validate the CAD*I database interface and the future applications developed in the WG4 with the standard library, WG4 has chosen the RELATIONAL DATABASE MANAGEMENT SYSTEM ORACLE supporting the standard SQL language .

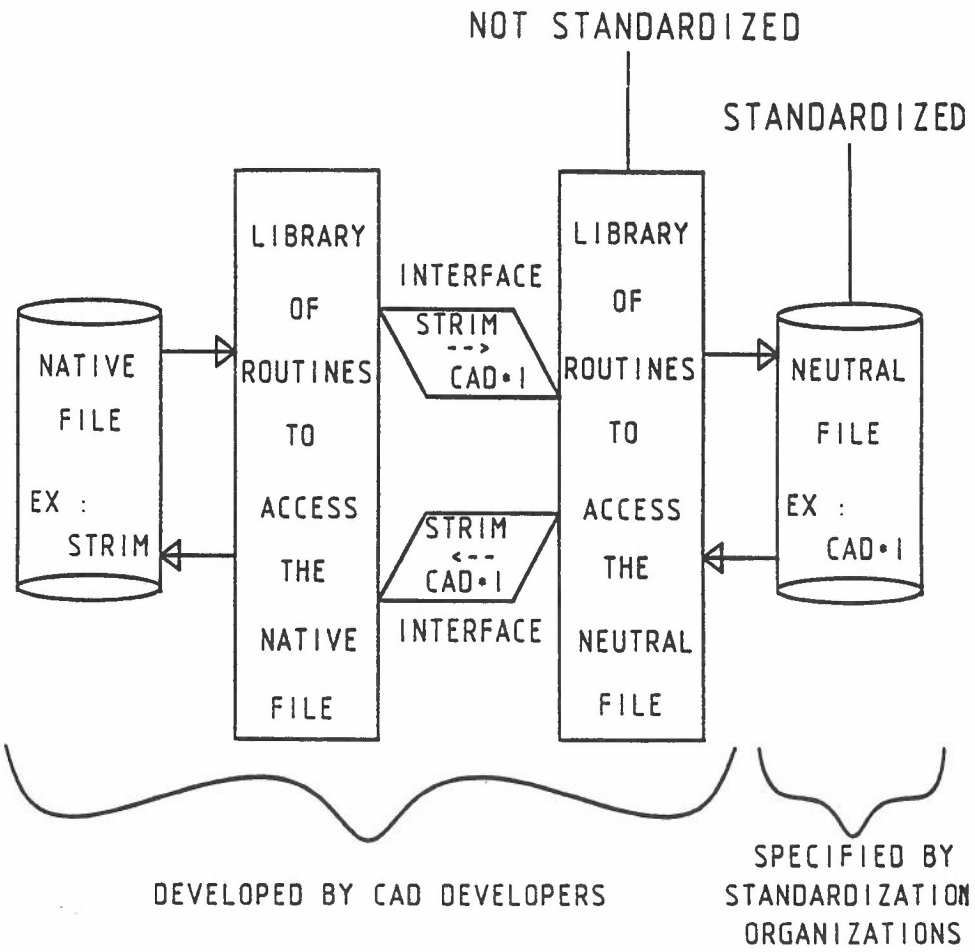


Figure 4.1 -1: General Structure of a CAD Software Interface

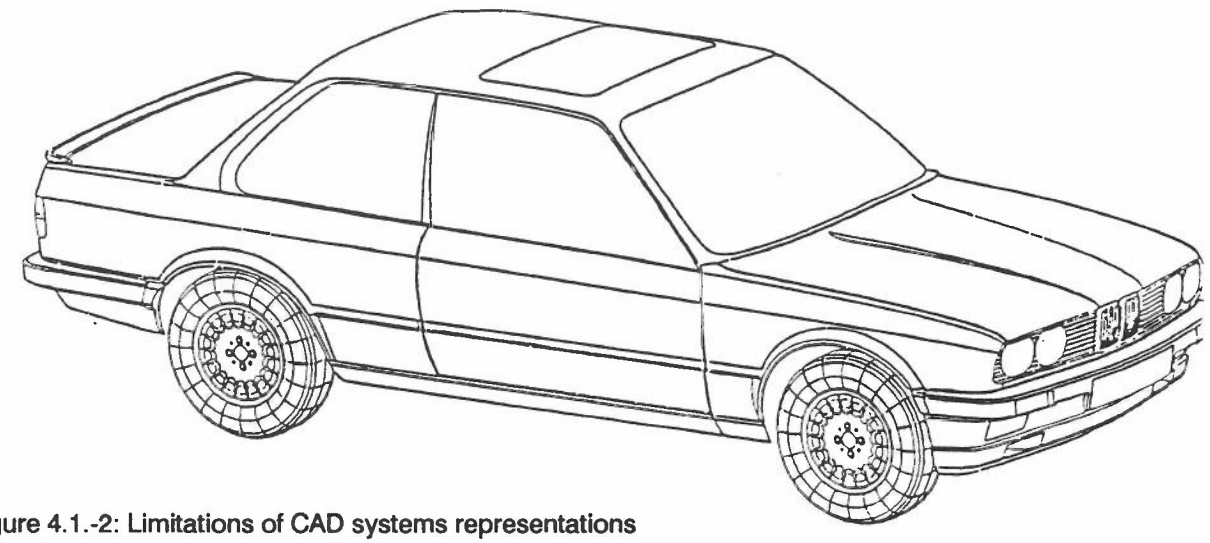


Figure 4.1.-2: Limitations of CAD systems representations

This Database Management System was chosen for the project for the following reasons:

- Relational database system
- The use of a standard language (SQL) to access the data stored
- Portability (IBM, DIGITAL, BULL...)
- Many references in industry.

4.1.2.3 To take one more step in order to represent an ever increasing volume of data, WG4 considers that it is necessary to complete the CAD*I data scheme with some ADMINISTRATIVE information.

Indeed in a database context, geometric information is not sufficient to describe complex objects, and the management of more complex products (many parts defined by different CAD users) requires access control to this data. In addition selection criteria are object oriented criteria and not geometrical ones.

In fact, the same TOOL: DATABASE MANAGEMENT SYSTEM is used by two kinds of users:

- Designers who want to define an object oriented model of their product,

- CAD developers who want a powerful tool to manage the information they produce.

Thus, one of the aims of WG4 is to identify and to include in the CAD*I data scheme some of the most commonly ADMINISTRATIVE DATA used in industry.

For example:

- access rights
- version
- variant
- description of project organization (GROUPS, USERS..)
- who is actually modifying an object?
- etc...

4.1.2.4 Development of Processors

The final goal of WG4 is to make a demonstration program of the results obtained in the Exchange and Storage of CAD*I data and to show that the CAD*I database Interface is a new alternative facing the Sequential ASCII file technique to access and exchange the CAD*I data.

Indeed, at present:

- Two types of storage techniques exist in the project:
  a) A sequential ASCII file technique. The CAD*I data stored with such technique is accessed sequentially via READ and WRITE FORTRAN statements.
  b) A Relational Database Management System technique. The CAD*I data stored with such technique is accessed via non procedural SQL commands.

- Two classes of CAD systems exist in the project:
  a) The class of CAD systems able to access CAD*I data (stored according to all the CAD*I data schema) with the sequential ASCII file technique (BRAVO, EUCLID, CATIA, ...)
  b) The class of CAD systems able to access CAD*I data (In fact only a subset of the CAD*I data scheme at present) with the RDBMS technique (TESTBED and STRIM: developments performed by the WG4).

- One Interface has been developed transferring the CAD*I data (only a subset of the CAD*I data scheme at present) from the CAD*I Neutral File to the CAD*I Database .

At present, only the relational formalism and the SQL language are standardized. The export files are not standardized. Therefore to enable two CAD systems running or two different computers to communicate, an interface between the CAD*I Neutral File and the CAD*I Database is mandatory, for those RDBMS which are not portable on different computers (see Figure 4.1-3).

Consequently, the main developments the WG4 done or to be done are:

- An interface to transfer CAD*I data (only a subset of the CAD*I data scheme at present) from the CAD*I database interface to a sequential ASCII file called the CAD*I Neutral File:

- The CAD*I NEUTRAL FILE <--> CAD*I DATABASE

- The following processors:

- The TESTBED <--> CAD*I-ORACLE Interface
- The STRIM <--> CAD*I-ORACLE Interface
- A demo query system <-- CAD*I-ORACLE Interface.

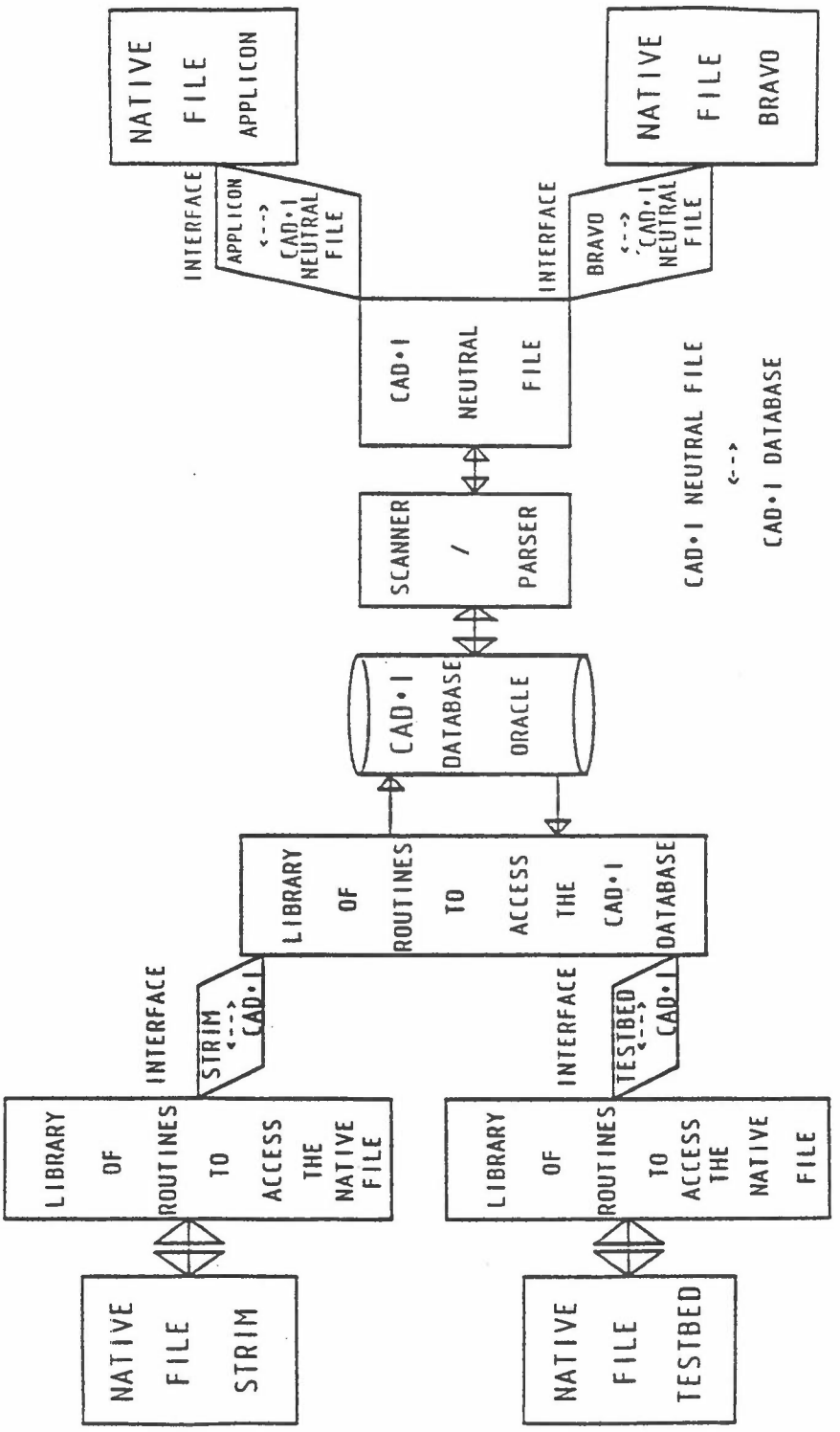


Figure 4.1.-3: Structure of the WG4 demonstration

4.1.3 ACHIEVEMENTS

4.1.3.1 Extension of the Data Scheme with the Administrative Data

4.1.3.1.1 Connection with the geometric entities

In the WG4 CAD*I database we handle geometry in form of closed models as well as in form of models divided into its single points, lines, directions, planar surfaces and topology. A closed model contains geometry in any format. It is impossible to look into the closed model, to recognize any structure in it. This view of data is used to archive and exchange data via a common CAD*I database. To inform about the content, each closed model possesses a set of administrative data.

Also each assembly in the CAD*I database must be described by a set of administrative data, this means administrative data can have a hierachical structure (look at Figure 4.1-4): an assembly is recursively built up by other assemblies and Components. The lowest level possessing a set of administrative data is the Component or the closed model. If various Components are collected to an assembly each Component possesses its own set of administrative data and in addition the whole assembly as a unit possesses a set of administrative data, which references the administrative data of the belonging Component. The term assembly is used only in connection with part or tool geometries. The same proceeding is valid for assemblies consisting of other assemblies and Components.

Summarized, administrative data are used:

- to describe a model
- to control access to a model
- to build classes of models

By the administrative data the technical designer can inform about what is in the CAD*I database, in which status, by whom created, valid or not, does there exist already a new version a.s.o. There exist many possibilities of inquiries.

4.1.3.1.2 The extension of the data scheme: Status

One of the aims of WG4 during the last period was:

- To specify the last version of Administrative data and access rights functionalities to be included in the CAD*I data scheme.

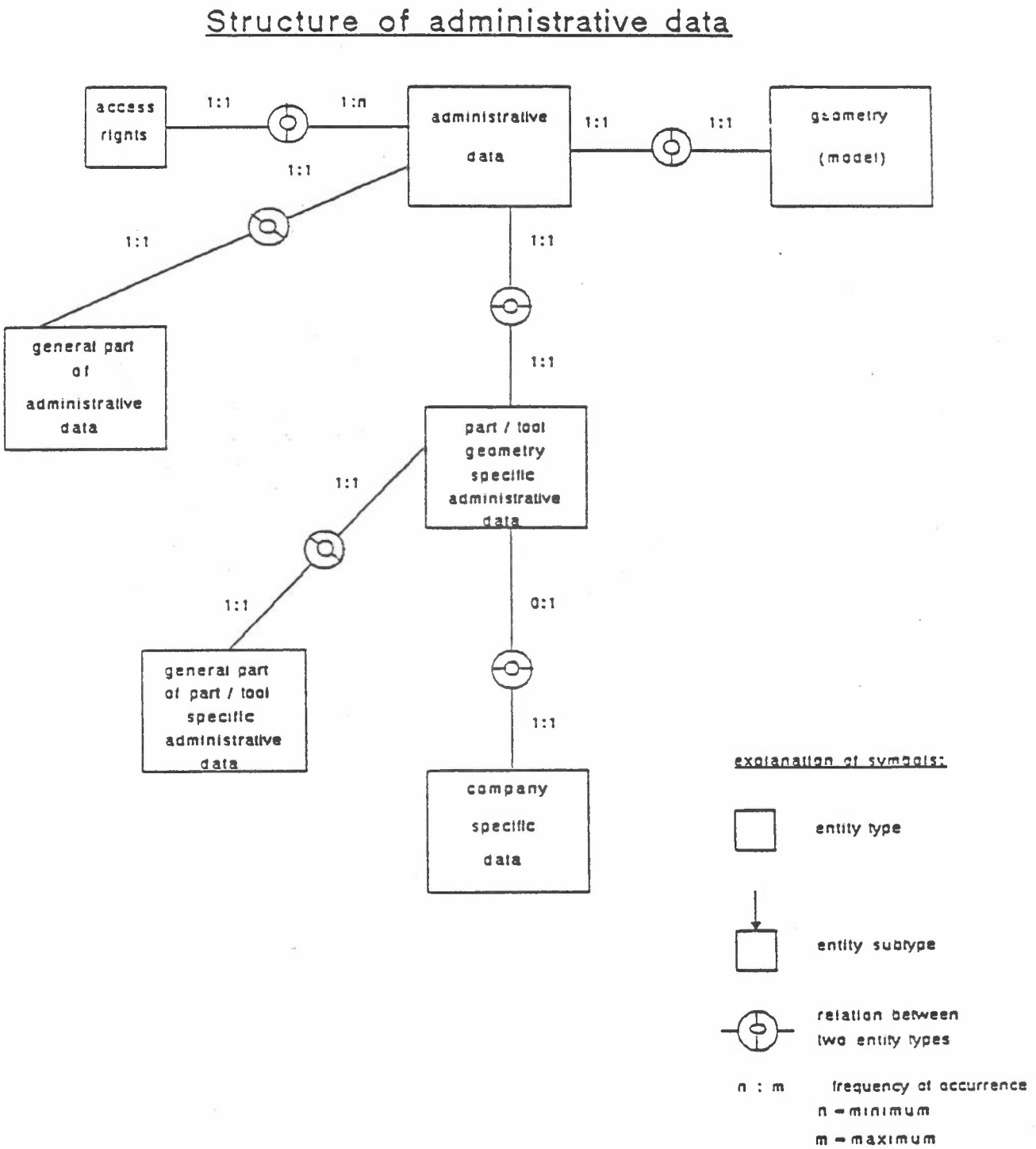


Figure 4.1.-4: Structure of administrataive data

The hierarchy of administrative data regarding assemblies

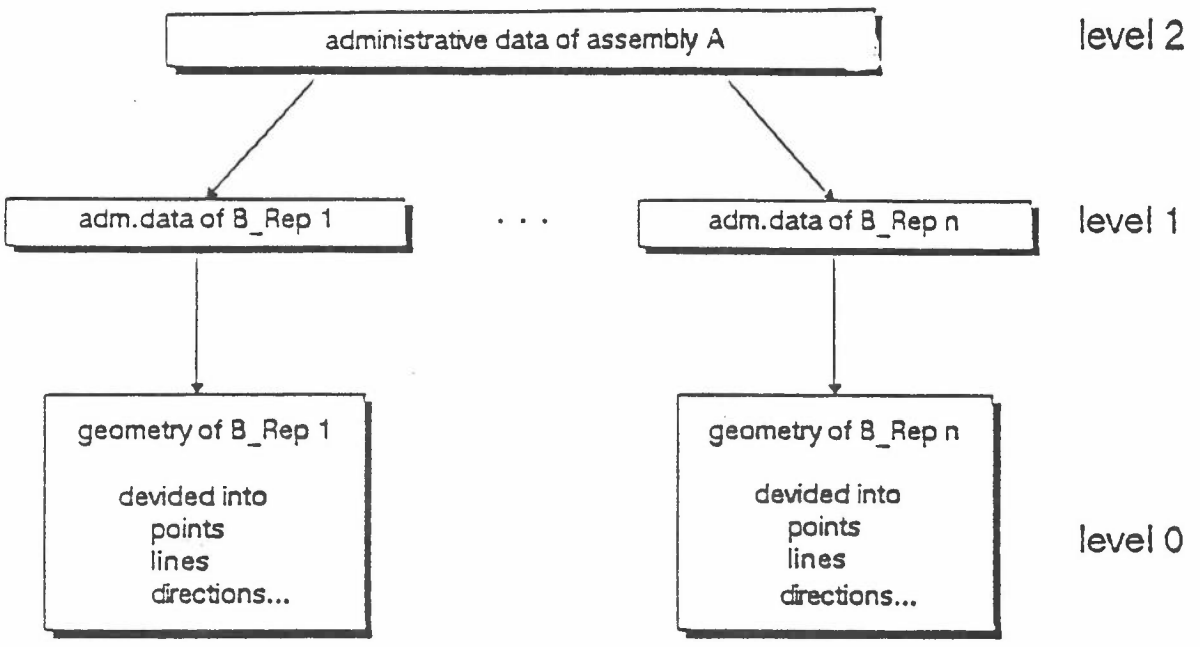
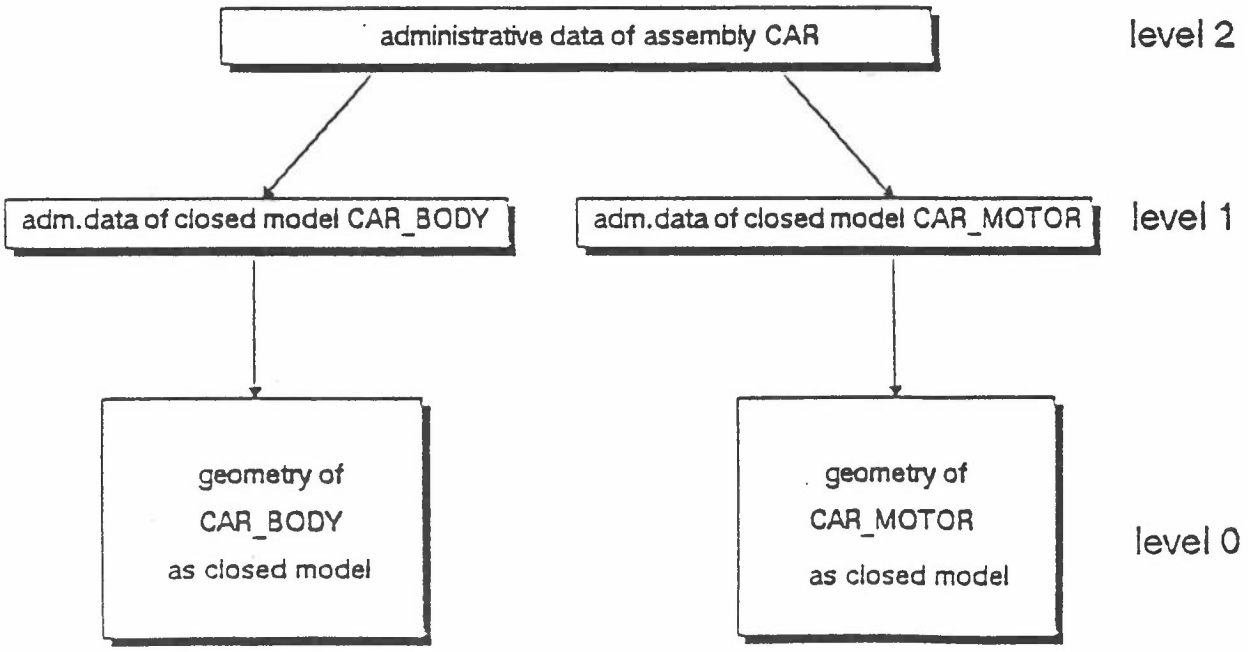


Figure 4.1-5:

The assembly A in this example consists of n B_Reps. Each B_Rep possesses a set of administrative data (level 1) which refers to the geometry of the B_Rep (level 0). The set of administrative data at the assembly level (level 2) refers to each set of administrative data of its B_Reps, but there is no direct reference to a geometry.

Figure 4.1-6:
An assembly can also be used to collect closed model to a unit; in this case we have the following representation (exmaple):



Regarding complex structures of assemblies (if an assembly itself consists of assemblies) data can exist in more than two levels as shown in the example

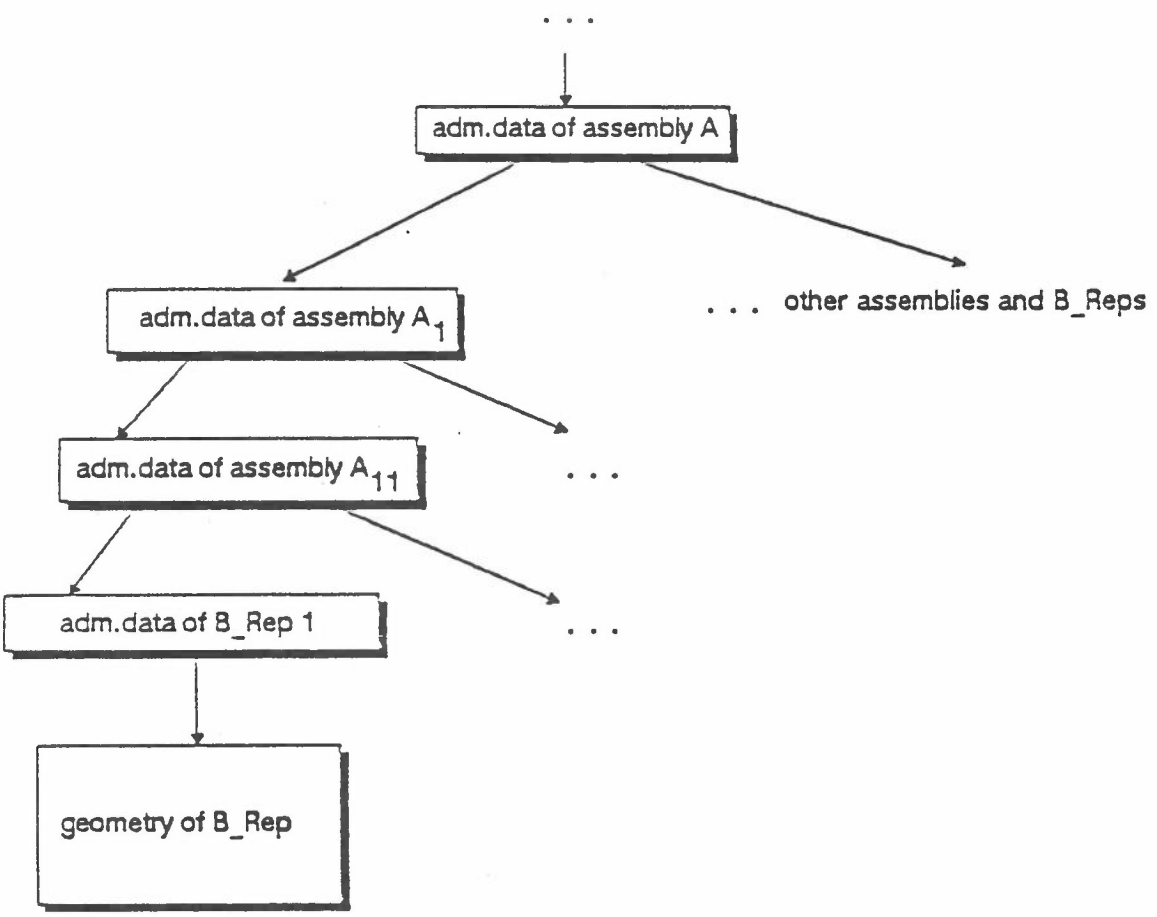


Figure 4.1.-7:
If an assembly is an element (a member) of another assembly, its set of administrative data is referred by the set of administrative data of the superior assembly.

- To specify and implement the routines handling the functionalities related to the Administrative data:
- Routines handling the Administrative data,
- Routines handling the Access_rights,
- Routines handling the Closed_model,
- Routines handling the multi-Assemblies configuration,
- Common routines (buffer management).

- To harmonize the routines handling:
- The Administrative data
- The routines handling geometrical entities(Scope mechanism, selection mechanism, access control...).

4.1.3.2 THE CAD*I DATABASE INTERFACE

4.1.3.2.1 The CAD*I Database interface routines implementation

4.1.3.2.1.1 The CAD*I data scheme implemented

CAD*I data stored in the relational DBMS ORACLE are Boundary representations of solids limited to planar faces and straight edges (see Figure 8 for the description of topological and geometrical information of a B_REP).

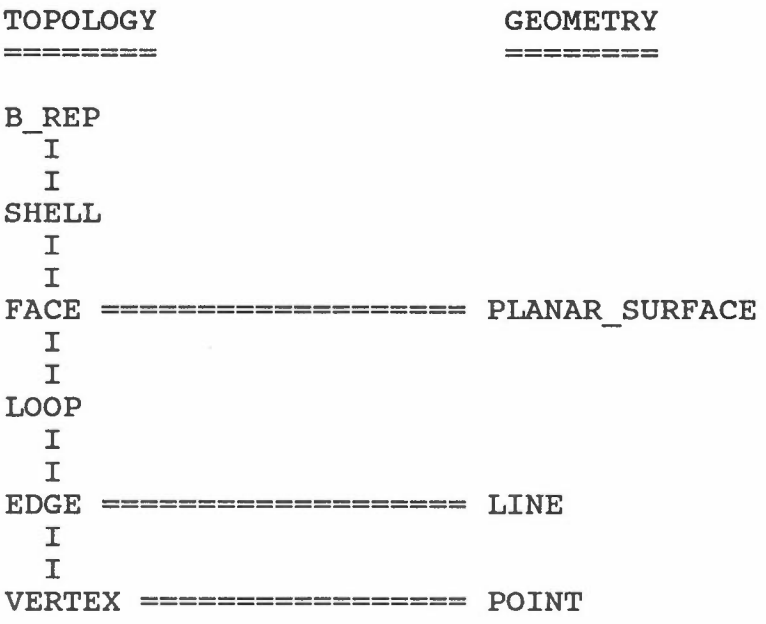


Figure 4.1.-8: Description of topological and geometrical information of a B_REP).

At the physical level, for any CAD*I entity (see the Figure 8), WG4 has defined the corresponding ORACLE table containing the necessaries attributes to describe the entity according to the CAD*I data schema.

Example: The Table of POINTS

SCOPE_PNT	PNT_NAME	X_PNT	Y_PNT	Z_PNT
1	3	0	0	0
1	15	100	0.7	0
1	42	0	0	100

4.1.3.2.1.2 The CAD*I Database Interface Routines implemented

The CAD*I DATABASE INTERFACE Routines are specified for FORTRAN applications and access the CAD*I data stored according to the CAD*I data scheme. They can be implemented equally on:

- A Sequential ASCII file,
- A Hierarchical database,
- A Relational database ...

At the end of January, 1988 WG4 has implemented 56 routines accessing a subset of the CAD*I data scheme defined by the WG1, WG2, and WG3. Figure 4.1-9 shows the main families of routines implemented:

" + " means the routine is implemented for the entity
" - " means the routine is not implemented for the entity

Remark: the CAD*I database interface routines have been implemented on the relational database management system ORACLE.

4.1.3.3 Processors Development

The implemented CAD*I DATABASE INTERFACE package of routines has been distributed to the WG4 partners to allow them to develop and test the following Interfaces:

- the STRIM100 <--> CAD*I-ORACLE interface.
- the CAD*I NEUTRAL FILE <--> CAD*I-ORACLE interface.
- the TESTBED <--> CAD*I-ORACLE interface.

ENTITY	SL 1	IN 2	UP 3	DL 4	LI 5	OP 6	CL 7
POINT	+	+	+	+	+	-	-
DIRECTION	+	+	+	+	+	-	-
LINE	+	+	+	+		-	-
PLANAR-SF	+	+	+	+		-	-
VERTEX	+	+	+	+		-	-
EDGE	+	+	+	+		-	-
LOOP	+	+	+	+		-	-
FACE	+	+	+	+		-	-
SHELL	+	+	+	+		-	-
B-REP	+	+	+	+	+	+	+
ASSEMBLY	+	+	+	+	+	+	+
WORLD	-	-	-	-	-	+	+

Figure 4.1.-9: CAD*I Database Interface Routines implemented

4.1.3.3.1 Development of the CAD*I-ORACLE <--> STRIM 100 interface:

4.1.3.3.1.1 The CAD*I-ORACLE --> STRIM 100 interface

To transfer the whole structure of a B_REP from the CAD*I-ORACLE database to the STRIM 100 native file, WG4 developed an interface software based on:

- The CAD*I DATABASE INTERFACE routines mentioned above and especially the routines to SELECT entities from the CAD*I-ORACLE database.
- The routines to WRITE a STRIM-volume in the STRIM 100 native file.

This interface is composed of two types of modules.

- Modules to handle the transfer of one entity from the CAD*I-ORACLE to the STRIM 100 native file.
- Modules to handle Mathematical functionalities.

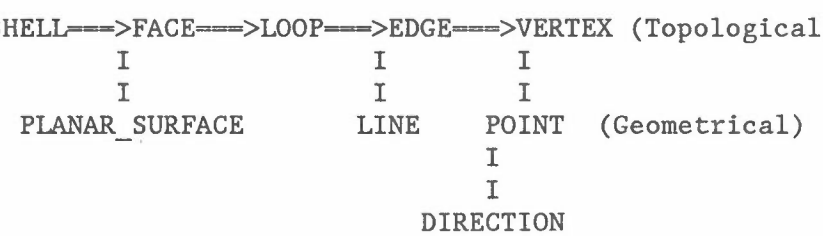
4.1.3.3.1.2 The STRIM 100 --> CAD*I-ORACLE interface

Respectively, WG4 is working on the implementation of the STRIM 100 --> CAD*I-ORACLE interface. This interface is based on the routines to READ the STRIM-volume (polyhedral volume) from the STRIM working area and the routines to INSERT the corresponding CAD*I entities in the CAD*I database according to the B_REP structure (with straight lines and planar surfaces).

4.1.3.3.1.3 Example: The CAD*I-ORACLE --> STRIM 100 Interface

The algorithm to transfer a B_REP from the CAD*I-ORACLE database to the STRIM 100 native file follows the following steps:

- TOP-DOWN Analysis of the CAD*I-ORACLE database according to the B_REP structure described below:



- As soon as one topological entity is well defined (e.g all the geometrical and topological information describing this entity are transferred in the STRIM

100 format), the transfer of this entity in the STRIM 100 format is carried out and its address is passed to the above topological entity.
- At the end of the transfer of all the entities of the B_REP, the STRIM 100 VOLUME is created in the working area of STRIM.

4.1.3.3.2 Development of the CAD*I NEUTRAL FILE –> CAD*I-ORACLE interface software:

This interface software is composed of a WG4-CAD*I-database-postprocessor written in FORTRAN 77 standard and working both in interactive and in batch mode. The CAD*I DATABASE INTERFACE routines are used to insert in the CAD*I-ORACLE database the entities read in the cad*i neutral file.

The postprocessor is built up by different modules, each of which handling a single entity or a special function. Besides the topological and geometrical information of the B_REP there is for the moment no other information (e.g assembly) supported.

The procedure to transfer a B_REP model from the CAD*I Neutral file into the CAD*I-ORACLE Database is the following:

- Generate a B_REP model (which contains only planar_surfaces with straight edges, and lines and points as geometric entities) in a CAD-system.

- Preprocess this model via the system dependant CAD*I-preprocessor into the cad*i neutral file according to the CAD*I format.

- Start the CAD*I scanner/parser software to scan and parse the CAD*I Neutral file.

- Start the WG4-DB-postprocessor to transfer the data into the CAD*I-ORACLE database. There are two different possibilities for naming the B_REP in the CAD*I database:
  - Manual input of the B_REP name
  - Generate a unique name (built up by date and time) by the postprocessor.

Several tests have been made to show the efficiency and the correctness of the WG4-CAD*I-database-postprocessor and especially the CAD*I DATABASE INTERFACE routines.

One main point was, to use the full variety of the CAD*I specification in the case of the geometric entities. That means WG4 tested not only the reference or attribute mechanism of a specific CAD*I preprocessor, but simulated all possible entity descriptions by editing the cad*i neutral file manually.

This process guarantees:

- An easier detection of errors depending on references or attributes.

- The possibility to postprocess a CAD*I Neutral File from any CAD*I preprocessor. The only restriction is the model itself (only B_REP with planar faces and straight edges).

4.1.3.3.3 Development of the TESTBED <--> CAD*I DATABASE interface:

Example: The CAD*I DATABASE --> TESTBED Interface

WG4 developed an interface software which transfer CAD model (with planar_surfaces and straight edges) from the CAD*I database to the TESTBED CAD system.

This interface uses the CAD*I DATABASE INTERFACE routines to select the entities of the created model from the CAD*I database and the EULER operators developed to create the corresponding model in the TESTBED native file.

a) The TESTBED Modeler

The solid Modeler TESTBED was developed at CRANFIELD Institute of technology, England to provide a flexible system for the testing of ideas developed by the CAM-I geometric modeling program.

The TESTBED model is based on a boundary representation and has limited geometric capabilities. Indeed, the surface type implemented are restricted to:

- Planes,
- Cylinders,

and the curves types are:

- Straight lines,
- Circular arcs.

The Native format structure is based on a sequential ASCII file technique of storage and includes:

- The number of occurrences of each entity type (Number of FACES,...)

- The list of attributes values of each occurrence of the entities ordered by entity.

b) The AI Interface

The Application Interface called "AI" is a set of routines to create (via a package of EULER operators) or to interrogate(via a package of interrogation routines) the objects directly by program. These routines are implemented in standard PASCAL.

c) The POSTPROCESSOR developed

The aim of postprocessor developed is to realize a Modeler independent system based on the utilization of EULER operators.

The sequence of EULER operators to construct an object is not unique but the modeling strategy had to be chosen in order to ensure the faster execution time and to test as far as possbile that the source structure is correct.

Limitation of the POSTPROCESSOR:

The method chosen is a step by step method allowing to construct all kinds of polygonal objects with or without holes except objects with holes caused by internal loops.

At present, about 10 models produced are available according to the TESTBED CAD system. (see Figure 4.1-10 showing one of the 10 models produced).

At the end of the fifth six months period, the tasks presently achieved by the WG4 are:

- Specification in HDSL of the administrative data to be included in the CAD*I data scheme

- Implementation of the first set of standard subroutines to handle the main structures of the CAD*I data scheme (WORLD, ASSEMBLY, COMPONENT) and the B-REP representation of SOLIDS (limited to planar FACES and straight edges) on a Relational Database system (ORACLE) supporting the standard SQL language

- Specification of the functionalities and subroutines related to the Administrative Data (Administrative Data, Access Rights, Closed Model, miscellaneous functions).

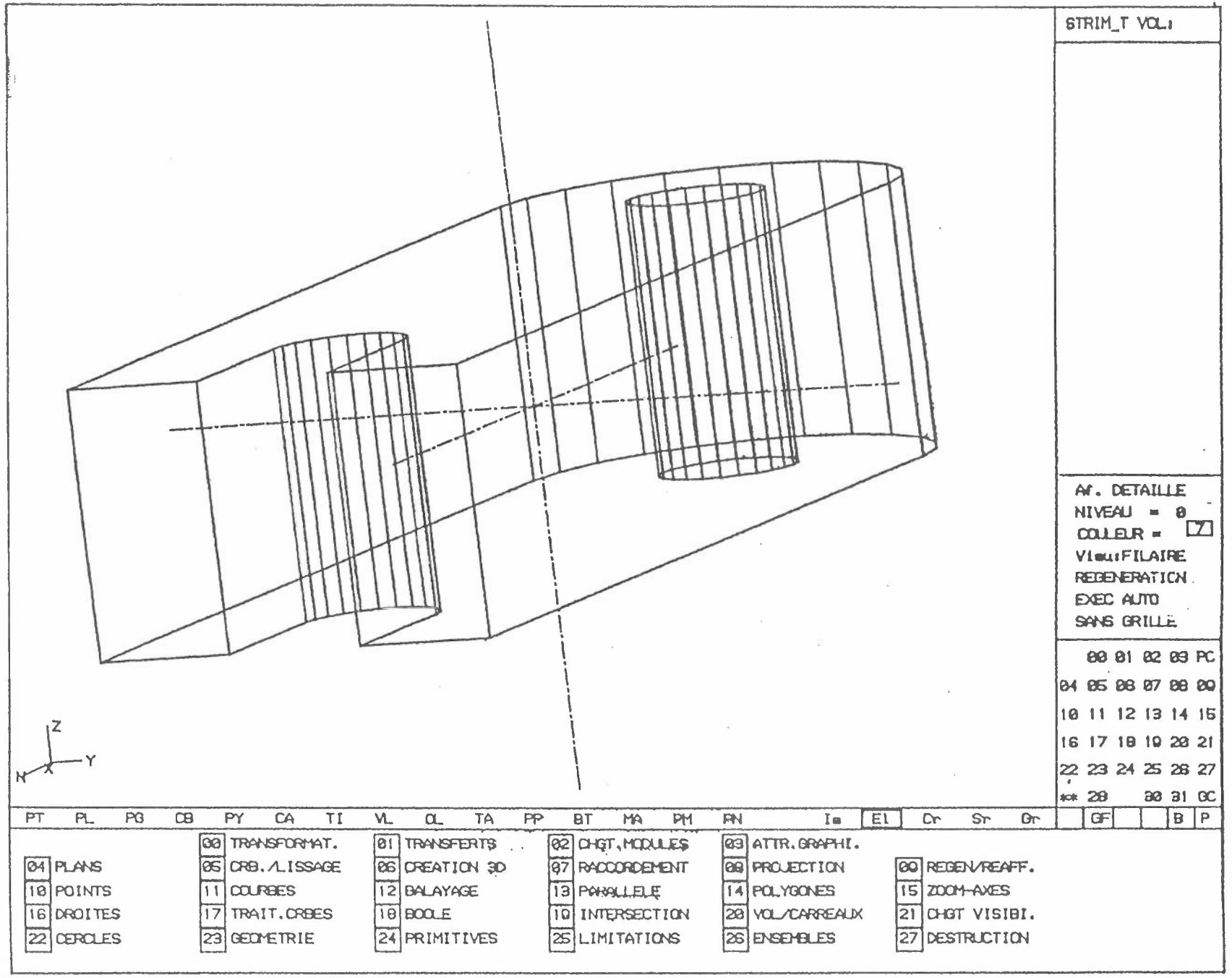


Figure 4.1.-10: TESTBED model transferred to the STRIM CAD system

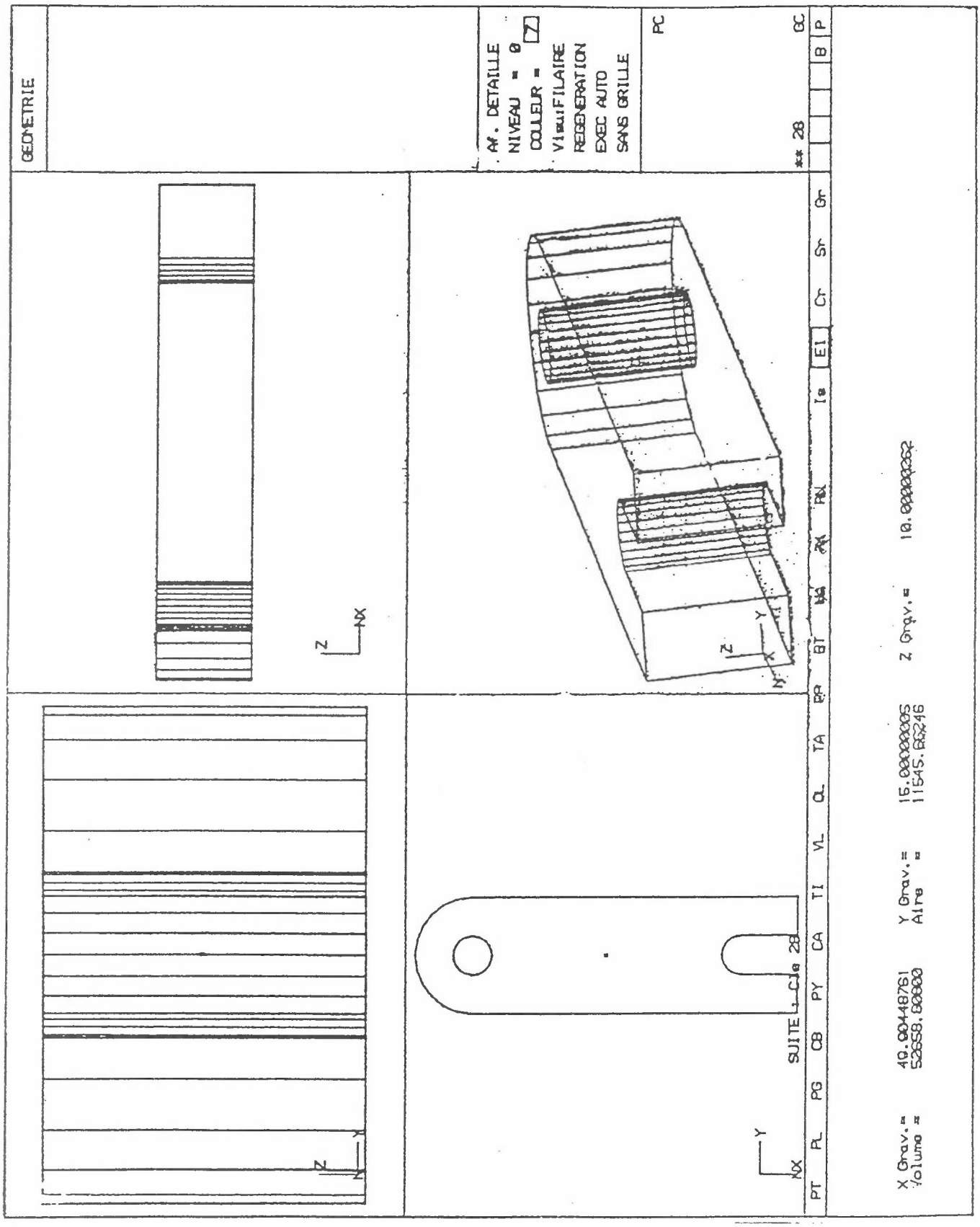


Figure 4.1.-11:
Visualisation at the STRIM screen of the Volume. Surface and Gravity center calculated.

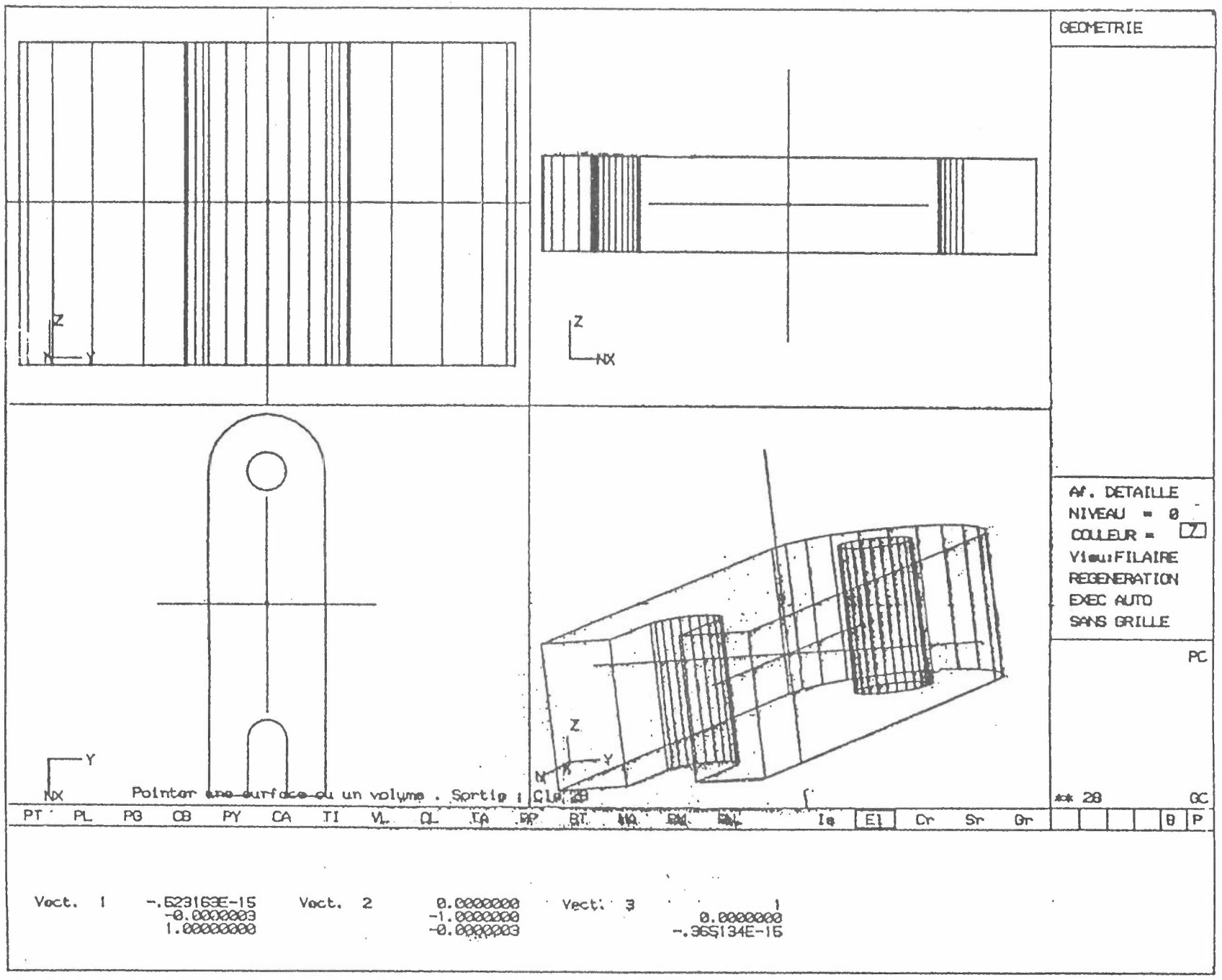


Figure 4.1.-12: Visualisation at the STRIM screen of the inertia center of the volume

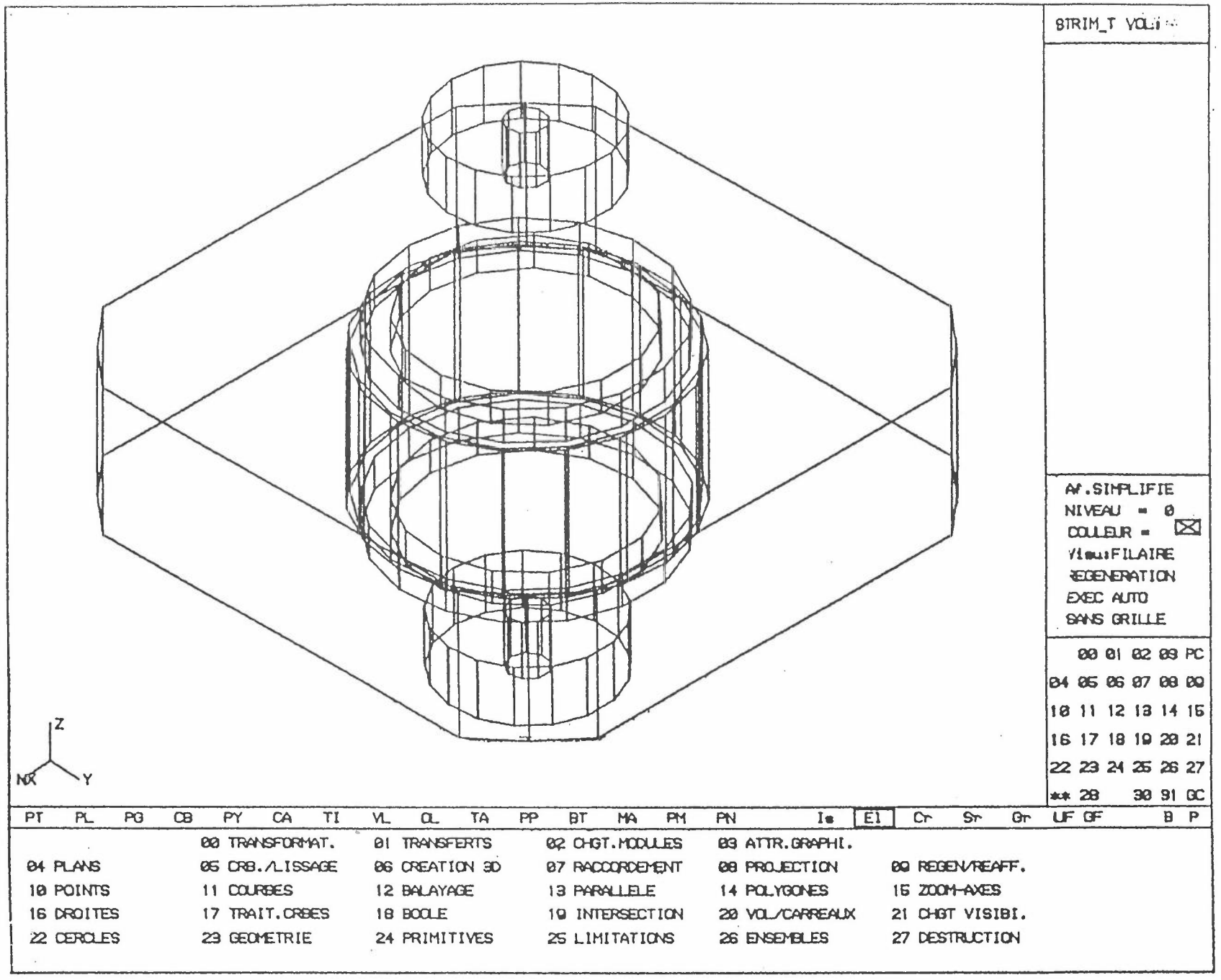


Figure 4.1.-13: STRIM model transferred in the CAD*I database (ORACLE)

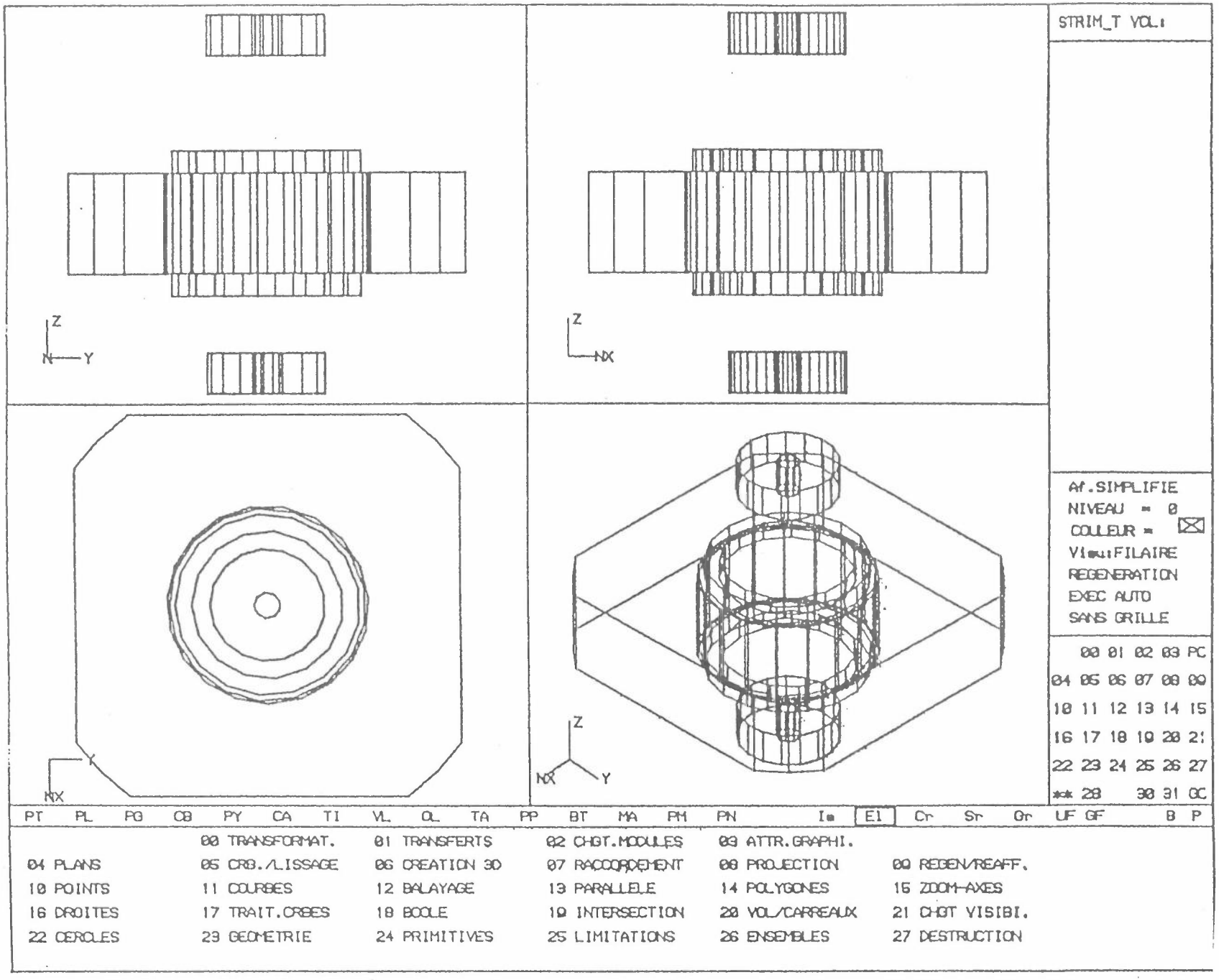


Figure 4.1-14: Example of CAD model transferred in the CAD*I-ORACLE database

- Implementation of these routines with the ORACLE Relational database system supporting the SQL language.

- Implementation of the followings Interface softwares:

  - TESTBED CAD system <--> CAD*I-ORACLE
  - STRIM100 CAD system <-- CAD*I-ORACLE
  - CAD*I-NEUTRAL FILE --> CAD*I-ORACLE
  - A demo query system <-- CAD*I-ORACLE.

4.1.4 MAIN RESULTS AND FUTURE TASKS

First results concerning the WG4 performed work during the last period are detailed on the following pages. The tests made during the interfaces implementations have shown that the performances of the processors developed must be improved. Because these performances are strongly bound to the CAD*I database routines performances, the future efforts of WG4 will act on the ameliorations of the CAD*I database interface routines performances and especially on:

- The Improvement of the physical CAD*I data structure,

- The Improvement of the CAD*I database interface routines performances,

- The Utilization of the DATABASE MANAGEMENT Tools to access data with better performances (Indexes, Clusters).

In order to extend the CAD*I functionalities already implemented, WG4 proposes to continue implementation of new geometrical elements.

For example: - Circles,
- Cylinders...

4.1.5 EXPECTED BENEFITS FOR THE USERS OF THE CAD*I SUBROUTINES

4.1.5.1 Benefits from standardizing routines specifications

a) CAD users can develop their specific application programs independently of any CAD system and any COMPUTER,based on:

   - The standard DATA scheme,

- The standard "CAD*I DATABASE INTERFACE".

b) CAD developers will spend less energy in developing, maintaining, and completing their interface softwares or application programs using the CAD*I DATABASE INTERFACE.

Indeed, the CAD*I DATABASE INTERFACE subroutines are developed only once and are immediately available to application programs to access CAD*I Data.

c) The reliability of CAD data exchange will be improved.

At present a lot of time is lost finding out which of the sending or receiving system is responsible for transfer errors. The "CAD*I DATABASE INTERFACE" can check (at least partially) the consistency of the neutral data to be stored.

Let's see the consequences of applying our concept to IGES; for example:

The routine writing in the Neutral File an IGES LINEAR DIMENSION ENTITY can check whether the intersection between the DIMENSION LINE 1 and the WITNESS LINE 1 of the LINEAR DIMENSION ENTITY (see Figure 4.1-15) corresponds to the arrowhead.

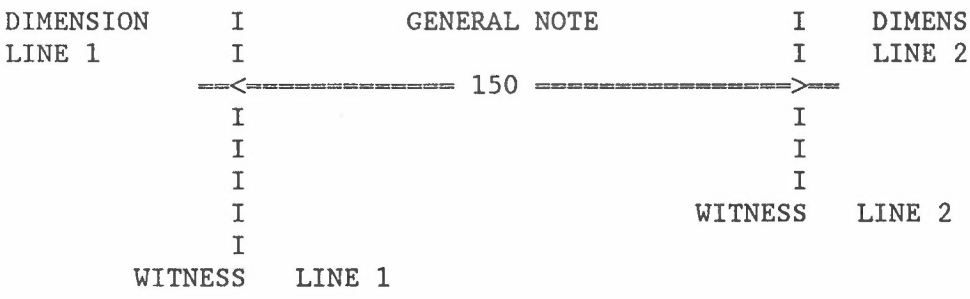


Figure 4.1.-15: Linear Dimension Entity (IGES)

Thus, if all applications use the same subroutines (specified and implemented by standardization organizations) to write the neutral entities, the number of incoherent Neutral Files would decrease considerably.

d) The precision of the data scheme specifications may increase progressively.

Some terms like "CONTINUITY" are used by standards like IGES, SET, VDA but are vaguely defined. "Continuity" requires the definition of tolerance and criteria (algorithm) to decide whether continuity is respected or not.

For example:

The safest way to create a file that contains a surface whose patches respect continuity is to call a STANDARDIZED subroutine (specified and implemented by standardization organizations) that writes a surface and returns an error code if continuity is not sufficient according to the standard.

4.1.5.2 Benefits from implementing on a relational database system

a) With a DBMS we ensure the applications and operations available today are well managed.

At present, in addition to the usual POSTPROCESSOR application using READ operation, and PREPROCESSOR application using WRITE operation, the USER APPLICATION programs using READ, WRITE, UPDATE and DELETE operations are increasingly being developed within companies (see Figure 4.1-16).

However such operations are difficult to implement with classic languages and sequential ASCII file techniques (the most often used at present).

A database management system which allows the handling of a large volume of data, and the associated standard SQL language which has a great flexibility for READ, WRITE, UPDATE and DELETE operations are the appropriate tools for all the programming applications.

For example:

Independently[stored objects can be related, later on, through common parameters that define geometrical constraints. A DBMS is very powerful for UPDATE operations. These parameters can be numerical parameters or geometrical parameters (line, curve...). Example: An object is stored with a reference to a parameter R1 (radius). A second object is stored with a reference to a parameter R2 (radius). Then appears the need to have a common value for the radius of both objects. The reference to parameter R2 can be replaced by a reference to parameter R1.

	PREPROCESSOR	POSTPROCESSOR	USER APPLICATIONS
READ ENTITIES		X	X
WRITE ENTITIES	X		X
MODIFY ENTITIES			X
DELETE ENTITIES			X

Figure 4.1.-16: The Operation Types executed by the different Applications

b) The implementation of the standard subroutines is valid for

- Any COMPUTER with FORTRAN compiler,
- Any RELATIONAL DBMS communicating via the SQL language.

Indeed the "CAD*I DATABASE INTERFACE" subroutines are composed of:

- standard FORTRAN statements,
- standard SQL statements.

c) A relational database system ensures SECURITY and TRANSPARENCY for the users of CAD data since:

CAD data stored in a relational database are accessible to CAD users via the standard SQL language. CAD users can therefore be informed of the evolutions of the implemented data scheme.

d) A database system improves COMMUNICATION between departments within companies:

Data produced is stored in a centralized database in a standard format. Therefore:

- The CAD data produced in the company are immediately available
- The modifications done on CAD data by any authorized CAD user are available for all authorized CAD users as soon as they are achieved.

e) Storage of CAD data in a database system is a prerequisite for the development of sophisticated functionalities:

- Storing different versions of the same object without duplicating information (common entities belonging to the same object can be referenced many times).
- Creation of sophisticated assemblies and selection of a subset.
  To see the placing of a dummy in a car, it is sufficient to select only the geometrical shell of the car and then check that they fit together.
- To retrieve data according to any geometrical or relational selection criteria.
  To study the way of adjusting a door to a side, CAD user selects only concerned entities.
- To allow the definition of complex relations between objects including parametrization possibilities.

4.1.5.3 Benefits from adding administrative data in the CAD*I data scheme

a) The subset of administrative data improves SECURITY, COMMUNICATION and PRODUCTIVITY within companies:

The administrative data protects each geometry against illicit access. Indeed every user is able to define access rights to his own geometry.

Administrative information is available on any object stored in the CAD*I database. (For example: Status of the object, Version, Variant, Owner, who is modifying the object ...)

A sophisticated management of versions can be described. It is possible to know exactly:

- Which geometrical entities were modified between two versions of a same object.
- Who made the modification.
- When the modification was made.
- Why the modification was made in case of modification of the geometry of an object the unchanged geometric entities are not duplicated.

4.1.6 OUTLOOK

In the next year, WG4 will concentrate on the development and optimisation of the demonstration program with the followings activities:

- Optimisation of the CAD*I database interface routines implemented and modifications of the processors developed
  according to these modifications.

- Implementation, tests and optimisation of the STRIM 100 --> CAD*I-ORACLE interface.

- Implementation of the CAD*I DATABASE --> CAD*I NEUTRAL FILE interface.

- Implementation of the new geometrical functionalities (Circles, Cylinders) WG4 propose to add in the CAD*I data scheme.

4.2 Working Group 4: Networks

4.2.1 Introduction

The CAD*I project is launched with the specific goal of establishing methods and techniques for the exchange of product definition data in particular those residing in the data base of a CAD system. In the CAD*I project emphasis has mainly been on the specification of a CAD*I reference schema that defines the product information to be transferred via the neutral representation.

Three main aspects of transferring CAD data are treated within the CAD*I project:

* the external aspect, that deals with the mapping of the CAD*I reference schema onto dissimilar CAD system representations,

* the internal aspect, that deals with the mapping of the CAD*I reference schema onto data base access services (e.g. SQL based data base systems), and

* the physical aspect, that is the mapping of the CAD*I reference schema onto services for the actual transmission of CAD data.

The latter aspect is of interest within this working group which prime objective is to study methods for exchanging CAD data via available networks.

4.2.2 CAD Data Exchange and Network Services

4.2.2.1 General introduction

The actual transmission of CAD data is with todays technology based on the use of computer networks where the nodes in these networks are access points to computer systems. A number of different technologies are available for the interconnection of CAD systems where the choice of an appropriate technology depends on the needs of the environment. The major requirement is the area of coverage that divide networks into so called Local Area Networks (shortly LANs) and Wide Area Networks (WANs).

On top of these networks, that specifies the physical connection of computer systems, several services are available for the control of data transmission from one location to another. The capabilities of such network connections and services should of course be so that they can transfer the CAD information correctly and completely.

In this working group we have chosen to distinguish between file transfer and message exchange:

* File transfer is the passing of the result of a process (e.g. a pre-processor) from one location to another, where the second process (e.g. a post-processor) is initiated and executed employing the received results.

* Message exchange is required when two concurrent processes, that require communication, are being executed simultaneously at separate sites.

Figure 4.2-1 illustrates this by comparing the information transfer via the public mail system and via the telephone system.

The public mail system is characterized by the transfer of complete and self-contained information that do not need any kind of synchronization between sender and receiver during the actual data transmission.

Contradictory to this the transfer of information via the telephone is characterized by many synchronization words (e.g. "hello", "yes", "no") and statements (e.g. "Do you ...?", "Yes, but I ...") that are meant for controlling the actual data transfer and the tasks performed by both sender and receiver.

The term "file transfer" covers the following two situations:

* file transfer between a local and a remote computer where the user has an account on both computers, and

* file transfer between a local and a remote computer, where the user has an account on the local computer and an electronic mail address of the receiver on a remote computer system.

In this working group, the first is called file transfer, and the second, mail.

To summarize, the network services provided for a given application can be divided into two layers, as shown in figure 4.2-2.

Within the area of CAD data transfer, the scope of these network services depends on the actual application. We have listed a few applications that illustrate the application of the diverse services provided by modern networks:

* CAD data transfer within a company
  LAN or WAN combined with file transfer services

* CAD data exchange between a company and its suppliers
  WAN combined with mail services

* Distributed implementation of CAD systems
  LAN or WAN combined with message exchange services.

Such application dependent choices of network services on the individual layers are within the international standardization bodies termed as "profiles".

4.2.2.3 Scope of network access within CAD*I

The main goal of the CAD*I project is as already mentioned mainly concerned with the transfer of information between design systems. Less effort has been directed towards the actual data transmission from one location to another.

The scope of this working group is to study the feasibility of existing methods for file transfer and intertask communication via computer networks. The approach, that has been used, is mainly concerned with the functionality of the different networks. Less interest has been directed towards how the network connections actual works.

The interface to the network is of interest. Comparing with the way of transferring letters via the public mail system the interfaces to that system are the gaps in the postal mailbox and in the receiving mailbox (figure 4.2-3). The message size and format are giving the dimensions of the enclosing envelope. The size, shape, and weight of the complete letter including the envelope have to be such that it can be handled by the mail system. But whenever it is in the mail system we do not really care if the letter is carried with ship, car, airplane, and so on.

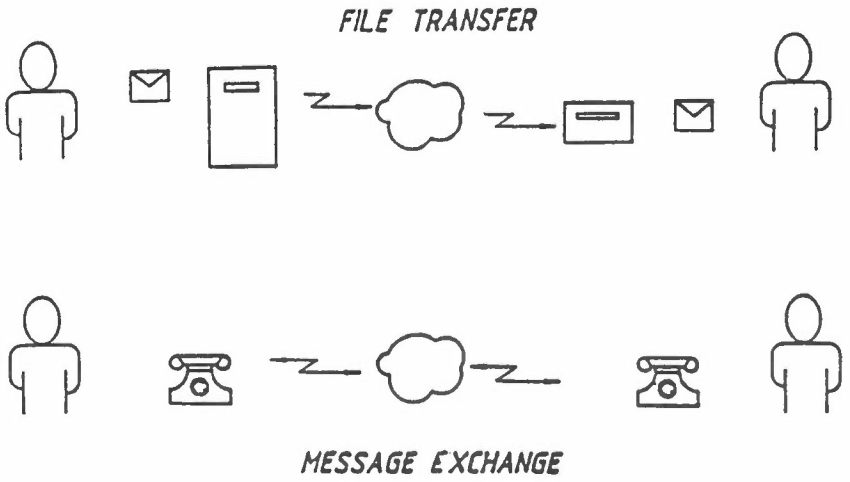


Fig. 4.2-1: Information transfer services

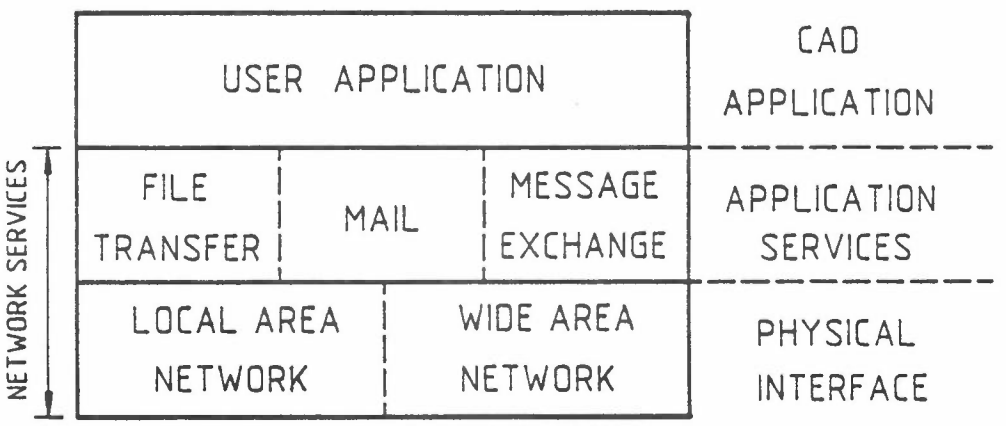


Figure 4.2-2: General two layered reference model for network services

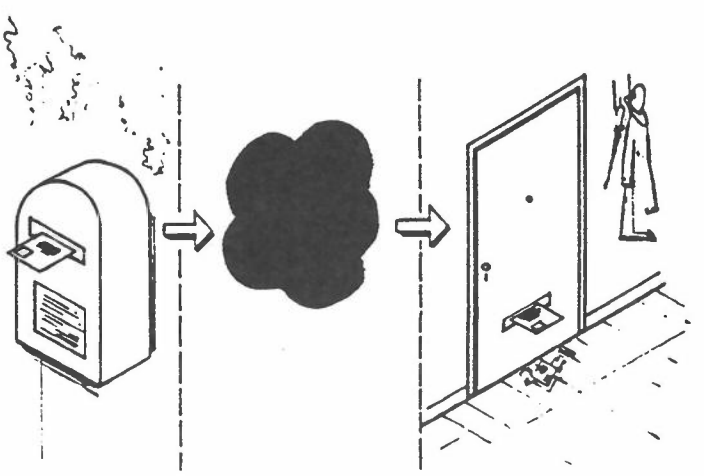


Figure 4.2-3: Interfaces to public mail

As a user of the mail system we do care about the functionality of the system and how easy it is to operate.

In the following we will give an overview of the file transfer experiments that have been performed within this working group. This will mainly be a summary of report /1/, that describes in details the experiments and results of the file transfer tests.

Following this, the distributed implementation of CAD systems will be described as a basis for studying message exchange services within CAD data transfer.

4.2.3 File Transfer

4.2.3.1 Application of file transfer within CAD

Only ten years ago, the increase of productivity in an industrial environment due to the introduction of CAD was measured in number of drawings per week. With the evolution of geometry modeling capabilities in CAD, emphasis shifted towards the transfer of digital representations of products, initially mainly drawings but now also the complete three-dimensional shape /2/.

Without leaving aside the importance of associated information, the transfer of geometrical data itself is of central interest for realizing the full potential of integrated design, planning, and production /3/, /4/.

Some examples are shown here to illustrate that geometric data are generated and used in almost every step of the manufacturing process.

A geometrical model is created during the design process. It describes the shape, topology, and dimensions of the designed product. Later on, during operation and task planning, the geometrical information constitutes the basis for defining geometrical trajectories that machine tools has to follow. This information can again be transformed into data for a program that controls the movement of the actual machine tool. During and after manufacturing, geometrical information related to the product arises from supervisory and quality control activities e.g. as a two-dimensional contour seen from a vision system, or as a three- dimensional shell measured by a laser scanner.

In the examples the transfer of information between the different applications, following the Open System Concept, is carried out by pre- and post-processor programs. Those programs make the local mapping between the native data format of a given application and the neutral representation of the transferred information.

The actual transmission of data is carried out by some kind of physical medium and as complete file transfer. The control of the data transmission is done by a so called file transfer program that makes a local mapping between the native file system and a file transfer protocol. Such a protocol is a convention between computer systems to ensure that data can be exchanged correctly and completely.

4.2.3.2 Purpose of experiments

The file transfer experiments performed within this working group served the following main purposes:

* establish an operational system of CAD file exchange facilities to be used as an infrastructure for the project itself.

* investigate performance and reliability characteristics of public available file exchange facilities according to todays state of practice.

* investigate the upcoming standards for computer network technology with respect to their applicability for CAD data exchange purposes, and plan for future CAD file exchange software architectures based on these standards.

In the following a brief overview of the results obtained by this working group is presented.

4.2.3.3 The CAD*I file transfer infrastructure

In order to provide a file transfer test facility, five types of computer connections have been established:

- The Public Package Switching Network based on X.25 between Cranfield Institute of Technology (CIT), the Technical University of Denmark (DTH), and two installations at the Nuclear Research Center in Karlsruhe (KfK). The X.25 connection was also used within the Bayerische Motorenwerke (BMW), and between

BMW and its suppliers. Also connection to the Eurokom mailbox service at the University of Dublin have been used.

- The Local Area Network at KfK. This private network is a so called Ethernet and two institutes (IRE and PFT) have been connected.

- The private Wide Area Network EARN (European Academic and Research Network) between CIT, DTH, and KfK.

- A direct RS232 connection between two personal computers at DTH has been used for evaluating and validating the file transfer results from network connections.

- Most recently, a X.400 connection has been established at KfK. This, and possible also other X.400 connections, will be the basis for future file transfer experiments within this working group.

The following file transfer facilities have been used for experiments:

- KERMIT as file transfer driver which is public domain software and available on a long range of computer systems. The use of KERMIT in this working group is based on X.25, Ethernet, and the RS232 connection. Furthermore, KERMIT represent a "standard" for most partners within the CAD*I project.

- The file transfer facilities of EARN between CIT, DTH, and KfK. This was already in use between DTH and KfK before the work in this working group was started.

- The file transfer facilities of Eurokom between DTH and KfK. Also this was in use before the start of this working group.

- RVS (Rechner Verbund System) as a file transfer driver which is used between several computers within BMW. The purpose of using this (non-public) software in the project and to compare it with public domain facilities was to identify features which have proven to be useful or required in industrial CAD file exchange practice, but which are not offered by public software.

In figure 4.2-4 is shown the connections established within this working group.

4.2.3.4 Examples from file transfer experiments

As an illustration of the file transfer experiments performed in this working group, the model shown in figure 4.2-5 were transferred from DTH to CIT and Eurokom, and transferred from one IBM PC to another via the RS232 connection within DTH. A more detailed overview of the connections established between the partners is shown in figure 4.2-6.

The model represented in the CAD*I neutral format is of size 32774 bytes. In all three transfer tests KERMIT was used as file transfer protocol. The results are shown in figure 4.2-7.

As a measure for the efficiency of the network connection including the file transfer driver an efficiency factor can be defined as:

$$\frac{\text{actual transfer rate} \times 100}{\text{theoretical transfer rate}}$$

In all three tests, the theoretical transfer rate was 1200 baud.

Assuming that the physical RS232 connection in itself does not consume any time during the transfer, the table shown in figure 4.2-7 allows to calculate the efficiency factor for this connection.

Now, assuming that the time consumption of the KERMIT drivers (both at the sender and the receiver) is constant during all three tests, the efficiency factor can be calculated for each. The results from the three experiments are shown in figure 4.2-8.

As it can be seen from the tables the actual efficiency of the established connections is in general very low. But, it should be noticed that those transfer rates can only be taken as examples. The actual transfer rate depends significantly on the time of day that the network is used.

It should be noticed that the only possible ways of increasing the efficiency of these connections are the following:

1. Increasing the theoretical transmission rate of the network connection. This will in most cases be impossible due to established hardware (e.g. telephone lines) and software.

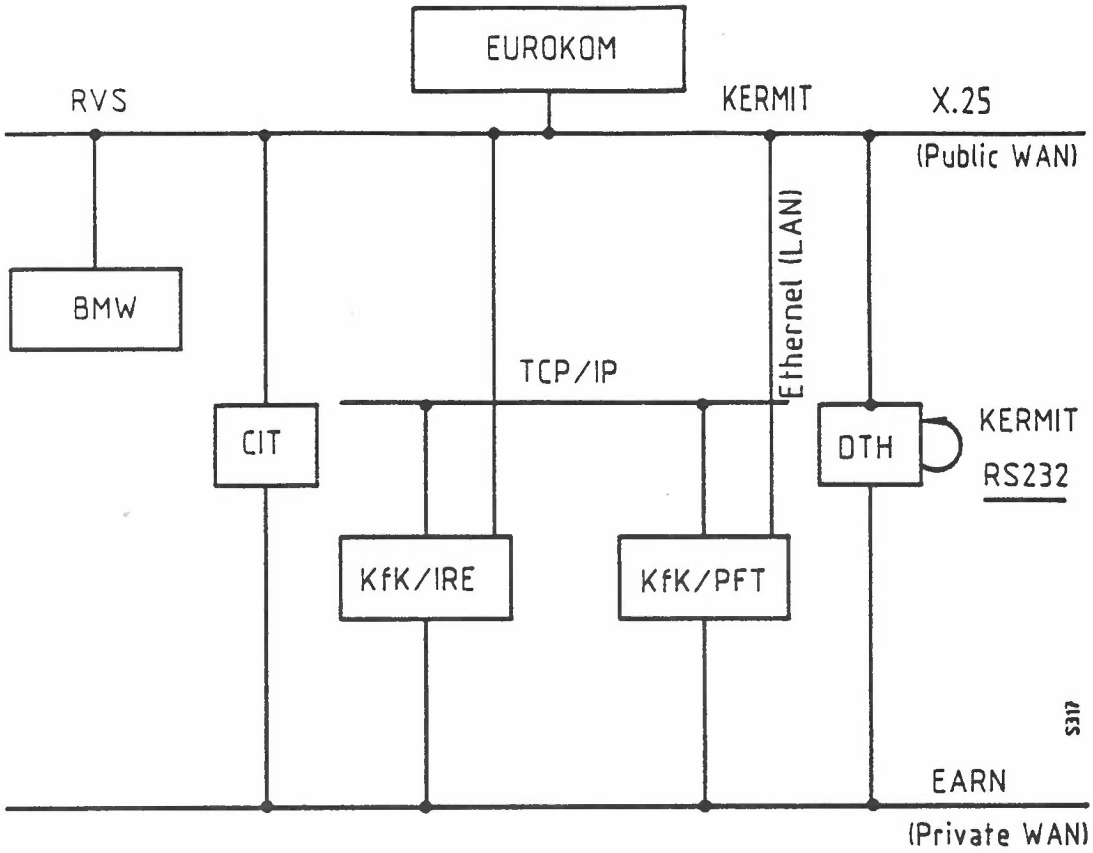


Figure 4.2-4: The CAD*I network

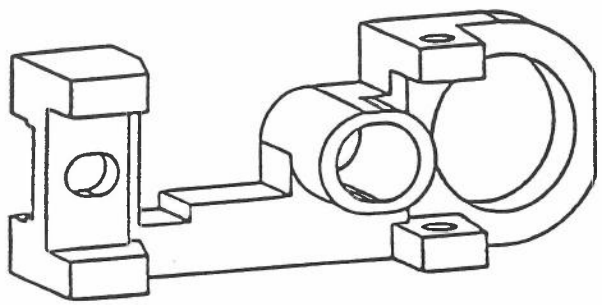


Figure 4.2-5: Test model (Gehauese)

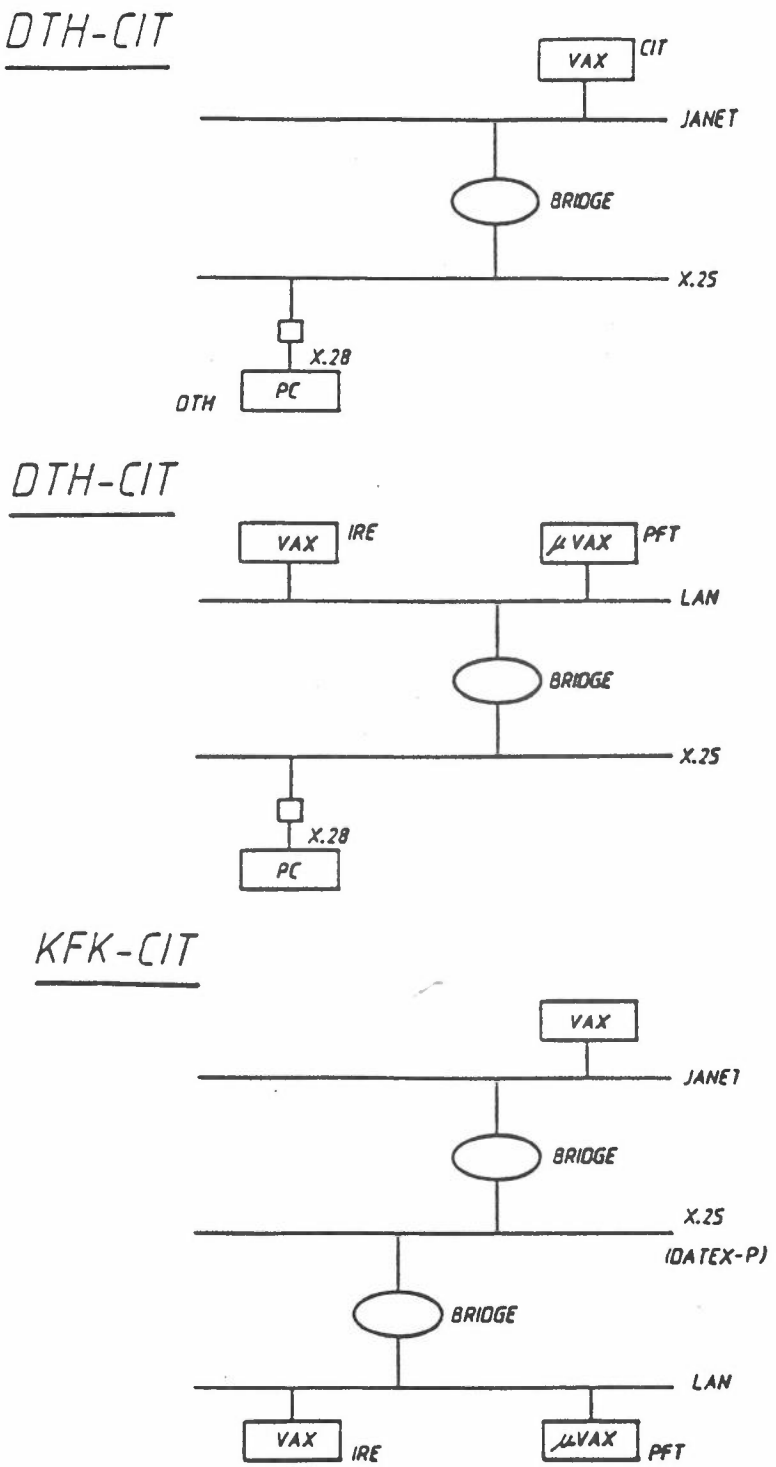


Figure 4.2-6: Detailed overview of partner connections

	TRANSFER TIME (Seconds)	TRANSFER RATE (Baud)
DTH—DTH (RS232)	360	728
DTH—CIT	1545	170
DTH—EUROKOM	980	268

Figure 4.2-7: Results from file transfer experiments

		CONSUMPTION	
	EFFICIENCY FACTOR %	KERMIT %	NETWORK %
DTH—DTH (RS232)	60	40	0
DTH—CIT	22	40	38
DTH—EUROKOM	14	40	46

Figure 4.2-8: Time consumption of network connection (in per cent)

2. Increasing the computing power on the machine handling the file transfer driver (e.g. KERMIT), as most of these include some kind of data compression. As an example KERMIT uses so called run-length encoding. When the same byte appears four or more times in a row in the data, it may be prefixed of the form

<prefix><length><byte>

where the <prefix> is another prefix character, normally tilde (~), ASCII 126. The <length> is a single-character field encoded and can present values from 0 to 94. The <byte> is the data byte, possibly prefixed by control and eight-bit prefixes.

3. Minimize the number of interconnected networks where so-called bridges and gateways consume time for converting between the different protocols used by the involved networks.

4.2.3.5 Results of investigation

4.2.3.5.1 Networks

Public Data Networks

Access to public data networks was established at all partners sites. The connections at BMW, CIT and KfK were all done from the local computer via a local PAD into the national PTTs data network service (in Germany called DATEX-P). The connection at DTH was done from the local computer via a PAD in the Danish PTT data network service, called DATAPAK. This connection is based on the CCITT X.28 Recommendation and in DATAPAK converted to the CCITT X.25 used by the other national PTTs.

The connections had to be made fully operational for file transfer with KERMIT. This involved primarily the coordinated adjustment of PAD and KERMIT parameters on both the sending and the receiving side such that all these parameters were consistent with each other and with the requirements of the local computer environment.

The experience gained during file transfer tests are:

* The file transfer succeeded without transfer errors even for very large files,

* The public data networks are stable and reliable,

* The file transfer depends heavily on the setting of PAD and KERMIT parameters. The inconsistency of these parameters caused problems. The most critical parameters are baud rate and parity,

* The effective transfer rate is rather low (ca. 200-500 baud). This makes the transfer of large files rather tedious,

* File transfer via the public data networks becomes rather expensive for large files (ca. 50 DM per 1MB transferred in Germany).

Local Area Network

In parallel to the analysis of public network facilities an analysis of private LANs available at KfK (Ethernet and DECNET) took place.

The purpose of using these non-public-domain networks in comparison with public networks was to identify the benefits which can be achieved by a higher performance of the network.

DECNET is a special communication facility for DEC computers. It consists of hardware and software components The software components are integrated in the VMS operating system e.g. the file transfer driver TCP/IP. The tests showed that there are no problems in practical use. Performance is absolutely sufficient with respect to stability and transfer rate.

Ethernet is one of the network connections that have been chosen for the OSI reference model. Ethernet (ISO 8802.3) does not contain file transfer capabilities. That means additional protocols for file transfer purposes had to be implemented. The file transfer tests via Ethernet were made with both TCP/IP and Kermit:

* The TCP/IP tests showed the best results as the transfer rate is very high,

* The usage of KERMIT via Ethernet provided still a better transfer rate than on X.25.

Academic and Research Networks

The file transfer tests with these networks were all based on the use of EARN. The tests succeeded without

major problems as long as the exchange experiments were performed between IBM computers. Even though file exchange with non-IBM computers could be performed successfully (in some cases, at least) either within EARN itself or via using gateways to other networks (e.g BITNET) this turned out to be too difficult for practical use. Even within the IBM family of computers difficulties arose when so called "partitioned data sets" were sent from MVS operating system environments to a CMS operating system.

The disadvantages of EARN are:

* Its limitation to IBM computers,

* Its restricted scope of participants, namely universities and research institutions,

* The "store and forward" principle of file transfer. That means a file is "stored" until the data set is "forwarded" to the next node. That principle increases the time for file transfer. Transfer of a file via EARN takes twice to six times the time of transfer via KERMIT on X.25. A more detailed investigation on EARN has been performed by DFN in Germany.

4.2.3.5.2 Network protocols

KERMIT

KERMIT was used as the file transfer protocol for most of the tests that were done in this working group.

The experience gained during file transfer tests via KERMIT are:

* KERMIT is available in public domain for a large number of computers and operating systems,

* KERMIT is easy to implement, but there exist problems with the implementation of the IBM/TSO version,

* KERMIT offers no check points and restart facilities. Such facilities have proven useful in case of network congestion,

* KERMIT has no provision for login scripts. That means there is no mechanism for automatic set-up of connections,

* KERMIT requires a special format for the input files. Special file formats such as containing Fortran Carriage Control require the use of conversion routines. Binary file transfer works only occasionally. However, some KERMIT implementations have facilities for setting file types such as text and binary files.

* The command structure of the KERMIT software for different operating systems is not uniform.

RVS (Rechner Verbund System)

Different test data sets (IGES- and VDAFS-files) were inter- changed via RVS by using IBM computers on both sites. The file size varied from 0.2 to 5 MByte. The DATEX-L (X.21) service was used with a baud rate of 9600.

The following experience could be made:

* even big files could be interchanged without problems,

* the use of data compression reduced the file size by 30 to 50 percent,

* the effective transmission rate for the compressed files were 6000 to 7000 bits/second,

* data link set up and termination was made automatically.

4.2.3.6 Upcoming standards

4.2.3.6.1 Introduction

Upcoming standards for computer network technology have been investigated with respect to their applicability for CAD data transfer. Within the area of network technology the development of standards is based on the so called ISO Open System Interconnection concept. Seven layers are specified in this concept for the data communication between dissimilar computer based systems.

The three lower layers specifies the physical connection of computer systems, and on the four upper layers services are provided for user applications. One such application is the CAD data transfer.

Figure 4.2-9 shows the existing standards and draft proposals that are presently available on the 7 layers of the OSI model.

4.2.3.6.2 File Transfer, Access and Management (FTAM)

FTAM is part of the system of standards constituting the "Open Systems Interconnection, Information processing systems standards". The standard is still under development in ISO at the level of a Draft International Standard. The standard is part of the Application Layer, (layer 7) of the seven layer OSI reference model. The standard forms the User Element or Application Process Interface to the 7'th layer together with a set of other protocols known as the Specific Application Service Elements.

The FTAM specification belongs to a group of basic standards for information processing and does as such not specify any unambiguous set of services/functions that a FTAM interface must support. However the standard defines certain service classes, e.g. The Transfer Class, The Management Class, The Access Class, etc.. At the most basic level the functions defined are grouped into functional units. An implementation must support a functional unit either completely or not at all. Furthermore mechanisms are defined which allow negotiation of functional units when the FTAM connection is initialised. Despite the standard's description of classes and functional units, it still leaves a set of decisions to be made on the user leading to incompatibility. These problems are expected to be solved by sets of functional standards. MAP and TOP both specify a set of functional units of FTAM: "a Limited File Transfer and Management Class", and thereby defines functional standards.

Also ISO specify subsets, options and parameters of base standards, the so called ISPs (International Standardized Profiles). Three classes of ISPs are defined:

T: Transport Profiles
specifying the various functions of the OSI layers 1 to 4

A: Application Profiles
specifying application functions (OSI layer 5 to 7) such as FTAM, MHS, Directory Services, Virtual Terminal, etc.

F: Interchange Format Profiles:
specifying formats for the use of IGES, CGM, ODA, etc.

ISO OSI 7-LAYER MODEL

LAYERS	FUNCTION	STANDARDS
USER PROGRAMME	APPLICATION PROGRAMS (NOT PART OF THE OSI MODEL)	ISO 8632. PART 1-4 (GKS)
LAYER 7 APPLICATION	PROVIDES SERVICES DIRECTLY COMPREHENSIBLE TO APPLICATION PROGRAMS.	ISO 8571. PART 1-4 (FTAM) ISO 9040/9041 (VT) CCITT X.400 SERIES (MHS)
LAYER 6 PRESENTATION	TRANSFORMS DATA TO/FROM NEGOTIATED STANDARDIZED FORMATS	ISO 8822 ISO 8823 ISO 8824 ASN.1 ISO 8825 BER
LAYER 5 SESSION	SYNCHRONIZE AND MANAGE DATA	ISO 8326 ISO 8327 ISO 8326 ISO 8327
LAYER 4 TRANSPORT	PROVIDES TRANSPARENT RELIABLE DATA TRANSFER FROM END NODE TO END NODE	ISO 8072 ISO 8073
LAYER 3	PERFORMS MESSAGE ROUTING FOR TRANSFER BETWEEN NON-ADJACENT NODES	ISO 8348. AD 1+2 ISO 8473 + AD 1 ISO 8648 ISO 9542 ISO/IS 8208
LAYER 2 DATA LINK	ERROR DETECTION FOR MESSAGE MOVED BETWEEN ADJACENT NODES	ISO/DIS 8802.2
LAYER 1 PHYSICAL	ENCODES AND PHYSICALLY TRANSFER MESSAGES BETWEEN ADJACENT NODES	IEE 802.3 IEE 802.5 IEE 802.4 CCITT X.21

Figure 4.2-9: Standards in the OSI Reference model

For FTAM the substructure of the Application Profiles is

AFT 1 File Transfer Service
AFT11 Simple (Unstructured)
AFT12 Positional (Flat)
AFT13 Full (Hierarchical)

AFT2 File Access Service
AFT22 Positional (Flat)
AFT23 Full (Hierarchical)

AFT3 File Management Service

A more detailed and introductory description of those profiles is given in /5/.

4.2.3.6.3 X.400 - Message Handling System

The recommendation X.400, Message Handling Systems - System model - service elements, was issued by the CCITT (International Telegraph and Telephone Consultative Committee) of the ITU (International Telecommunication Union).

In X.400, the following message handling services are defined:

- the interpersonal messaging (IPM) and
- the message transfer (MT).

There are additional recommendation for the message handling service MHS:

- X.401, MHS - Basic service elements and optional user facilities
- X.402, MHS - Overall architecture
- X.403, MHS - Conformance testing
- X.407, MHS - Abstract service definition conventions
- X.408, MHS - Encoded information type conversions rules
- X.409, MHS - Presentation transfer syntax and notation
- X.410, MHS - Remote operations and reliable transfer server
- X.411, MHS - Message transfer layer
- X.413, MHS - Message store, abstract service definition
- X.419, MHS - Protocol specification
- X.420, MHS - Interpersonal messaging user agent layer
- X.430, MHS - Access protocol for Teletex terminals

Besides X.400, there is the X.500 series of recommendations concerned with the use of directory. The directory services defined by X.500 capabilities are

useful for a variety of telecommunication services. X.500 describes how directory can be used in message handling. Details are specified in the following recommendations:

- X.500, The Directory - Overview
- X.501, The Directory - Models
- X.509, The Directory - Authentication
- X.511, The Directory - Abstract service definition
- X.518, The Directory - Procedures for distributed operations
- X.519, The Directory - Protocol specifications
- X.520, The Directory - Selected attribute types
- X.521, The Directory - Selected object classes

None of these recommendations specifies the transport technique. This may depend on the implementation. It may be a block oriented protocol, like X.25 (DATEX-P), or line oriented.

X.400 offers itself for CAD file transfer as it does not require access rights to the receiving computer environment. Instead, (as with public mail) the file is sent to the electronic address of a receiving person.

4.2.3.7 The CAD*I Metafile Utility

4.2.3.7.1 A future CAD Data Exchange Environment based on FTAM

As part of the work within this working group, it has been investigated to use FTAM as a future basis for the development of a File Transfer Utility including the pre- and postprocessing tools for CAD*I neutral files.

The application of a flexible and user friendly file transfer utility is one of the future needs that already can be foreseen within CAD data exchange. Many things, that are not relevant for the actual transfer of information, could be hidden for the user. Examples are the dial-up, connect and login procedures for connecting two computer systems.

The ideal use of such an integrated CAD data exchange and file transfer environment would be, as shown in figure 4.2-10, that the user of a pre- or postprocessor simply state "get" or "send" that model from/to "NN".

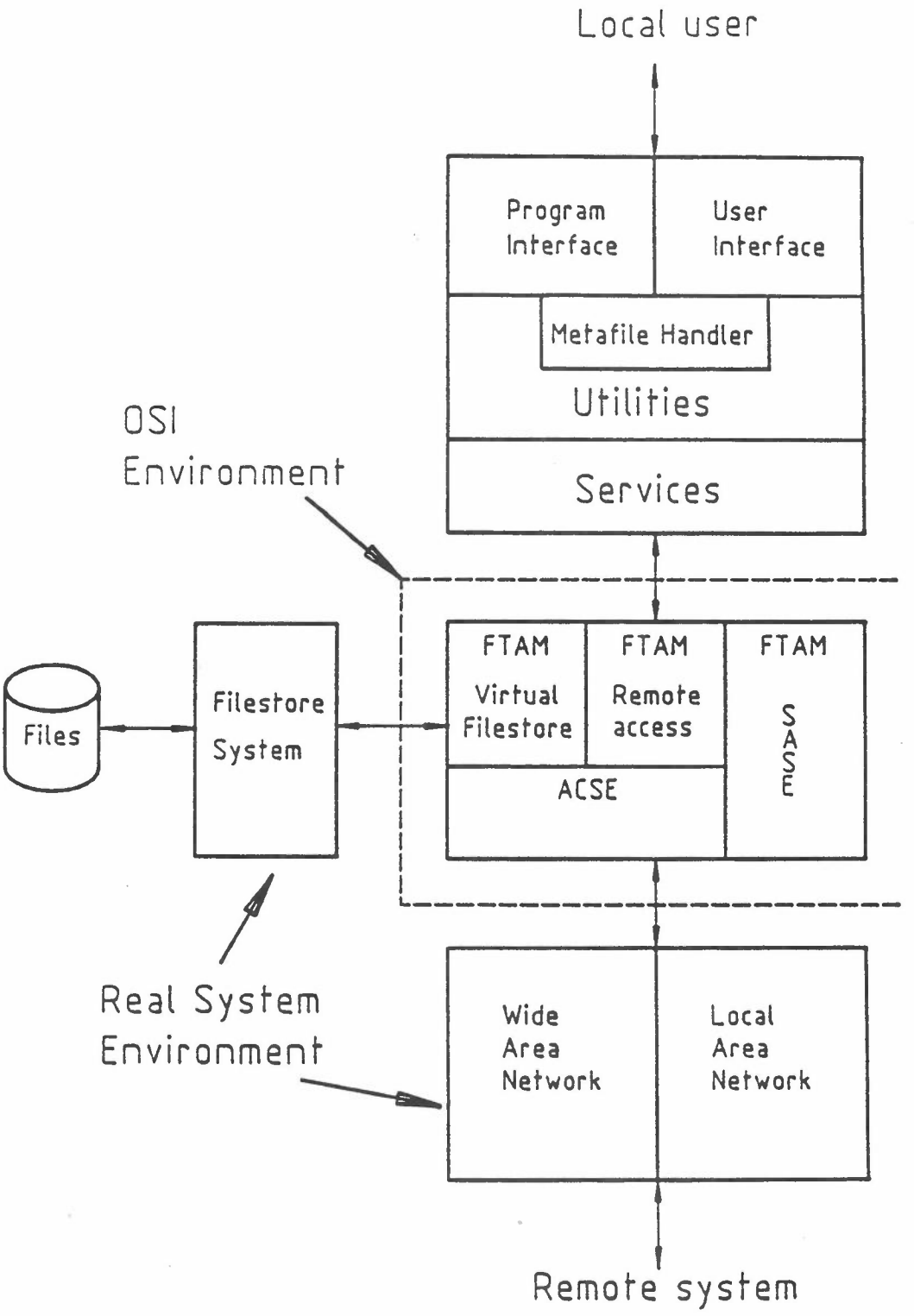


Figure 4.2-10: The CAD*I metafile utility

As illustrated in figure 4.2-10 such an utility could be divided into three levels:

- Application (above the 7 layers of the OSI model)
- Application interface (layer 5-7 in the OSI model)
- Network (layer 1-4 in the OSI model)

In the following the first two levels will be described in more details.

4.2.3.7.2 The Application Process

The application process should be structured into a set of well defined functions that are requesting services and getting feedback from the underlaying application interface. These functions may be accessed interactively and by application programs e.g. the CAD system itself. Such an application process is not contained in the OSI 7 layer model. The functions that have been considered for this first implementation study are:

- A metafile handling utility.

  The metafile is the enclosing envelope for possible many neutral files e.g. CAD*I neutral files, letter neutral files (directives from sender to the receiver), computer graphic metafiles, etc.

  The metafile handling utility is meant as a tool for merging metafiles, copy or move neutral files from one metafile to another, and list the content of a metafile. This function is mainly interactively.

- A CAD*I pre- and postprocessing utility.

  This function cover the application of pre- and postprocessor programs that translate to and from neutral files. The files might be local or remote files. File names or eventually nick name of models are kept in a global directory service.

  More than one set of pre- and postprocessors may be used within this utility. The implementation levels within the CAD*I specification /2/ defines subsets, that might cause different types of pre- and postprocessors to be implemented.

- A global CAD model directory.

  This function is an optional service that might be implemented in more permanent CAD data exchange environments. The main objective of this function is to ensure that a group of CAD systems can share the

same, physically distributed CAD models. This could also be used for being updated or keep others updated with the latest versions of specific CAD models.

4.2.3.7.3 The Application Interface

The Application interface is a support mechanism for the application process when different types of information are to be transferred via computer networks. Its main functions are:

- file management services
  The file management service assists the user in gaining access to

  *copying files,
  *creating files,
  *deleting files,
  *allocate/deallocate files, and
  *list files

  Furthermore the file management takes care of a file security service. Two aspects should be covered:

  *password protection

  This should guarantee that no user or process can access the transferred file unless the proper password is provided.

  *encipher facility

  Enciphering the file prior to and deciphering it after transfer provides additional protection.

- local directory service

  The local directory service is establishing the local user environment via an environment table . The local names can be chosen freely by the user. They will however be connected with the OSI names via the environment table. A global directory service is not possible based on FTAM, as the necessary function are contained in the FTAM phase: Filestore management phase, a phase for study in future extensions of ISO 8571.

- compress-file utility

  The transmission of large data files over wide area networks is a slow process, which could be speeded up by 30 to 50 percent by proper compression techniques.

- test-file utility

  When a new transmission partner is established, or for supervision of established lines.

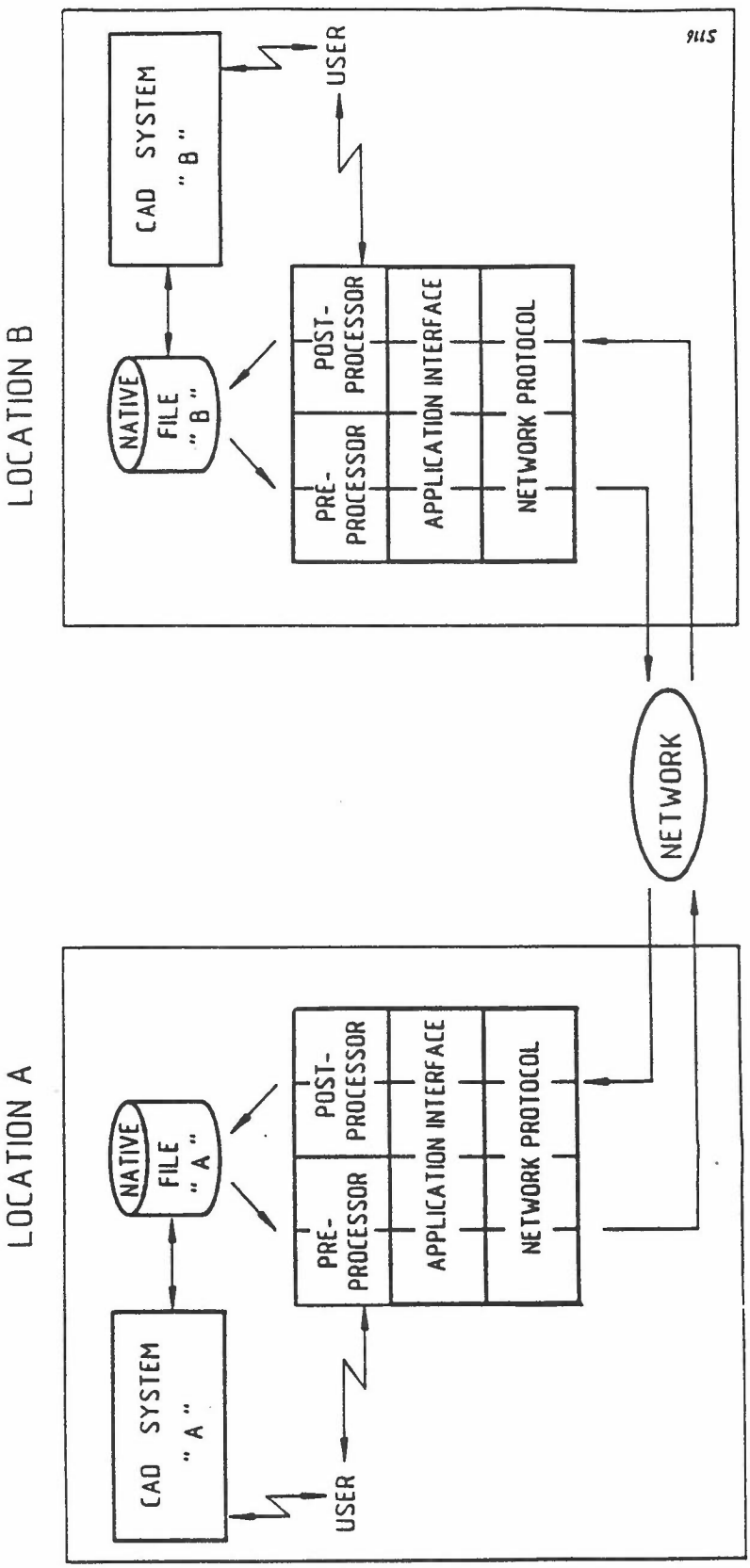


Figure 4.2-11: Concept for local implementation of FTAM based CAD interface

- restart capabilities

  Restart capabilities could be implemented here outside the FTAM protocol - unless the FTAM implementation guarantees the file transfer. The reason for doing this is that two important functional standards, MAP and TOP do not implement this function using FTAM.

A major problem when considering the use of FTAM for the CAD*I File Transfer Utility, is the selection of functions necessary to provide for the set up specification. A local model of the CAD*I future File Transfer Utility is given in figure 4.2-11. Local Model here means the local environment that would have to be established by the project partners. The local model is split into three major parts:

- the OSI Environment,
- the Real System Environment, and
- the Application Process.

4.2.3.7.4 The OSI Environment

The OSI Environment consists of:

- the FTAM Virtual Filestore, and
- a remote file access interface connected to a Wide Area Network or a Local Area Network.

The Virtual Filestore forms a neutral interface to the real files of the local system. The Virtual Filestore acts in the role of responder to FTAM requests originating from either the local user or a remote user. To ensure the integrity of the accessed data, functions that prevent e.g. simultaneous write access are necessary. Also functions that permit recovery in the transmission of large CAD data files via e.g. checkpoint insertion are part of the specification.

The remote file access interface is transmitting FTAM requests initiated by the local user to remote Virtual Filestores and transmits responses back to the local user. Both the remote and the local FTAM interfaces are part of the local FTAM implementation.

4.2.3.7.5 The Real Systems Environment

The Real Systems Environment is not part of the OSI-model, but an interface from the OSI-Environment to the Real-System-Environment is necessary and achieved by implementing a so called "mapping" between the environments. Mapping of the Virtual Filestore for

existing computers can be done at different levels of the existing real filestore system. The possibilities are:

* hardware interface (e.g. disk-controller)
* operating system interface
* programming interface (e.g. FORTRAN)
* DBMS-interface (e.g. SQL)

For future computers it must be expected that the lowest level, the hardware interfaces, will be used for computers supporting the standard. In the meantime the three remaining possibilities must also be considered.

It is clear, however, that the evaluation of the possibilities must examine the systems/languages at hand to ensure that they support the required functionality to the largest extent possible. This will ease the implementation considerably.

Implementation or mapping of the Wide Area Network is also required for practical tests and it would be a natural step to investigate if an X.25 based transport profile could be used and connected to FTAM using a strongly limited subset of OSI-protocols.

4.2.3.8 Conclusions on file transfer of CAD data

Network connections based on public and private network have been established and served as an infrastructure for the CAD*I project itself. During file transfer tests between the partners within the WG4/Network, performance and reliability have been investigated.

It was found that todays technology within computer network is sufficient with respect to reliability i.e. that test files were transferred without any errors. Most problems were discovered when the network connections had to be established. The setting of PAD, transfer protocol, and other parameters are the major problems when newcomers are starting on this.

Transferring large files as e.g. CAD files via public networks is today a very time-consuming and expensive task to perform. This is a fact because of the low transfer rates used in public networks. But it is also a fact that due to the equipment connections (usually serial interfaces with 1200 or 9600 baud) the transmission rates (of Megabit size) offered by networks can not be utilized within modern computer technology. This is not always the case within private networks, where so called DMA (Direct Memory Access) can be used.

Data compression has shown to be very effective when transferring large files, but the KERMIT method is not sufficient. Activities within ISO utilizing the Abstract Syntax Notation (ASN.1) and the related Binary Encoding Rules (BER) might be a future principle for data compression. However this is outside the scope of this working group.

Upcoming standards for computer network technology have been investigated with respect to their applicability for CAD data transfer. Today, the ISO proposal for a the file transfer standard, FTAM, looks promising for future use of CAD data exchange. The main advantage of FTAM is that it reveals the user from many tedious actions that today have to be performed without having any direct relation to the transfer of information from one system to another. Examples are dial-ups, establish the connection, type in login procedures and so on.

Based on the experience gained during transfer tests a so- called CAD*I Metafile Utility has been proposed based on the use of FTAM. This is meant as an environment for the daily practical use of handling and transferring CAD*I (and other) neutral files. Such systems will surely appear as soon as the area of CAD data transfer becomes mature.

4.2.4 Message Exchange

4.2.4.1 Application of message exchange within CAD

Originally, the term "message exchange" had been used in the project. However, it turned out that in the ISO-OSI environment this term is used both for mail application (called "interpersonal message exchange") and for manufacturing processes communication. In order to make a clear distinction, CAD*I chose to use the term intertask communication for its purposes.

Intertask communication is required, if a complex system is distributed among various processors, each one responsible for a certain task (or set of tasks).

As an example of such a complex system with distributed tasks this working group chose to look at CAD systems implemented in a distributed manner. Today CAD systems do not provide for such a facility (except in certain vendor implementations which are not revealed to and of no concern to the user).

A CAD system is built by the following principal system functions:

*user communication interface,
*the actual processor (graphics or CAD modeller), and
*the data base management system.

With todays technology those system functions, or modules, are physically located at one place e.g. within the same room or within the same building. It is possible with the upcoming network services like MMS (Manufacturing Message Specification), DTP (Distributed Transaction Processing) and others to implement CAD modules physically distributed. But seen from a users view point the feasibility of the system should not be changed.

The concept of distributing the various CAD system functions is not basically new. An important step in this direction was done at DFN (Deutsches Forschungsnetz) in Germany with a "Modelling in a network" project. A similar DFN project is concerned with "Graphics in a network".

The DFN projects are primarily concerned with interfacing the communication processor with the modeller; the CAD*I project is more concerned with the interface between the modeller and the data base management system.

Within the CAD*I project the interface to data bases is dealt with in WG4/DB. A set of routines have been defined as a proposal for a "standard interface" to data base management systems. This will allow applications to utilize the geometrical information residing in the data base. One such application is the CAD system itself but also other applications might use such an interface.

In this working group we have focused on the use of those data base interface routines on top of intertask communication services provided by modern network technology.

4.2.4.2 Architecture of a distributed system based on message exchange services

The goal of the investigations of working group 4 (networks) in the CAD*I project in the area of intertask communication is to identify the possibilities for distributing CAD system modules in a computer network environment. Other groups have studied (or are still studying) a system distribution where the user-interface module is implemented remotely from the modeller. Such

investigations are performed within the project "Verteiltes Modellieren" (distributed modelling) in the framework of the Deutsches Forschungsnetz (DFN). In the CAD*I project, emphasis is on another interface: the interface between a CAD data base management system and other modules, one of which might be the modeller of a CAD system. The architecture is (or at least: may be) similar for any of these interfaces. The general approach is described here.

First of all we have to define certain characteristics of a "module" which will be required to implement this module for remote access. The requirements are as follows:

1. A module may consists of a number of routines some of which are invoked only from the inside this module while others may be invoked from outside. In this context, those which may be invoked from outside are called "procedures".

2. A module may use of a global data structure for internal communication between its routines and for storing its state. The communication from and to the outside of the module is, however, restricted to parameters which are passed during a call to one of the procedures.

3. The parameters must be of type "value" not of type "reference" (pointer). This is no principal restriction as any parameter transfer may be implemented as a "call by value". If it was a "call by reference" originally it copying the referenced data will always allow to satisfy this restriction.

Let us now investigate the interface of a module as defined here to the rest of a CAD system.

Figure 4.2-12 shows a CAD system consisting of a main part and a module. Two services are provided by the M-module which are implemented as procedures M-SUB1 and M-SUB2. parameter-1 and parameter-2 are the two corresponding set of parameters.

Let us now assume the module M is to be implemented as a remote service on a remote computer while the main part is to remain on a local computer. This requires some elementary networking services:

1. The overall environment consisting of the local and the remote computer hardware, and the operating system plus the interconnecting computer network must provide

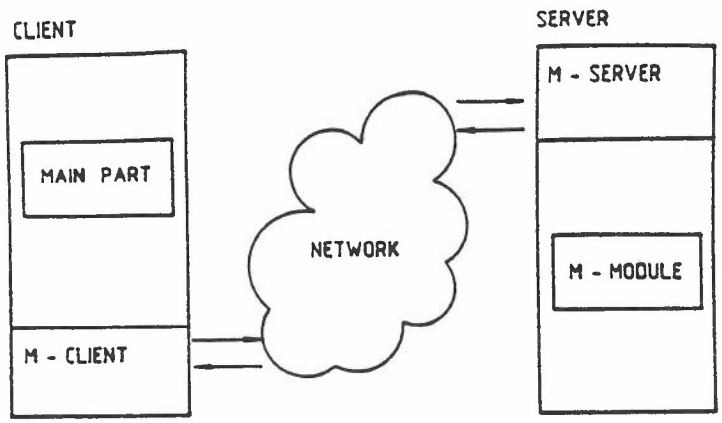


Figure 4.2-12: The original system with the M-module

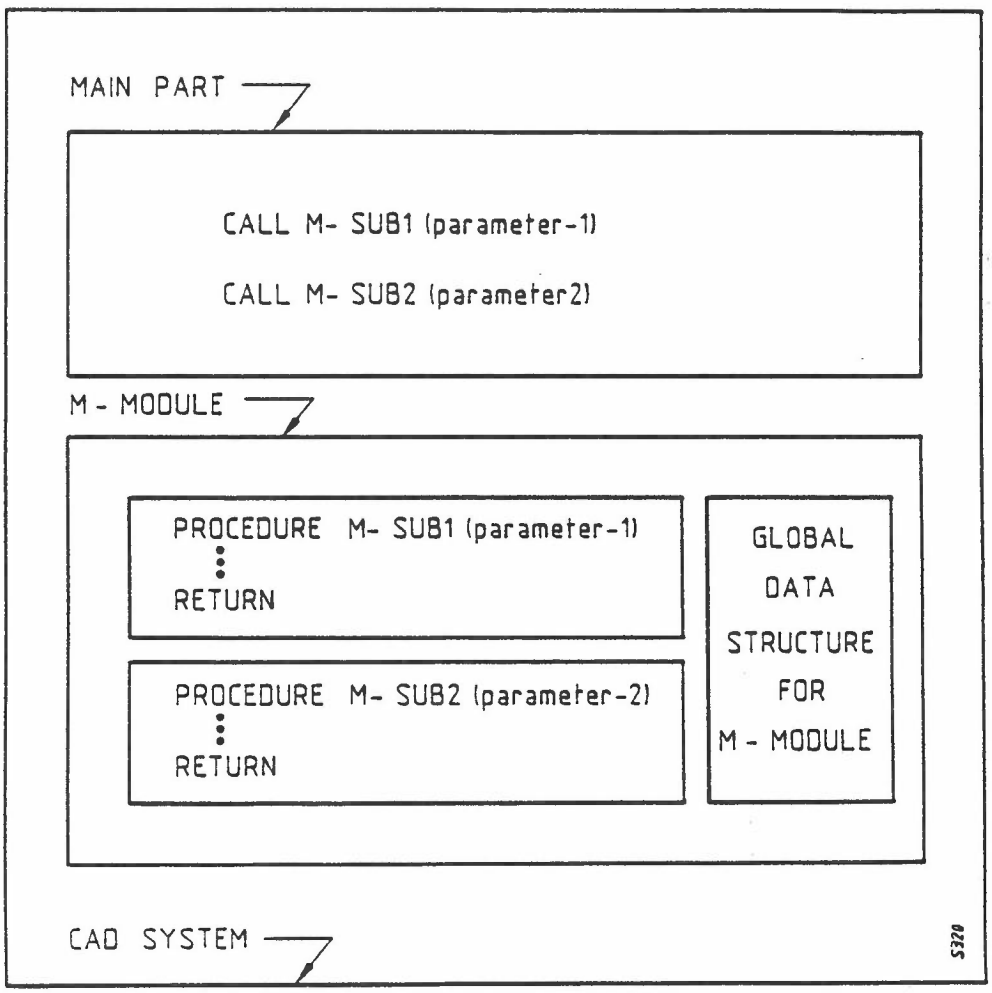


Figure 4.2.-13: General architecture of a remotely accessible module M

for the possibility to start and to terminate a program on the remote computer.

2. Two programs, one of them running on the local computer and the other one on the remote computer must be able to communicate via two basic services:

   * send: a sequence of data (called record) is transmitted to the other program. Flow control is immediately passed to the next statement after send; send does not imply waiting for response.

   * wait for reply: the program flow is halted until a record is received from the other program.

Note that we do not discuss any error recovery here. This issue will be considered later.

Figure 4.2-13 shows the general architecture of a remotely accessible module M. We introduce the following notation.

M-CLIENT:
These are routines implemented on the local system which will be bound into the CAD system instead of the original module. The specifications for the names and parameter lists of the M-client procedures are identical to the original M-procedures. Thus, the main part of the CAD system does not require any change. The M-client is shown in figure 4.2-14.

M-SERVER:
This is a main program implemented on the remote computer which will invoke the procedures of the original M-module and provide it with proper parameter lists. Thus, the original M-module does not require any change (except for the transport to the remote computer where it has to be bound into the M-server). The M-server is shown in figure 4.2-14.

Let us now take a closer look at the M-client and the M-server. The M-client contains as many procedures as the M-module made available. Each of these client procedures has the same structure:

- First, a unique code is established for each of these procedures. This code will allow the M-server to switch to the proper M-procedure.

- Then, the parameter data which were passed during the call are collected in a contiguous record R which is sent together with the code to the M-server.

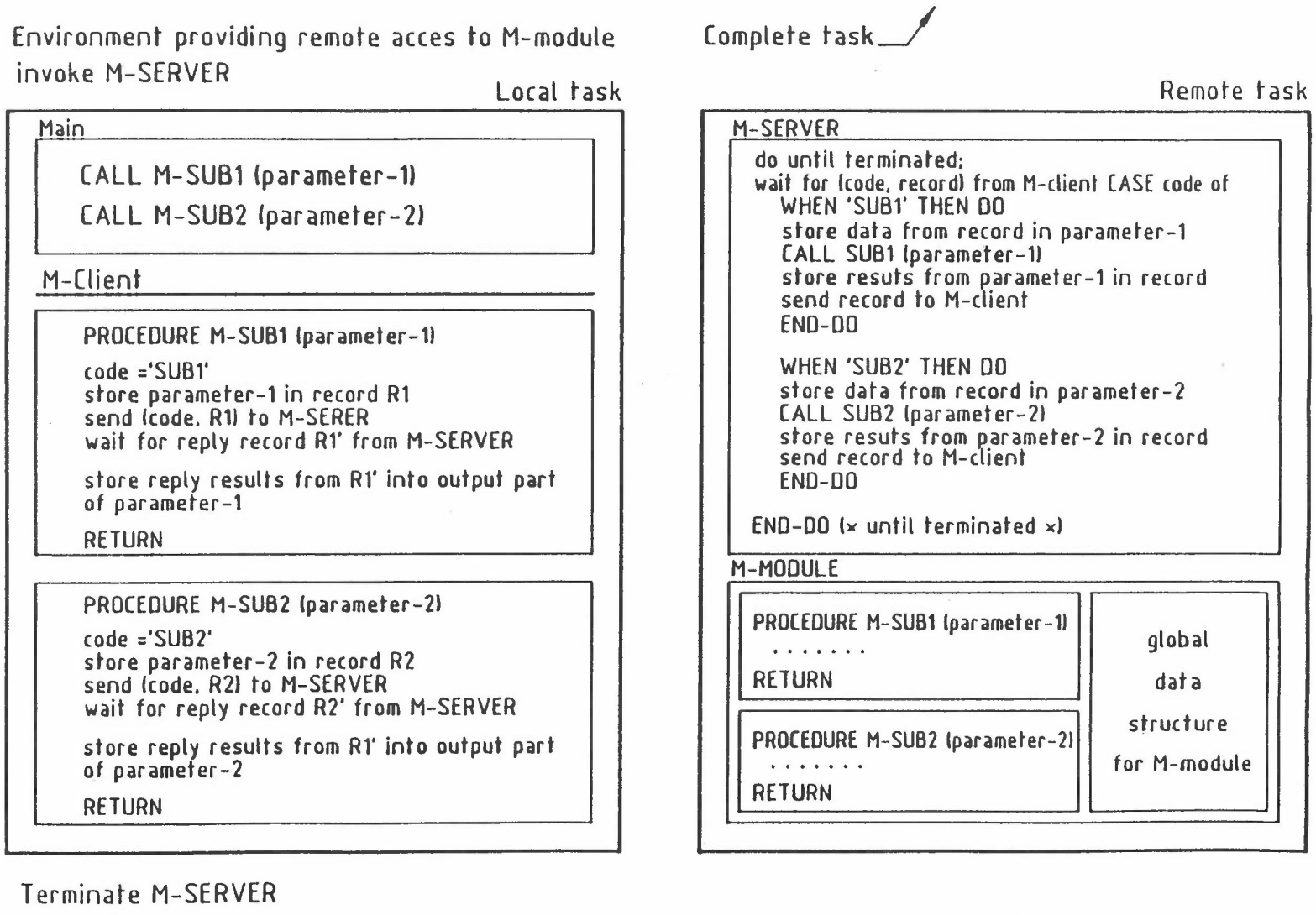


Figure 4.2-14: Functional description of M-client and M-server

- Next, the M-client procedure waits for a reply from the M- server. The reply delivers a record R' from which all the parameters, which are returned to the point of invocation, may be retrieved. The CAD system proper does not notice that the M-module was not local (except for the longer time required to perform the action).

The M-server is a main program which was started remotely and which repeatedly wait for a record from the M-client. When such a record is received the accompanying code will identify which M-procedure is to be CAlled. The data contained in the record will be passed to that procedure. The results from the procedure will be collected in a reply record which is then returned to the M-client. The M-server will continue to operate until it is terminated from the CAD system environment.

Note that providing the M-client and the M-server is not sufficient to make the whole process functional. The whole environment must be prepared for this kind of service. In particular, the possibility to start the M-server and to terminate it remotely must be established in the operating system environment.

4.2.4.3 M-client and M-server considered as software machines

It may be useful to correlate the M-client and M-server facilities with the concept of software machines as developed in /6/. Using that concept, the use of a remotely installed M-module can be considered as a set of interleaved processes. The process with the longest lifetime is the "resource management" which provides for the availability of the remote M-module in the overall environment. Within this process a sub-process "use of the M-server" may be started and terminated as required. This process is idle while the server is waiting for a record from the client. The shortest living process in this hierarchy are the remote invocations. They are started when a client procedure is called and they are terminated when control is returned to the CAD system. The lifetimes of these processes is illustrated in figure 4.2-15.

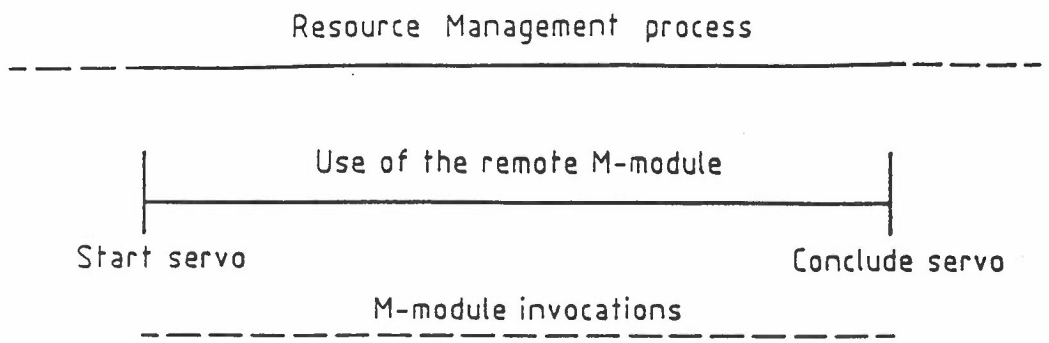


Figure 4.2-15: Lifetime of distributed processes

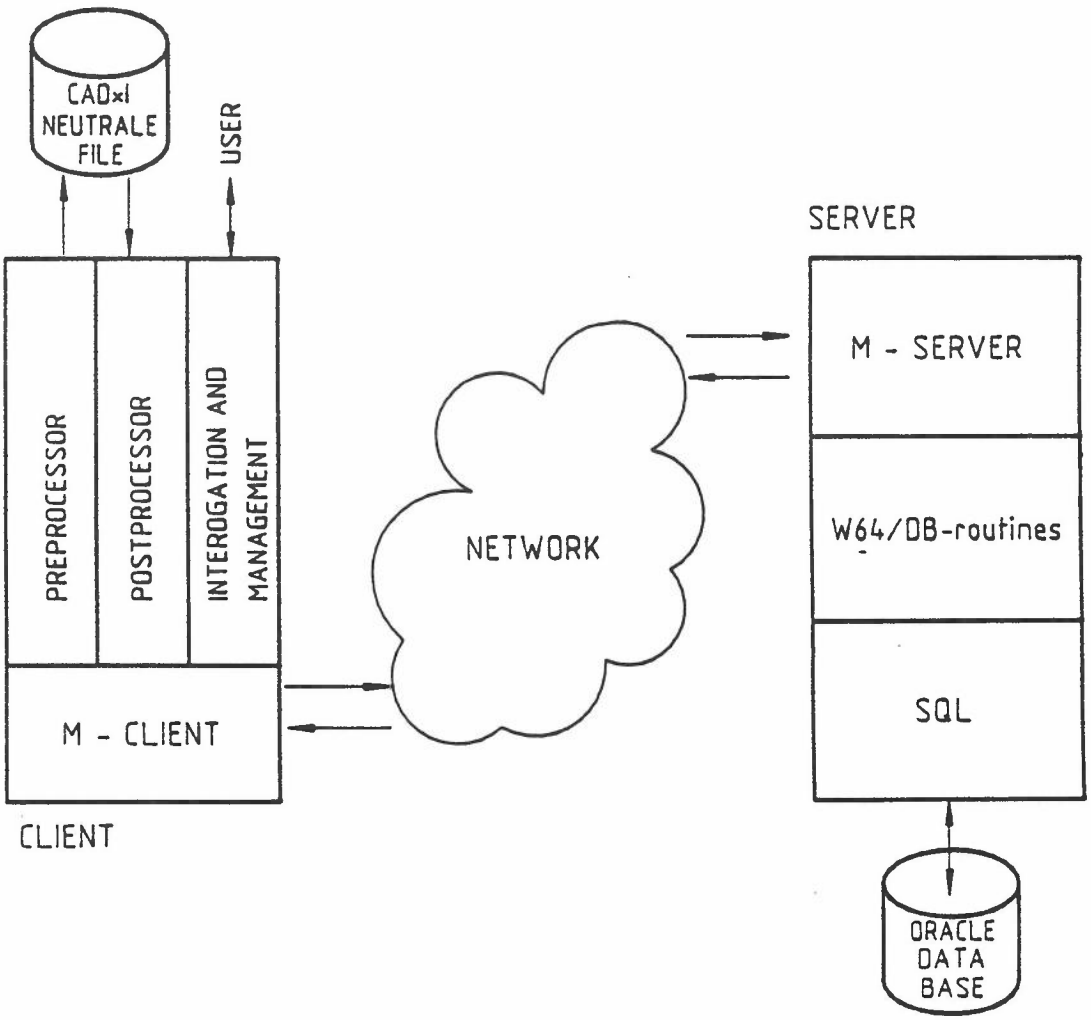


Figure 4.2-16: Implementation for feasibility study

4.2.4.4 Feasibility study on an actual implementation.

4.2.4.4.1 Introduction

As it has been mentioned before, this working group will focus on the use of the interface routines developed by WG4/DB as a basis for a feasibility study on intertask communication services in systems with distributed task processing. The goal for establishing a test facility is to implement pre- and postprocessor programs, that on the one side generate and read CAD*I neutral files respectively, and on the other side utilize the data base interface routines for the interaction with an underlying data base.

In figure 4.2-16 is shown a schematic illustration of such an implementation.

The procedures, that the server makes available, are read, write, inquiry, and delete information about geometrical models residing in the data base. Today, the interface routines support the polyhedron representation of solid models, but these can easily be extended for covering more general B-rep models.

The main issue in this context is to find candidates that can support the intertask communication services required in this implementation.

4.2.4.4.2 GOCS - the remote GKS service in DFN

A solution very similar to the one described in chapter 4.2.4.2 was developed and implemented in the framework of DEutsches Forschungsnetz (DFN) for remote invocation of GKS. GOCS (GKS Oriented Communications System) can be distributed on several hosts in an open and heterogeneous network. The system is based on the OSI reference model and uses the transport layer T.70 which is standardized in ISO 8072 and ISO 8073. The interactive possibilities of GKS, namely the asynchronous event input should not be restricted by the distribution, but the interface should be synchronous, because most protocols like T.70 do not support asynchronous events.

In addition to the basic GKS functionality (both GKS kernel and workstation capabilities) GOCS provide modules offering services for data exchange and communication in a wide area network. GOCS also provides the services that are required to start application programs based on GKS when kernel and workstations reside on different hosts. GOCS consists of entities some of which are GKS specific

and some applicable to other distributed applications. The T.70 protocol offers services for process-to-process communication (connection establishment, connection release, transport of user data). The part of GOCS which realizes the data exchange between kernel and workstations (WSI protocol) consists of the Remote Workstation Controller (RWC), the Local Workstation Controller (LWC) and the Converter, Encoder, and Decoder.

There is a fundamental difference between net service as file transfer (FT) and remote job entry (RJE) and GOCS. RJE and FT are entities which can be developed as permanent process accepting connection requests, whereas GOCS has to support any user written GKS applications which cannot be installed permanently. Thus the GOCS offers some non-GKS-specific services, the Application Support and Management (ASM), and the Distributed GKS Manager (DGM).

To start applications on a remote host the concept of the ASM was developed. This entity offers starting, controlling, and interrupting of remote applications, which run in a user region while the ASM proper is allocated to the operating system as a whole. The ASMs on kernel and workstation side are to be installed as permanent processes which are normally idle but start on request any application in the requested user region.

To ensure that the GKS application can be used without change from remote and local workstations the concept of the DGM was developed. Before the application program is started and after the termination of the application program data are exchanged between the DGM entities which are required for the execution of the application. These (pre/post) dialogues support routing via DTE address and user identification, and allocating or assigning of GKS connection identifiers to hardware devices.

The principles and techniques developed for the OSI Reference Model are utilized in GOCS. The reference model identifies the protocols and entities required and allows the protocols to be related to specific layers. The GOCS entities and protocols are located in the application and presentation layer.

The WSI application layer protocol syntax (WSI-ALPS) defines the common understanding of the graphical information which is exchanged between GKS kernel and workstations as defined in the functional description of the WSI.

The WSI presentation layer protocol data syntax (WSI-PLPDS) defines the rules how graphical information is exchanged in a common syntax and encoding. Two encoding types are supported. The type may be changed within the session.

The management protocol defines the understanding between the DGM entities.

The ASM protocol allows to start and to control applications on remote hosts.

4.2.4.4.3 Manufacturing Message Specification (MMS)

The Manufacturing Message Specification (MMS) is one of the services proposed by ISO (DIS 9506) for the communication service between manufacturing equipment that basically are elements of a distributed system. Each element is responsible for a certain task (or set of tasks) and it is possible to synchronize the tasks by exchanging messages between the individual elements. MMS is one of the communication standards on the 7th layer of the OSI Reference Model.

The main purpose of MMS is to control and monitor manufacturing equipment of different types and of different vendors. It is necessary to describe the functionality of each device in a way that is independent of specific device functions, as such equipment (e.g. robots, CNCs and PLCs) has to operate using the same communication services (e.g. start, stop, download and upload). For that reason MMS specifies a so called Virtual Manufacturing Device (VMD) that is an abstract representation of a specific set of resources and functionality at the real manufacturing device, and a mapping of this abstract representation to the physical and functional aspects of the real manufacturing devices.

Before starting a communication between devices a framework has to be established that defines the requirements and capabilities of communicating MMS devices. The VMD is used to identify for an external device if the necessary requirements are available at the server's side. The device that issues the communication service is called the client. The responder of a communication request is called the server.

The client tries to establish the MMS environment by sending an initiate request to the server, that might response either positive or negative to the request. Getting the initiate indication the server checks for the requested requirements can be available for the client. A

negative response on the initiate request means that the request has been rejected by the server. A positive response tells the client that the request has been confirmed and the MMS environment is established. The general structure of such confirmed services is shown in figure 4.2-17.

The sequence of operations for exchanging messages between two devices are shown in figure 4.2-18.

The MMS will be studied further for identifying the commands that can be used within this working group.

4.2.4.4.4 Remote Database Access (RDA)

An ISO standard for remote data base access (RDA) has been proposed and today, it exists as a draft proposal (DP 9579) /7/. The standard proposal is meant to facilitate access to remote data bases from intelligent workstations and other data base systems.

The RDA standard proposes, among other things, services for the use of RDA in a relational data base environment which conforms to the ISO Data Language SQL.

The ISO material about this draft proposal is currently not available to this working group.

4.2.5 Conclusions

The implementation of a distributed data base system utilizing the WG4/DB routines is under consideration within this working group. In the actual implementation two microcomputers (PCs) have been considered as being hosts for the calling task (the CAD*I processing environment) and the called task (the remote data base system). The reason for using PC's as hosts is that MMS products today are available for these machines (e.g. CONCORD). Furthermore, the price of such equipment is today within the range of 7.000 -15.000 ECU including network connections and MMS software (full 7 layer implementation). Other similar products like RDA are not likely to appear within the time period of the CAD*I project.

Before acquiring equipment for establishing a MMS environment the necessary commands of MMS for transferring the code for subroutines and the additional parameter lists have to be investigated in details.

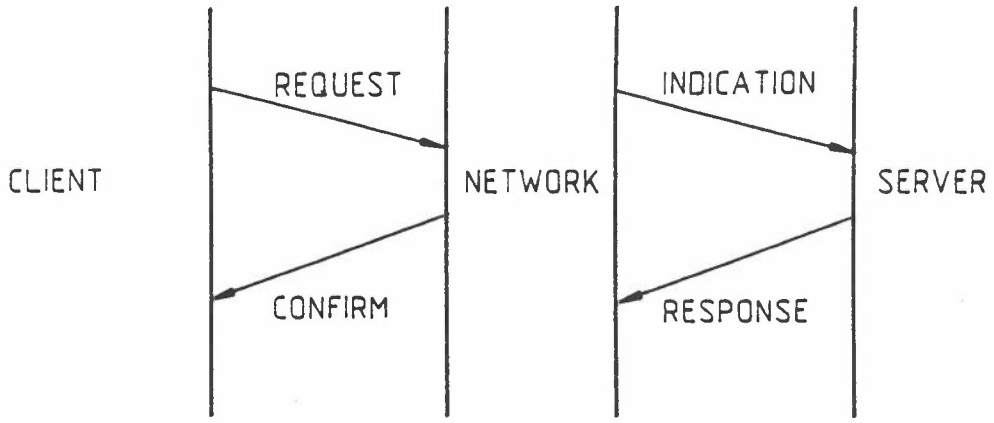


Figure 4.2-17: General structure for confirmed services

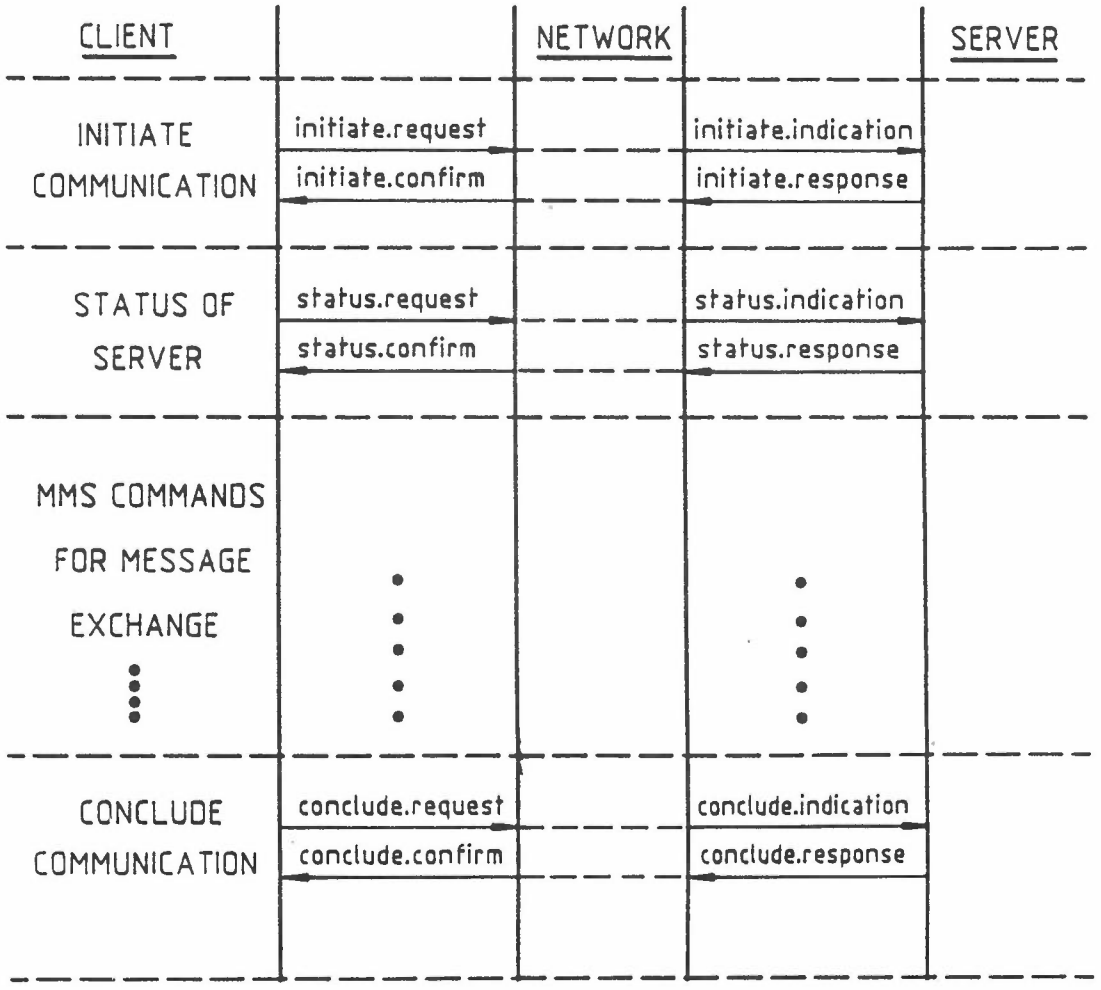


Figure 4.2-18: Operation sequence for message exchange

Also the use of ORACLE as the data base system may cause some problems as a FORTRAN precompiler translating SQL statements into FORTRAN statements is not yet available. This is only the case when using PCs. The ORACLE library that can be linked to the program body will be compatible to a Microfocus version of Fortran that at the moment is not known to this working group.

A shift to Pascal or C will be a possibility but this will require a full rewriting of pre - and postprocessor software, and of the WG4/DB routines.

To demonstrate the principles in having a remote data base system, a simple test facility is being established at DTH. This test facility will use RS232 as the physical connection and dedicated client and server routines for establishing the transfer mechanisms as mentioned in the former chapters.

Furthermore, a test implementation of the processing environment utilizing the WG4/DB routines is planned to be implemented at DTH. This will be located on one host and will be used for comparing this solution with a distributed solution.

4.2.6 References

/1/ Palstrøm, B. (Ed.):
"CAD Data Exchange via Networks - File Transfer", July 1988, (Final report concerning file transfer experiments within the CAD*I project)

/2/ Schlechtendahl, E.G. (Ed.):
"Specification of a CAD*I Neutral File for CAD Geometry, Version 3.3", ESPRIT Project 322, Springer Verlag, 1988.

/3/ Trostmann, E. et al.:
"CAD data exchange via Neutral Interfaces", Enterprise Network Event (ENE'88), June 5-9,1988, Baltimore, USA.

/4/ Kroszynski, U. et al.:
"Geometric Data Transfer between CAD Systems: Solid Models", (submitted for publication)

/5/ Klaus L. C. Truoel:
"International Standardized Profiles - FTAM as an Example of Harmonized OSI Standards", Enterprise Network Event'88, Baltimore, June 1988

/6/ Encarnacao, J. and Schlechtendahl, E.G.:
"Computer Aided Design - Fundamentals and System Architectures", Springer-Verlag, 1983

/7/ Belisle, P. and Jansson, H.:
"OSI - What Is Next?", Enterprise Network Event (ENE'88), June 5-9,1988, Baltimore, USA

5. Working Group 5: Advanced Modelling

5.1 Introduction

This part of status report No. 5 covers the results reached by the advanced modelling group within the period from November 1st, 1987 to October 31st, 1988. A comparison of the achieved results to the described objectives of the technical annex will be given.

Extract of the technical annex:

Summary:

While present CAD systems are characterized by a rather primitive operational modelling interface (set operators, elementary geometric operators), future CAD systems are envisaged to require a high-level communication interface with pattern recognition and semantical scene analysis capabilities. Such interface techniques are to be developed in the project, and their impact upon the other CAD interfaces is to be determined.

Objectives:

Global objectives are increasing of effectiveness and user orientation of communication interfaces of CAD/CAE systems (between system and system user) by improving the communication techniques. Actual CAD/CAE command languages are very formal and not really interactive and should be replaced by graphical/interactive and nonformal communication techniques based on A.I. functions like pattern recognition and semantical scene analysis.

Detailed objectives are:

- Analysing needs and requirements of product and process design concerning the user-system communication,
- specification of a technical term dictionary and related technical semantics,
- analysing recent A.I. methods to be applied in CAD/CAE communication techniques (e.g. pattern recognition, semantical scene analysis),
- specification of new communication techniques using above results to gain interfaces which are able to serve also not well trained users and such which are not familiar with I.T. but carry high-level technical know-how,

- development of these (or a subset) techniques,
- implementation and testing of a prototype interface,
- verification of the prototype in a pilot installation.

The technical advantages against recent dialogue techniques applied in CAD/CAE systems (especially in solid modellers) are enormous.

5.2 Report on performed work

5.2.1 Handsketching Input System (HIS)

In Status Report No. 3 /1/ a detailed description of underlying concept of the Handsketching Input System was given. The principles are as follows:

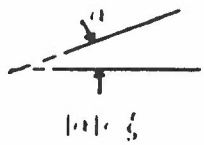
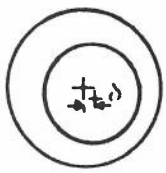
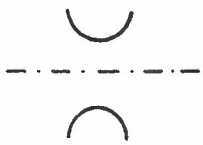
1. The user input (sketched geometry) has to be recognized in its syntactical meaning.

2. The semantic meaning has to be interpreted with respect to previous input or to the state of the model.

3. The recognized entities and their semantics have to be stored in a model based on the principles of geometric constraints.

To be able to meet these requirements a conceptual structure of the HIS based on a two layer architecture was proposed. The main task was the definition of a rule base to describe the semantic of geometric associations. Figure 5.2.1-1 shows a subset of the rule base.

Based on the proposed system architecture a functional model of the software system was given in Status Report No. 4 /2/. The software implementation could be finished during the last half year. A robust prototype, on the level of a university, of the HIS is available. During the Industrial Fair at Hannover (20th - 27th April 1988) and the "CAD/CAM-Öffentlichkeitstag" (14th October 1988) a demonstration to a wide public could be given. The available prototype is written in standard PASCAL with VMS extensions and running under VAX-VMS and GKS on a black and white or colour VAX station. Further work outside of the CAD*I contract will be the integration of the X-Window software into the existing user interface.

GA	rule	example
//	par(i,j) := [i: typ(LINE), j: typ(LINE), |win(i,j)|<ε, => ins_par(i,j)], [...].	
	win(i,j) := [i: typ(LINE), j: typ(ARC), tangente(j,P,k), |win(i,k)|<ε, => ins_tan(i,j)], [...].	
	koax(i,j) := [i: typ(CIRCLE), j: typ(CIRCLE), |abs(M(i),M(j))|<ε, => ins_koax(i,j)], [...].	
	sym(i,j,k):= [i: typ(ARC), j: typ(ARC), k: typ(SYM_LINE), |abs(i,k)-abs(j,k)|<ε, => ins_sym(i,j,k)], [...].	

legend:

i - existing entity
j - new entity
k - tangent or axis of symmetry
P,M - points

Figure 5.2.1-1: Rule base of the context analysis system

5.2.2 Design by Technical Terms Implementation (DTT)

As a basis for the development of the DTT-Modeller, an earlier version of the DICAD-System (DICAD = Dialogue Oriented Computer Aided Design) is used. DICAD was developed at the RPK-department of Karlsruhe University, West Germany.

The key numbers describing the version used within the ESPRIT project are illustrated in fig. 5.2.2-1.

Basic Issues

The functionality of the basic modeller allows the handling of single vertices, edges, sets of edges, faces, sets of faces, as well as volumetric objects of the b-rep type. The surface types of the system are planar, conical, and cylindrical faces. The line types supported by the system are straight line, circular line, ellipsis, and a general section curve. Typical objects which can be modelled based on this model schema are represented in fig. 5.2.2-2.

There are several possibilities for geometric modelling. One possibility is the insertion of predefined geometric elements such as straight line, rectangle as a polygon, rectangular face, or cube. A second possibility is the dimensioning of elements as well as the positioning and the orientation of them. Additionally, the sweep-operation serves to generate volumetric elements based on face-type input. A limited set of boolean operators and deletion operations allows for the manipulation the modelled objects (see fig. 5.2.2-3).

Associativities

Based on the given types of geometric elements, the possible relationship between them have been explored. It is distinguished between "trivial" and "interesting" associations. Trivial associations are those which are implicitly included into a b-rep schema such as adjacent faces. Interesting associations are defined as those which can deliver additional information about geometric elements.

There was introduced the distinction between an action and a reference element. The classification schema of associations is a matrix of possible relations between the two sets of elements

Action Set = (Vertex, Edge, Face, Volume)

Design by Technical Terms (DTT)

based on DICAD system
(Dialogue orientated Integrated CAD - System,
developed by Institute RPK,University of Karlsruhe)

Programming Language:	Pascal
Number of Modules:	1047
Number of Statements:	46.798
Number of Menues:	~80

(Status: September 1988)

Figure 5.2.2-1: Key numbers of Dtt-Modeller

Volumes

Faces

- planes
- cylinders
- conic surfaces

Edges

- straight lines
- arcs
- ellipses
- curves

Vertices

Examples:

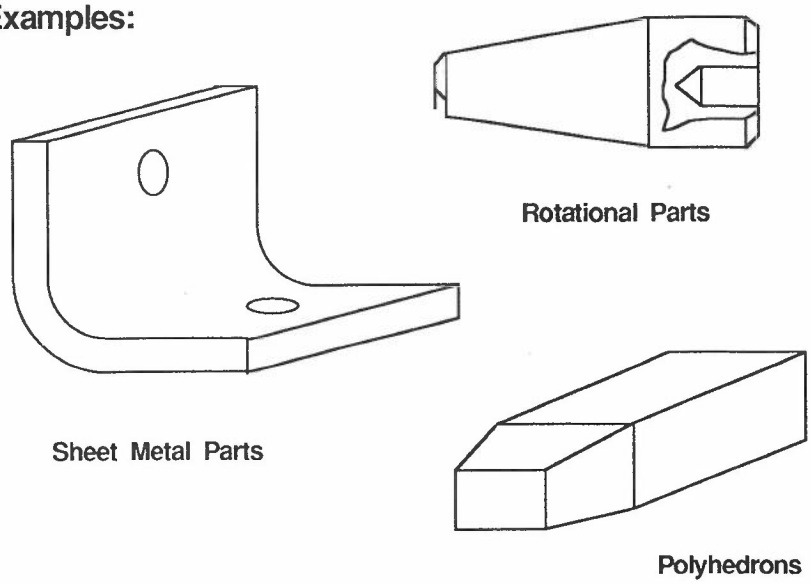


Figure 5.2.2-2: Typical objects of the DTT-Modeller

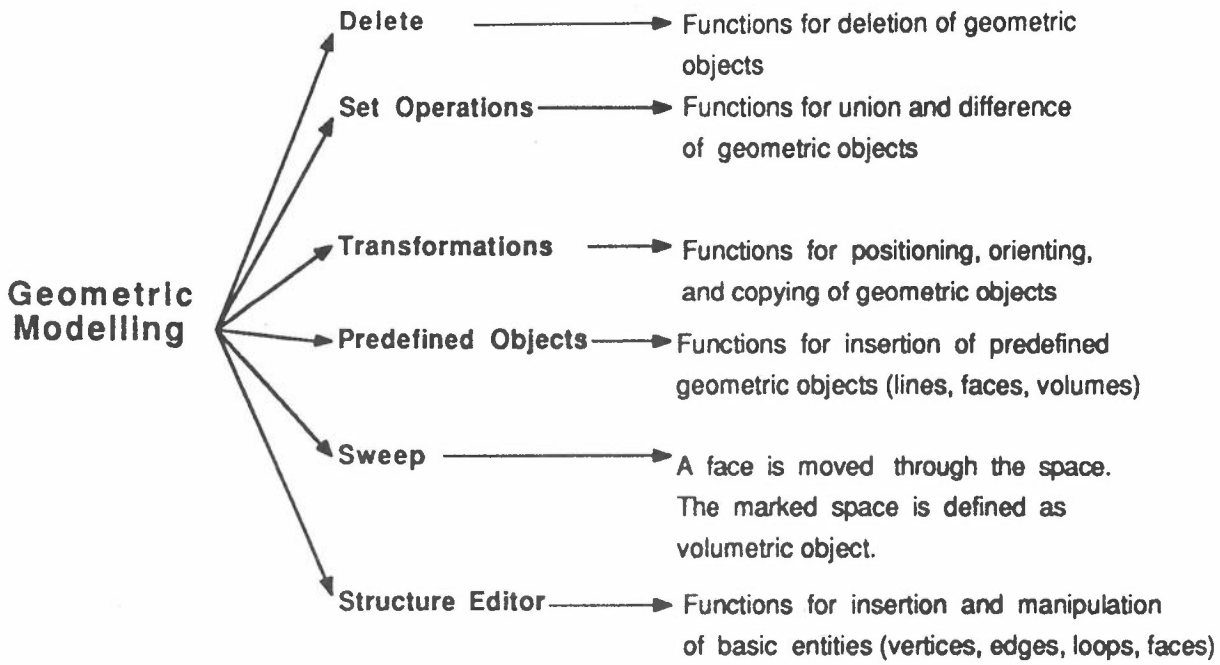


Figure 5.2.2-3: Operations for geometric modelling within the DTT-Modeller

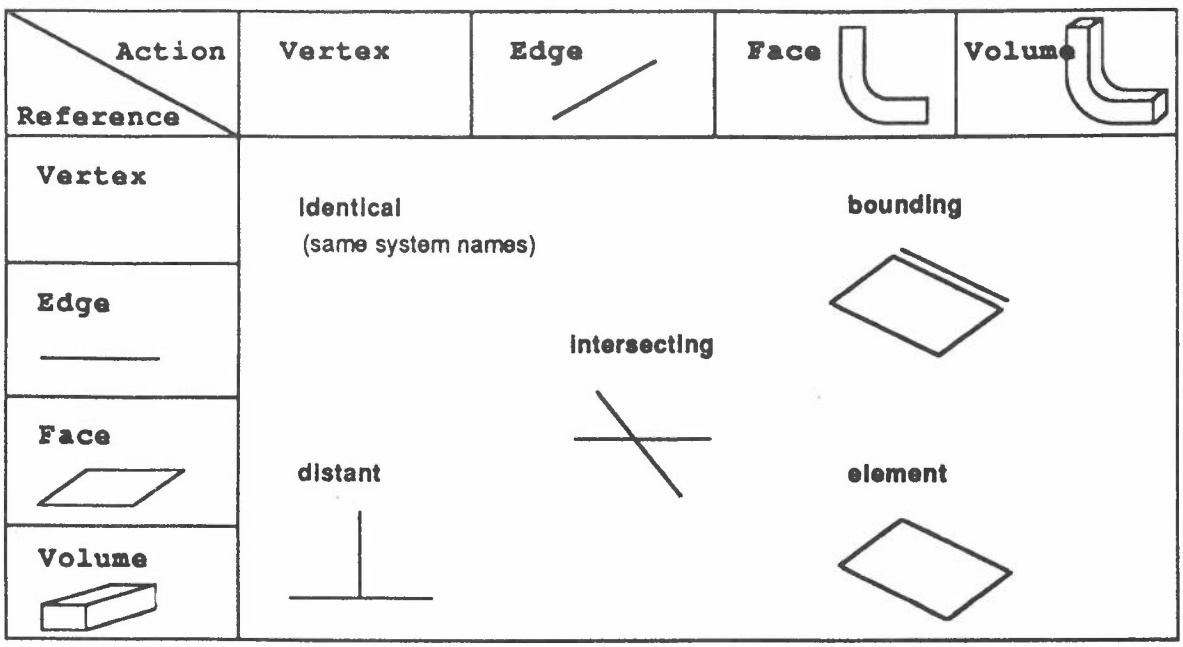


Figure 5.2.2-4: "Trivial" associations

Reference Set = (Vertex, Edge, Face, Volume)

Some trivial associations can be found as summarized in Fig. 5.2.2-4. The characteristic of trivial associations is, that they occur between a lot of combinations of geometric elements and that they contain only low-level information.

For example there can be a vertex identical to another vertex, an edge identical to another edge, etc. A vertex can be distant to another vertex, to an edge, to a face, or even to a volume, as well as edges, faces and volumes can do so. Geometric elements can intersect each other as well as bound, or be element.

Higher-level information however can be generated out of regarding further possibilities to relate action elements to reference ones. Therefore the general possibilities to use the terms published by former papers are now inserted in such a classification schema (see fig. 5.2.2-5).

It is difficult to draw a line between geometric and technical associations. Therefore, the following definition is made:

- Geometric associations are constraints between geometric elements which allow the determination of some attributes of the geometric elements. No analysis of the technical context is needed to obtain an unambiguous result.

- Technical associations are those whose meaning is determined by looking at the technical context.

Some of the "interesting" associations can lead to modifications of the geometric elements only if the technical context is clear. Therefore the meaning of the technical context shall be clarified.

Technical Modelling

Technical modelling is defined as modelling technical elements, i.e. products, assemblies, parts, form features as well as their attributes (position, orientation, dimensions, tolerances, materials, surface qualities, etc.). Technical-associative modelling is defined as modelling the relations between the technical elements. This information often allows to determine the attributes of technical elements. Both, technical modelling and technical-associative modelling are required to build a set of modelling operations for supporting the preliminary design. To develop these modelling

operations, the product model of the DICAD system has been augmented by the technical shape model. As a result, it can be distinguished by modelling

- product
- assemblies
- parts
- features

as well as all their interrelations. A further concept with equal importance also has been introduced to distinguish between

- construction points
- construction lines
- construction faces
- construction spaces.

An overview about the technical modelling is given by fig. 5.2.2-6.

Sectional Surfaces

To solve construction problems often certain views or sectional views of the interesting part are made. The result of the handsketching input system (HIS) for example is of 2d-type. To use a CAD-system in a convenient way, e.g. examine manufacturing data, a three dimensional structure of the part must be present. A sweep operator transforms the two dimensional drawing data to a complete three dimensional structure. To do this, the interesting face and trajectory are given. The area touched by moving the face along the trajectory defines the resulting solid. A lot of possible trajectories exist, but in this work only a translational, defined by a translation vector, and a rotational trajectory, defined by a rotation axis and an angle, are considered. These limits are already suggested by the modeller, because it knows only planar, cylindrical and conical surfaces. The definition shows, that a lot of parameter restrictions have to be noticed, e.g. a face must not be moved along a vector, which is rectangular to its normal vector. The result would not be a solid.

To understand the aspects of a technical sweep operator the technical details of the underlying modeller must be explained. The available modeller knows elements, which store information going beyond pure geometrical information. Associations of several elements and organisational objects - such as form features, parts or assemblies - belong to this set. The technical sweep has the job to transfer the technical information from the input face to the volume structure and to store

Action / Reference	Vertex	Edge	Face	Volume
Vertex	(not equal) equal	aligned	aligned	
Edge	aligned	parallel angular aligned concentric	parallel angular coaxial	
Face	aligned	parallel angular coaxial	parallel angular aligned flush coaxial imposed	imposed
Volume			imposed	imposed fitted in

Figure 5.2.2-5: "Interesting" associations

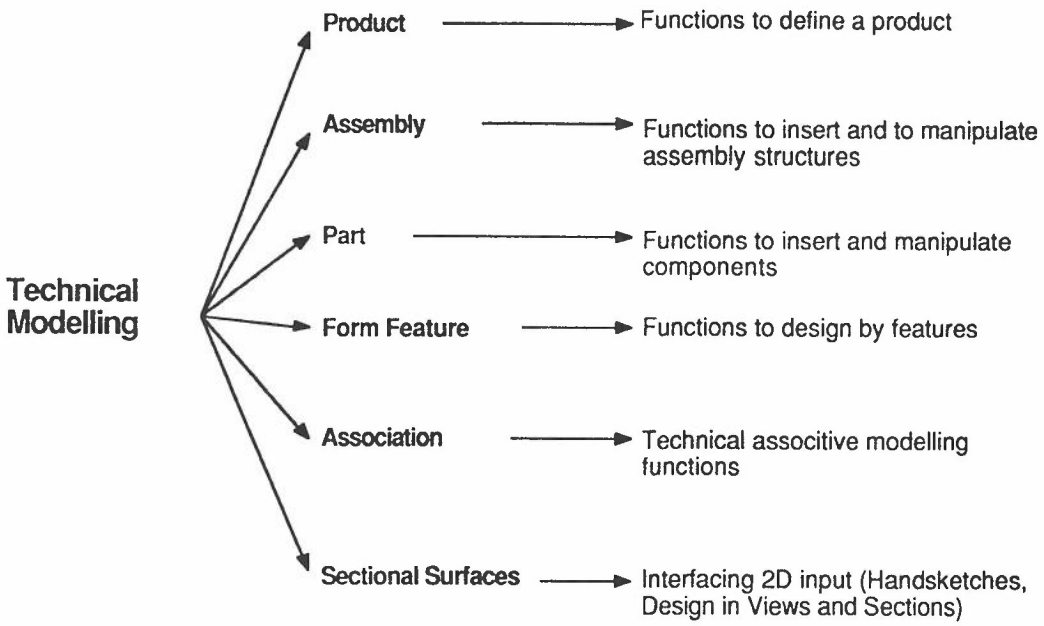


Figure 5.2.2-6: Operations for technical modelling within the DTT-Modeller

explicitly the technical semantic of the sweep operation itself.

The sweep transforms two-dimensional geometric data into real solid structures. Consequently it takes place after programs, which build up these two-dimensional structures. Of course, these programs should also insert the technical information. The result structure of the sweep is a complete boundary representation of a three-dimensional object including the technical information. To utilize properly this structure efficient 3D-operators are necessary. Fig. 5.2.2-7 shows the principal operation which a technical sweep performs.

The role of the technical sweep operator within the overall system structure is to build a link between 2D-input and a 3D-representation within the DTT-Modeller. Therefore it represents the link between the handsketching input system (HIS) and the DTT-Modeller. However, also within the DTT-Modeller 2D-structures can be modelled interactively or by an application program. 2D-structures can contain associativities or feature information. Both are transformed into 3D-associativities or -features. The role of the technical sweep operation within that area is illustrated in fig. 5.2.2-8.

Technical Associations

The methodology of using technical associations was described in earlier reports. Therefore two things are described here: First, a formal procedure to define a new association, second, the application of the associativities concept to dimensioning.

Procedure to define a new technical association:

a) Select a new term as a technical operator
b) Verbal definition of the semantic of the new term
c) Definition of interface parameters
d) Definition of transformation algorithm
e) Test, if all required modules exist
f) Definition of interfaces of new modules
g) Implementation of new modules
h) Implementation of new technical operator
i) Integration into modeller
j) Test of new association

For the purpose of dimensioning, parametric modelling operations were developed. These modelling operations and - in addition to that - a set of dimensions are now implemented.

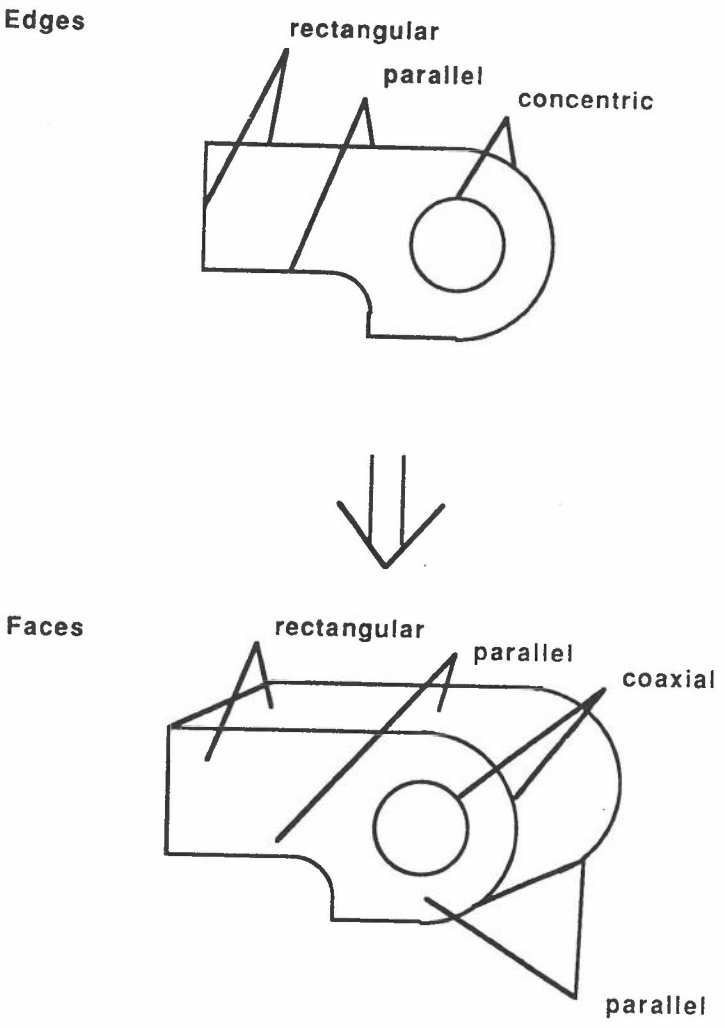


Figure 5.2.2-7: Technical sweep operator

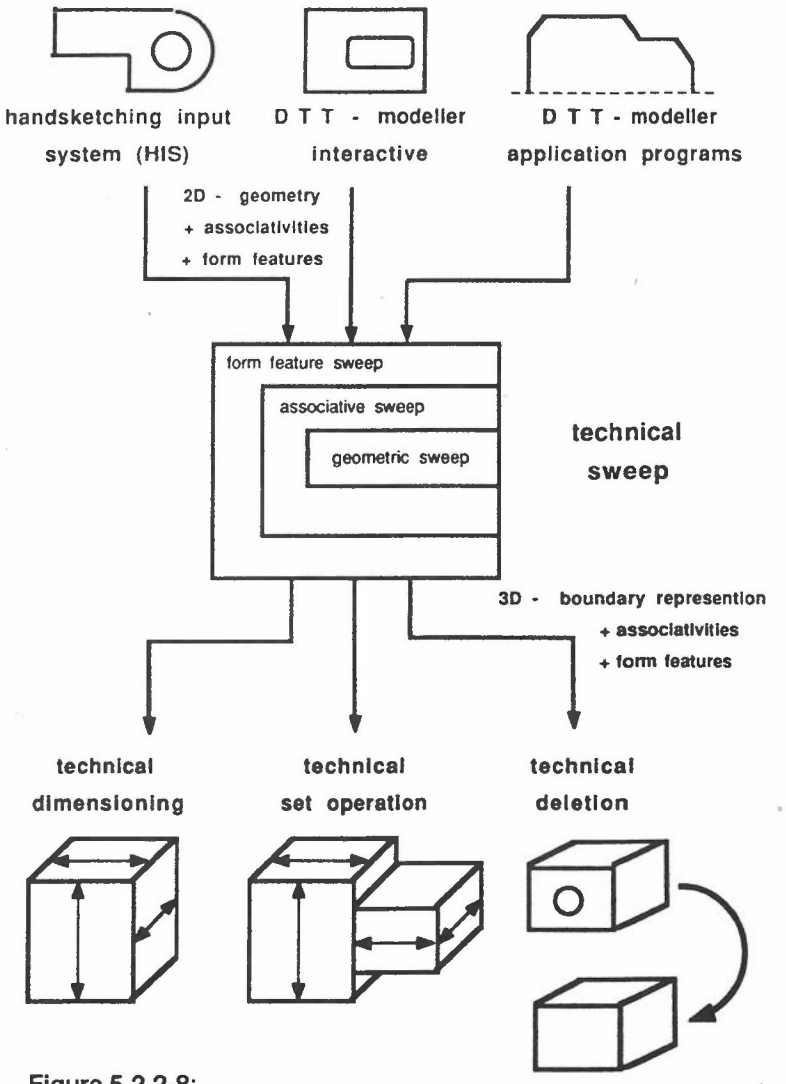


Figure 5.2.2-8:
The role of the technical sweep operation within the DTH-Modeller

Corresponding to conventional design where most of the dimensions respectively their metric value imply a geometric association (e.g. distance parallel-100), the dimension was set as a combination of geometrical attribute (distance) and geometrical association (parallel).

The integration of technical attributes (tolerances) is planned. The types of dimensions see fig. 5.2.2-9 were defined according to the requirements of dimensioning and the geometric operands of the model.

Operations considered are as follows:

insert
modify
delete
change (of action and reference element).

Form Features and Parts

It is possible to use associative methods in addition to basic modelling methods in single part and form feature modelling that is e.g. to describe a form feature associatively and to modify it.

The following example describes this approach.

Initial status of design: A single part is available and has been described by dimensions.

1) Generating of geometry of a new form feature (geometric operational)
2) Geometry described by dimensions (geometric-associative)
3) Defining of geometry including dimensions as form feature "slot" (technical-operational)

Step 1 - 3 see fig. 5.2.2-1o.

4) Storing of form feature "slot" for further use
5) Changing into wanted dimensions (geometric-associative)
6) Positioning and orientating by technical associations and operators of assembly-modelling (flush, centered) (technical-associative)

Step 4 - 6 see fig. 5.2.2-11.

7) Using boolean operation (see fig. 5.2.2-12).

At the same time the form feature "slot" (with geometry described by dimensions) is available in a file for future technical modelling. Therefore the geometrically operational insertion of geometry can be omitted, a

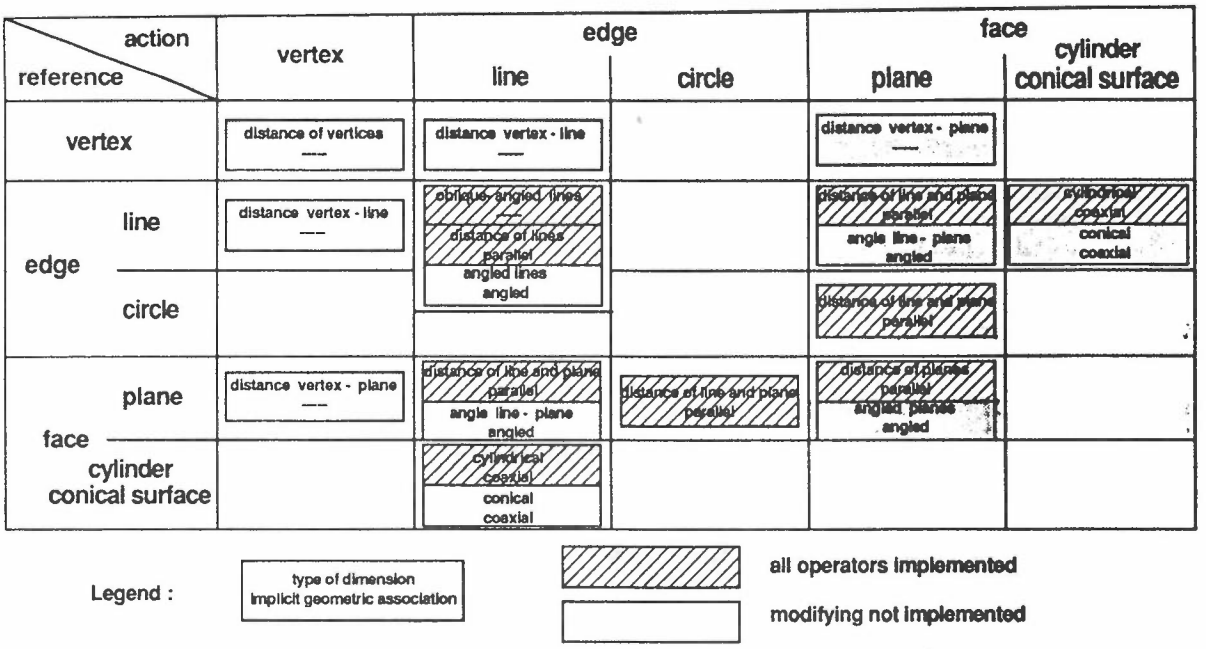


Figure 5.2.2-9: Associativities for dimensioning

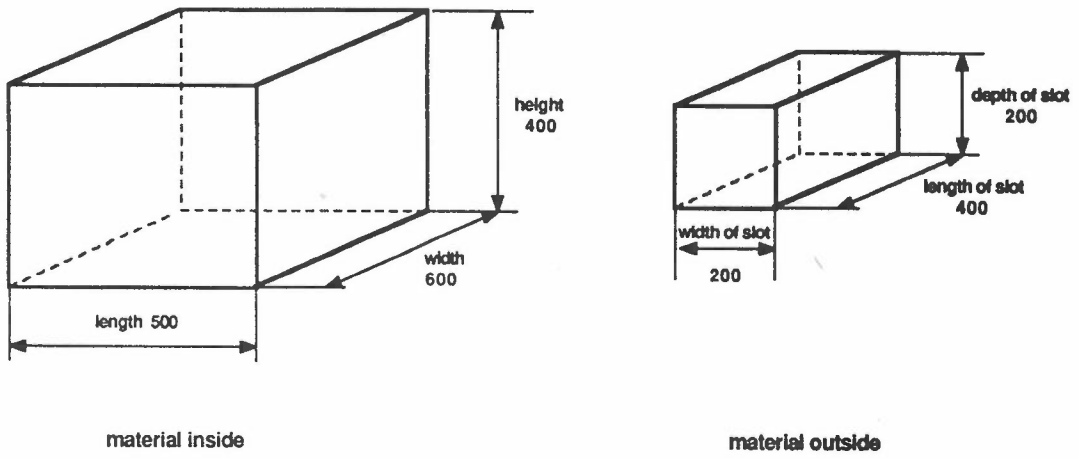


Figure 5.2.2-10: A cube and a slot

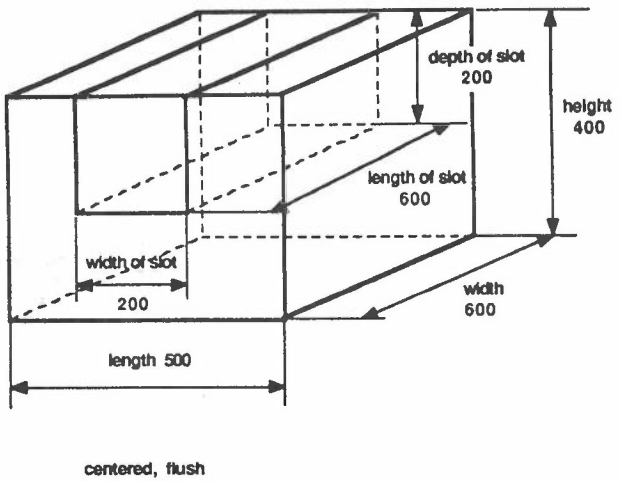


Figure 5.2.2-11: Slot dimensioned, positioned, and orientated with respect to the cube

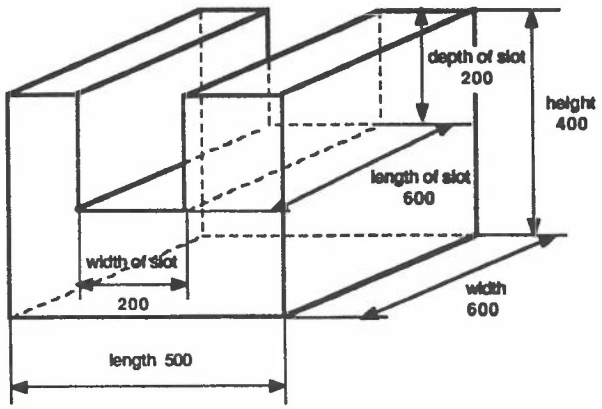


Figure 5.2.2-12: Result of boolean operation

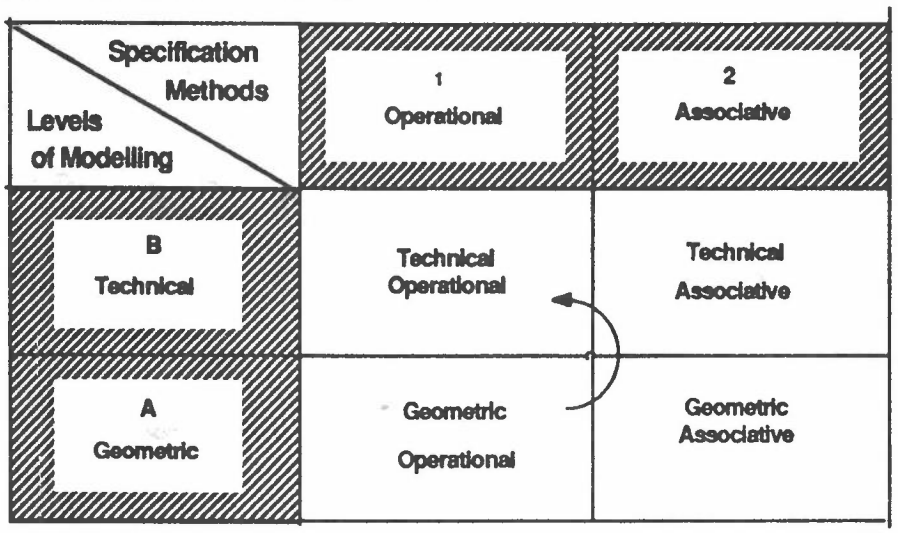


Figure 5.2.2-13: Covered subset of general modelling level concept

restriction on technical and associative methods can be made.

In terms of our general modelling concept a design cycle was performed, which covers the subset of technical and technical-associative modelling (see fig. 5.2.2-13).

5.2.3 Technical Modelling (TMI) and Geometric Associativity Interface (GAI)

The status of work related to the development of interface specifications is described within this chapter.

Concerning the actual realization status related to our general concept of internal interfaces (see /2/) a picture can be drawn as follows (fig. 5.2.3-1):

Initial experiences have been gained by translating descriptions of technical terms as formalized in /3/ into Pascal procedures for technical-associative methods (TA_*). Required basic methods, which are first extensions of AIS are the following geometric-associative methods:

- ga_mod_faceparallel
  makes an action face parallel to a reference face
- ga_mod_edgeparallel
  makes an action edge parallel to a reference edge
- ga_mod_coaxial
  makes an action face coaxial to a reference face
- ga_mod_angular
  modifies the value of an angle
- ga_mod_matr
  makes the material direction of two faces, two edges, or an edge and a face equal or unequal.

Each of the listed associative methods first has to make a context analysis. Depending on the membership of the action element either to a volume, to a face, or to a single edge, and depending on the membership to the same or to different parts the corresponding actions are to be taken.

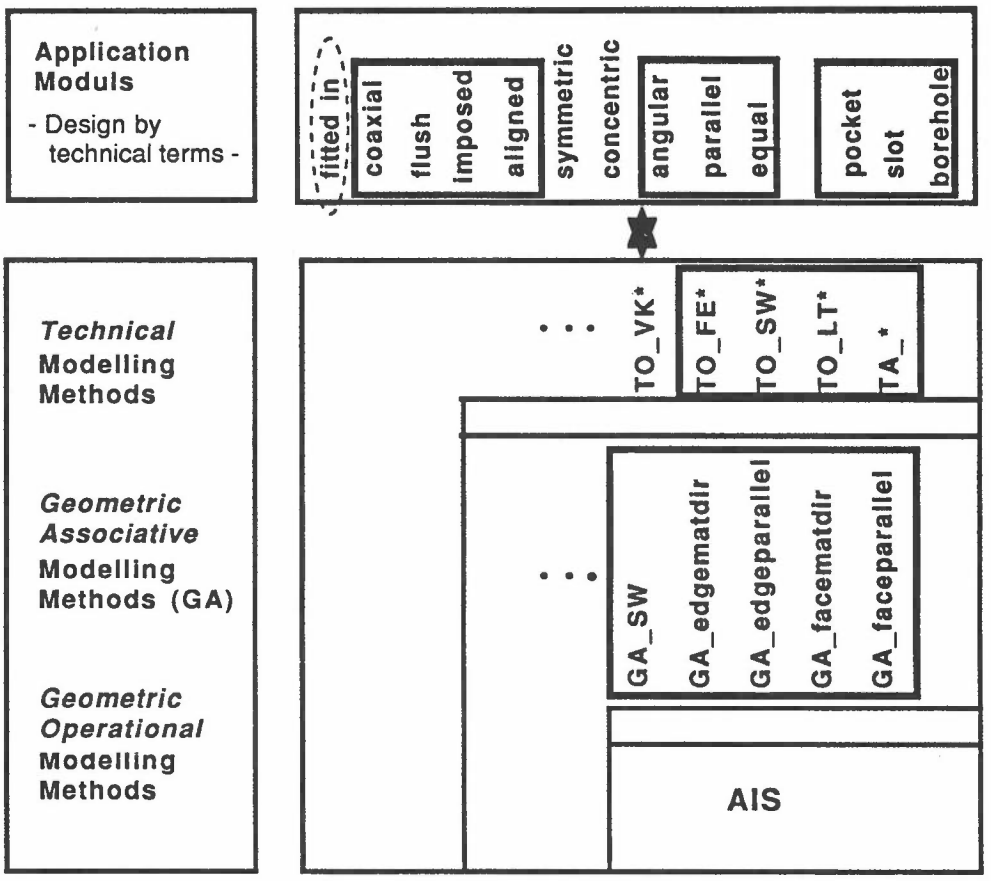
If the identified elements belong to different parts some kind of assembly operation is performed by using technical-associative transformation operations (TO_*). Objects which are related to the part by former use of technical terms are moved also. Actual work is spent on the combination of terms between the same two parts.

For example the operation "fitted-in" modifies the dimension of an action part as well as it makes the action part flush or coaxial to the reference part.

The operation can also be used for associative design by features, however only as long as a feature is not really inserted into the part (in terms of boolean operations).

If the identified elements belong to the same part the actions to be taken must switch from assembly modelling to part modelling which is in that sense a kind of dimensional changes within a part. This task can be performed now by a sequence of associative positioning and orienting and a subsequent dimensioning of a part.

Initial versions of the geometric associativities interface and the technical and technical-associative interface have been set up now and are included onto the next 4 pages.



Legend

finished in progress

TO_ -Technical Operational
TA_ -Technical Associative
FE* -Insertion of Form Features
LT* -Linear Transformations
SW* -Sweeping
VK* -Boolean Operations
GA_ -Geometric Associative
AIS -Application Interface Specification

Figure 5.2.3-1: Realisation Status

Geometric Associativities Interface (GAI) - Overview

Group 1 - Test Functions

ga_test_faceparallel
: test, if two two faces are parallel

ga_test_faceedgeparallel
: test, if an edge is parallel to a face

ga_test_edgeparallel
: test, if two edges are parallel

ga_test_coaxial
: test, if an edge is coaxial to a face
with a cylindrical or conical surface associated

ga_test_angular
: calculates the angle between two faces
or two edges

ga_test_matr
: test, if two entities in relation to an edge
have the same material direction

Group 2 - Insertion Functions

ga_ins_faceparallel
: test, if two faces are parallel
if true, then insertion of the geometric association
"parallel"

ga_ins_faceedgeparallel
: test, if an edge is parallel to a face
if true, then insertion of the geometric association
"parallel"

ga_ins_edgeparallel
: test, if two edges are parallel
if true, then insertion of the geometric association
"parallel"

ga_ins_coaxial
: test, if an edge is coaxial to a face
with a cylindrical surface associated
if true, then insertion of the geometric association
"coaxial"

ga_ins_faceangular
: calculation of the angle between two faces
insertion of the geometric association

"angular"

ga_ins_edgeangular
calculation of the angle between two edges
insertion of the geometric association
"angular"

ga_ins_pointdistance
insertion of an explicit distance between two
vertices

Group 3 - Modification Functions
---

ga_mod_parallel
test, if the modification of the value for the
association "parallel" is possible, if true then
translates the corresponding face and all
related loops, edges, vertices and associated
geometry

ga_mod_coaxial
test, if the modification of the value for the
association "coaxial" is possible, if true then
dimensions and translates the corresponding face and all
related loops, edges, vertices and associated
geometry

ga_mod_angular
test, if the modification of the value for the
association "angular" is possible, if true then
rotates the corresponding face and all
related loops, edges, vertices and associated
geometry

ga_mod_edgeparallel
makes two edges of two different entities parallel
by rotating one entity

ga_mod_faceparallel
makes two surfaces of two different entities
parallel by rotating one entity

ga_mod_matr
turns the material direction of an edge and a surface
or two surfaces or two edges to the
same/opposite direction by rotating one entity

Group 4 - Deletion Functions
---

ga_del
deletes a specified association

Technical - Associativities Interface (TAI) - Overview

Group 1 - Test Functions

ta_test_taentity
: test, if a technical association entity exists that belongs to the entity

ta_test_auxentity
: test, if a technical auxiliary entity as construction line or space exists that belongs to the entity

ta_test_axis
: test, if an axis exists that belongs to the entity

Group 2 - Insertion Functions

2.1 Technical auxiliary entity insertion

ta_ins_conspace
: defines an entity as a construction space

ta_ins_conface
: defines an face as a construction face

ta_ins_conline
: defines a line as a construction line

ta_ins_conpoint
: inserts a construction point

ta_ins_axis
: inserts an axis

2.2 Technical association insertion

ta_ins_prod
: inserts the association "product" between two assemblies or entities

ta_ins_assembly
: inserts the association "assembly" between two entities

ta_ins_entity
: inserts the association "entity" between two faces or form features

ta_ins_formfeature
inserts the association "form feature" between two faces or edges

Group 3 - Modification Functions

3.1 General position modification

ta_mod_trans
translates all entities of one assembly except the reference entity

ta_mod_rot
rotates all entities of one assembly except the reference entity

3.2 Technical position modification

ta_mod_flush
makes an entity flush to another

ta_mod_alignface
makes two entities align to a surface

ta_mod_alignedge
makes two entities align to an edge

ta_mod_coaxial
makes two entities with a cylindrical or conical surface associated coaxial and inserts the axis

ta_mod_ontop
puts one part on top of a second part after identification of two surfaces

Group 4 - Deletion Functions

ta_del_taentity
a technical association entity as flush, coaxial, "product" or "assembly" is deleted

ta_del_auxentity
a technical auxiliary entity as construction line or construction point is deleted

5.2.4 Design by Features

Design by features is an example of a technical design method using a conventional technique. Form features is a large and complex area of study. In /4/ it is shown that features have very many uses, in design, process planning, NC data generation, dimensioning and tolerancing, inspection, etc., etc. Although often discussed, there is no common definition available, and most authors say: "A form feature is a region of interest on the surface of a part" (see fig. 5.2.4-1).

We agree with that point of view in principle because the definition of form features gives the freedom to include not only collections of faces **but** also e.g. edge modifies like bevelled, or surface finish, tolerances, or feature specific dimensions, or feature specific associations to related objects, i.e. functional interfaces between parts or between features.

The latest definition given by Prof. Pratt is as follows /5/ (see fig. 5.2.4-2), and covers the aspect, that the definition is valid not only for B-Rep-based modellers but also for CSG-based ones:

> The features input by the designer do not necessarily correspond with the features required for other operations, and some kind of automatic feature recognition (or at least feature transmutation) will be necessary in the integrated systems of the future.

To handle features, there are two possibilities in principle:

First, to offer a set of form features during product design, for example by form feature menues.

Second, to do the construction first in a pure geometric way and to analyze the parts **after** they are geometrically complete described (so called ex-post analysis).

Two proceedings are known in principal for the situation, that a part is already geometrically described:

First, the manual specification of form features by an interactive process.

Second, the automatic analysis of a geometrical complete described part (see fig. 5.2.4-3).

Since limited time remains in the current CAD*I Project it is proposed to avoid these complications by requiring

Form Feature

A region of interest on the surface of a part

Different Views

Design

Analysis

Manufacture

Quality Assurance

Figure 5.2.4-1: Definition and uses of form features

What is a Form Feature?

A related set of elements of a product model, conforming to characteristic rules allowing its recognition and classification, and which, regarded as an independent entity, has some function during the life cycle of a product.

Figure 5.2.4-2: What is a form feature

the designer to work in terms of the specific set of features appropriate for the manufacture of parts by conventional machining methods. This will result in the specification of a modeller tailored for that particular application, and not in general suited for other purposes. Nevertheless, this approach will permit identification of many of the requirements for design by features, in terms of data structures, informational considerations and operational needs.

One other complication of this type of design will also be neglected. This is the necessity for each individual organization to be able to configure the system to work in terms of the classes of features it finds most convenient. This will vary between companies, depending for example on their particular manufacturing capabilities. The proposed simplification allows us to work in terms of a relatively small number of pre-defined feature classes and to concentrate on the types of feature operations required in a feature-oriented design system.

Design for Manufacture of Machined Parts

A good designer will provide a product design which is both functional and feasible to manufacture. This requires some knowledge of manufacturing processes, since even the use of today's sophisticated solid modelling systems does not prevent the generation of designs which are difficult - sometimes impossible - to manufacture. Forcing the designer to work in terms of machinable features will to some extent alleviate this problem /6/. Furthermore, this type of approach leads itself to the later implementation of "advisory" systems which provide manufacturing information within the system, available to the designer when appropriate /7/.

The suggested approach allows the design process to proceed by simulating effect of machining operations. The material removal is represented by the use of set theoretic (boolean) operations /6/ or generalizations of such operations /6/. This method of design avoids to some extent the notorious difficulties of language and terminology currently experience by designers /8/.

Before investigating the modelling aspects of machined features some preliminary work is necessary. The required tasks are

(i) Identification and classification of the most commonly used machined features in EEC countries.

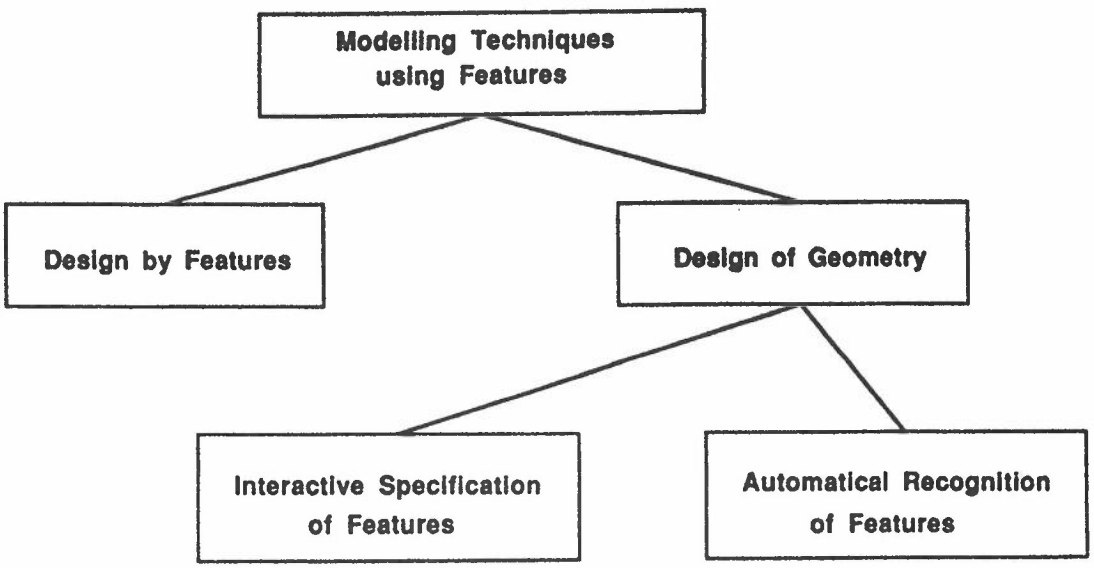


Figure 5.2.4-3: Principles for handling features

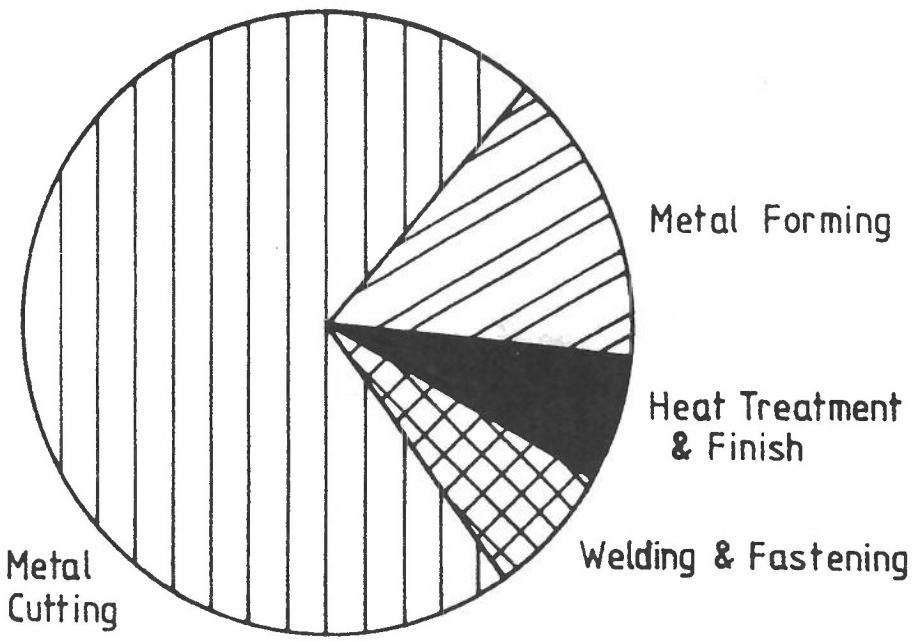


Figure 5.2.4-4: Manufacturing process in the UK engineering industry

(ii) Determination of the most suitable type of solid modeller for implementation of these feature classes.

For example, it is shown in /9/ that in the United Kingdom engineering industry metal removal operations account for about 7o % of manufacturing operations. Of these, some 6o % are the conventional operations of milling, drilling and turning (see fig. 5.2.4-4 and fig. 5.2.4-5).

Despite its simplicity, the proposed system will be designed to handle interactions between features and patterns of features. If time permits a basic demonstration system will be developed. Consideration will be given to the problems of interfacing to libraries of standard or user-defined features for design, and to libraries of machine tools and cutting tools for the planning of manufacture.

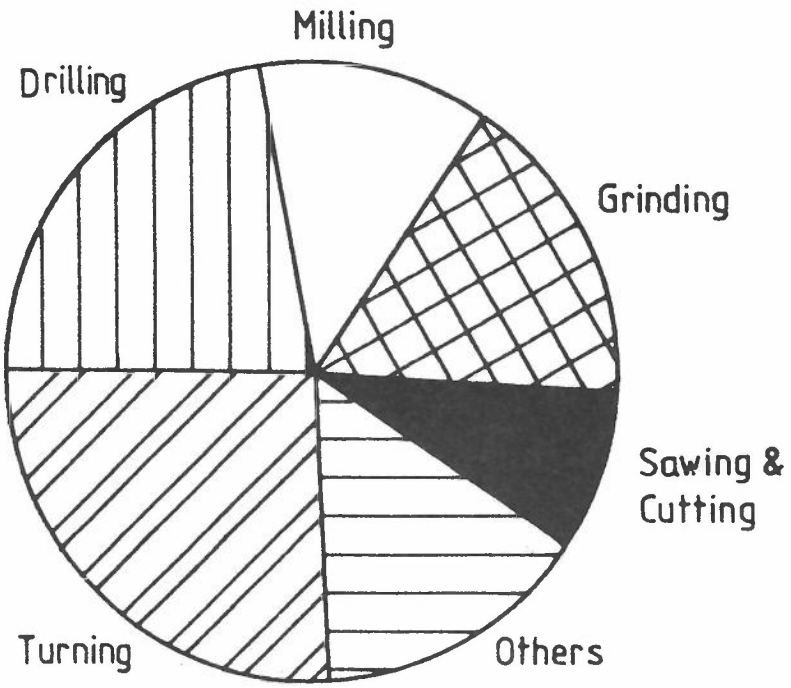


Figure 5.2.4.-5: Distribution of Metal Removal Operations in Fig. 5.2.4.-1

5.2.5 Relevance of STEP

The reason the work within the ISO/TC184/SC4/WG1 is relevant for WG 5, is mainly the model definition, which contains sections about assembly structures or feature definitions /1o/. The following figure shows an example for an implicit bore-hole (fig. 5.2.5-1):

Indeed it seems, that STEP is on the way to standardize all kinds of technical elements, which are the "atomic" components within the DTT-Modeller.

A new approach also discussed within the STEP community is the definition of standard parts. This happens because the German manufacturing industry uses a large number of standard parts. Therefore a library of standard parts should be available for CAD systems.

Effective integration of a standard parts library into a CAD-system requires:

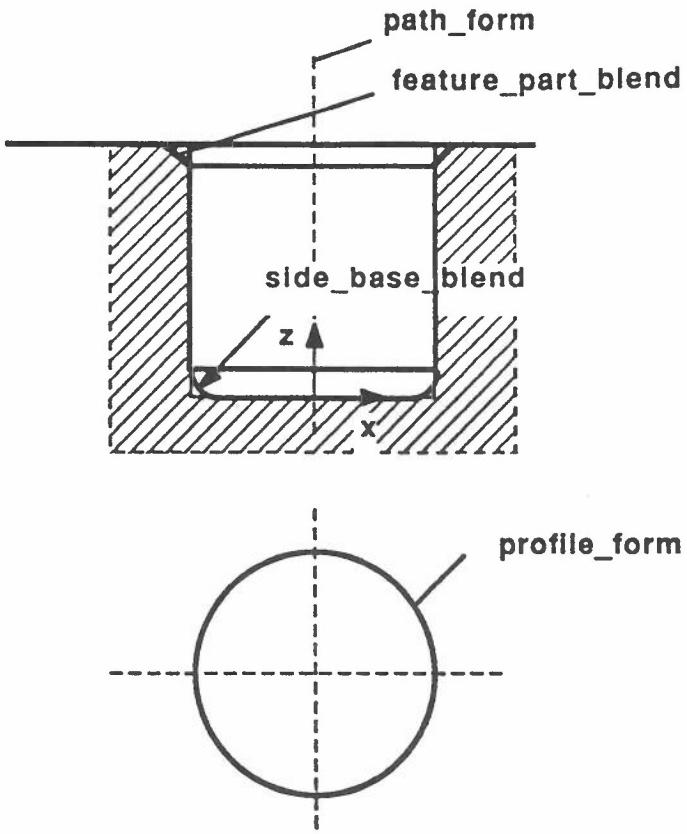
1. Support for the exchange of standard part libraries
2. Support for the definition of additional standard parts
3. Support different modes of representation (e.g. perspective view, symbolic representation...)
4. Store the parametric definition of a standard part separately from the table of standard parameter values
5. Represent the context in which standard parts should be used and in particular functional compatibility constraints between mating parts e.g. the size of a bolt and it's corresponding hole.

Similarities were found between the STEP form feature proposals and the DIN standard parts definition. The structure of standard parts as proposed within the DIN standard can be mapped into form features available in STEP. However the most important elements of the standard parts definition are lost as STEP can neither represent parametric models nor the tables of standard parameter values.

A comparison between the STEP form feature concept and the DIN standard part concept is given in fig. 5.2.5-2.

Therefore initial attempts have been made to use the STEP format for exchanging data from the DTT-Modeller to the outside world.

A STEP Pre- and a Post-processor have been written for the DTT-Modeller. Actually, these processors are based on the Washington Edition (Feb. '88) and support only so-called "Manifold-Solid-B-Reps". Also only geometry



implicit_form_feature	implizites Formelement
related_constraints ^	Zwangsbedingungen
implicit_depression	implizite "Einstülpung"
feature_part_blend ^	Fase oder Rundung
swept_depression	Gesweepte "Einstülpung"
feature_location ^	Lokales Koordinatensystem
in_swept_depression	"Hineingesweepte Einstülpung"
profile_form	Profilgeometrie
path_form	Pfadgeometrie
side_base_blend	Fase oder Rundung

Figure 5.2.5-1: Implicit bore-hole in STEP

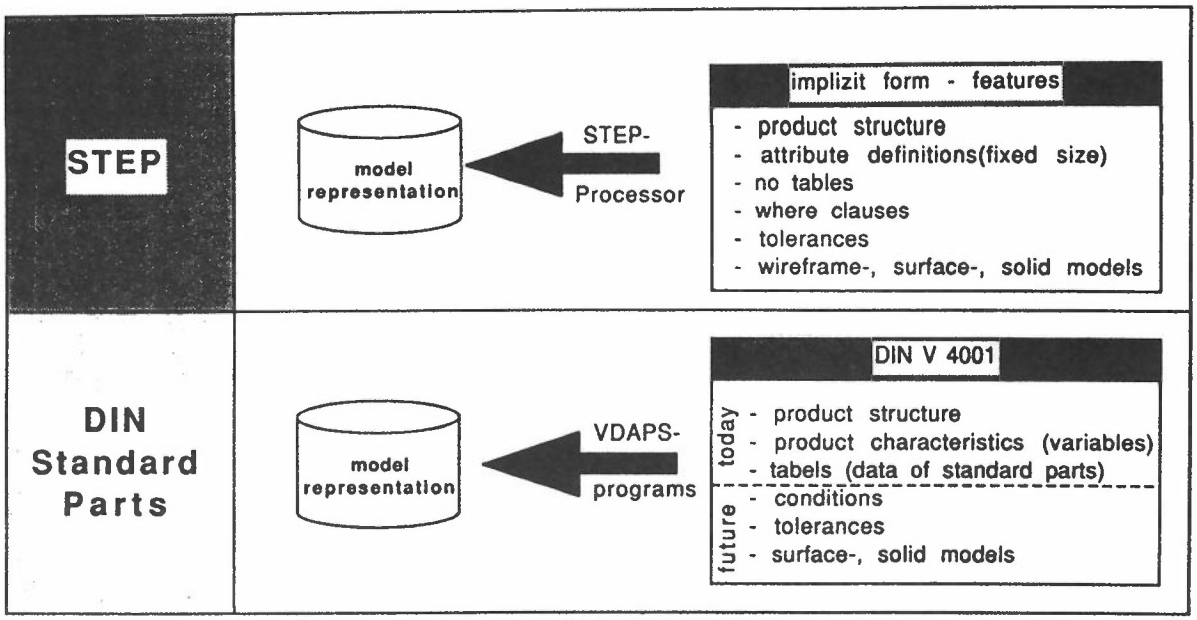


Figure 5.2.5-2: Comparison of STEP form feature concept with DIN standard parts concept

supported by the DTT-Modeller has been implemented (see section 5.2.2).

However, initial experiences also have been made in processing assembly and for feature structures. The only issue missed in all standardized data exchange proposals is a possibility for the exchange of associativities.

In principal there is now a link between a major portion of the CAD*I work as illustrated in fig. 5.2.5-3.

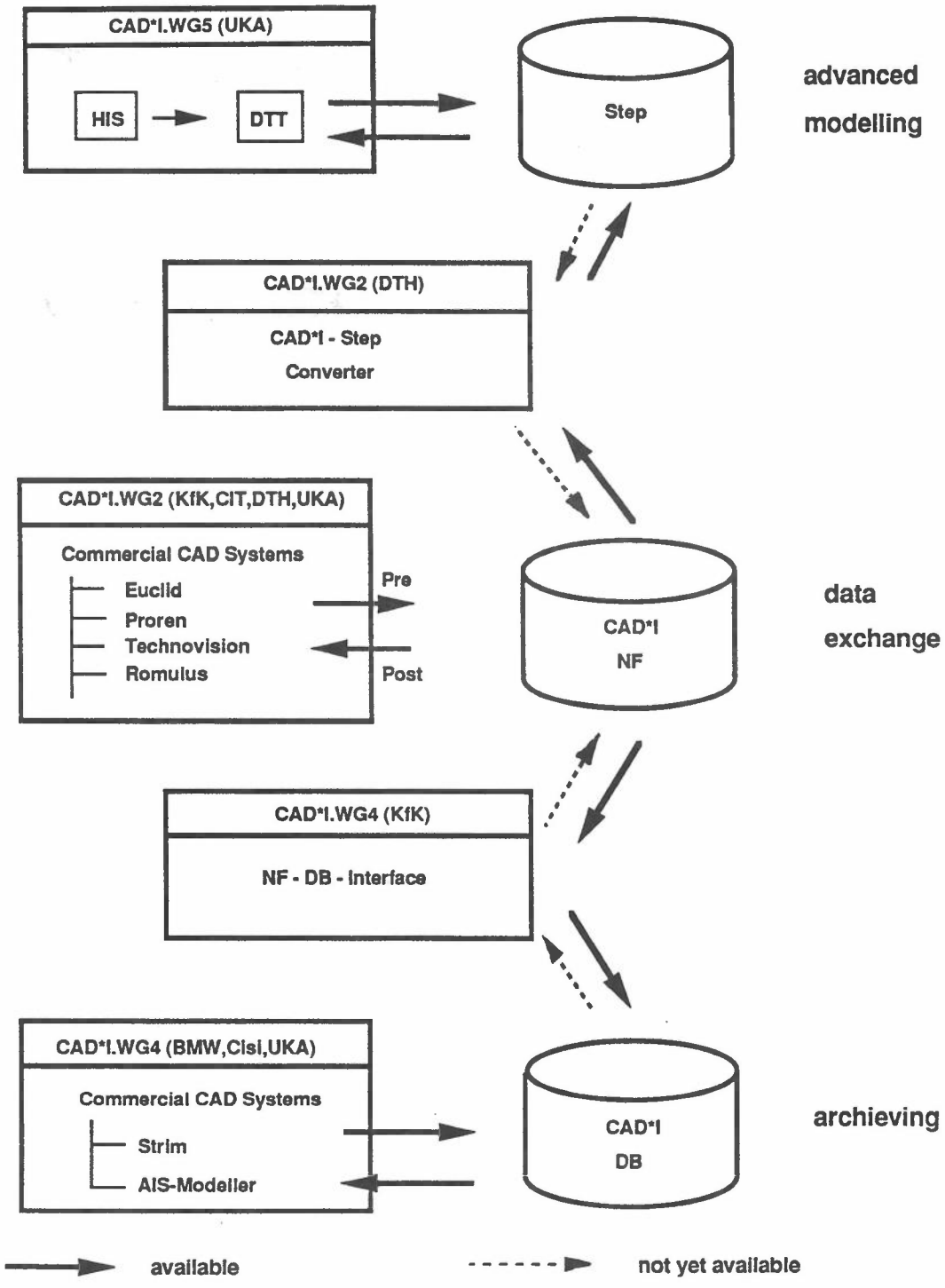


Figure 5.2.5-3: Link between CAD*I work

5.3 Results

5.3.1 HIS

A final prototype is available. The program works on a VAX under VMS V5.0 operating system.

For the user interface GKS level 2C and the DECWindow system is required. The existing GKS implementation supports all VAXStation devices.

5.3.2 DTT

Initial implementation being tested. Final version expected for the end of 1988.

The program works on a VAX under VMS V5.0 operating system.

For the user interface GKS level 2B is required. The used GKS implementation supports all Tektronix 41xx and 42xx devices.

5.3.3 Interfaces

Initial version of HMMI reviewed by CIT.

Initial version of GAI to be reviewed by CIT.

Initial version of TMI being reviewed by CIT.

5.3.4 Features

- Initial approach for design by features is implemented within the DTT-Modeller.
- A trial implementation has also been made of the feature modelling concepts for the ROMULUS-Modeller. This work indicates that the suggested approach is a feasible one. A report on this aspect of the CIT work is in preparation.

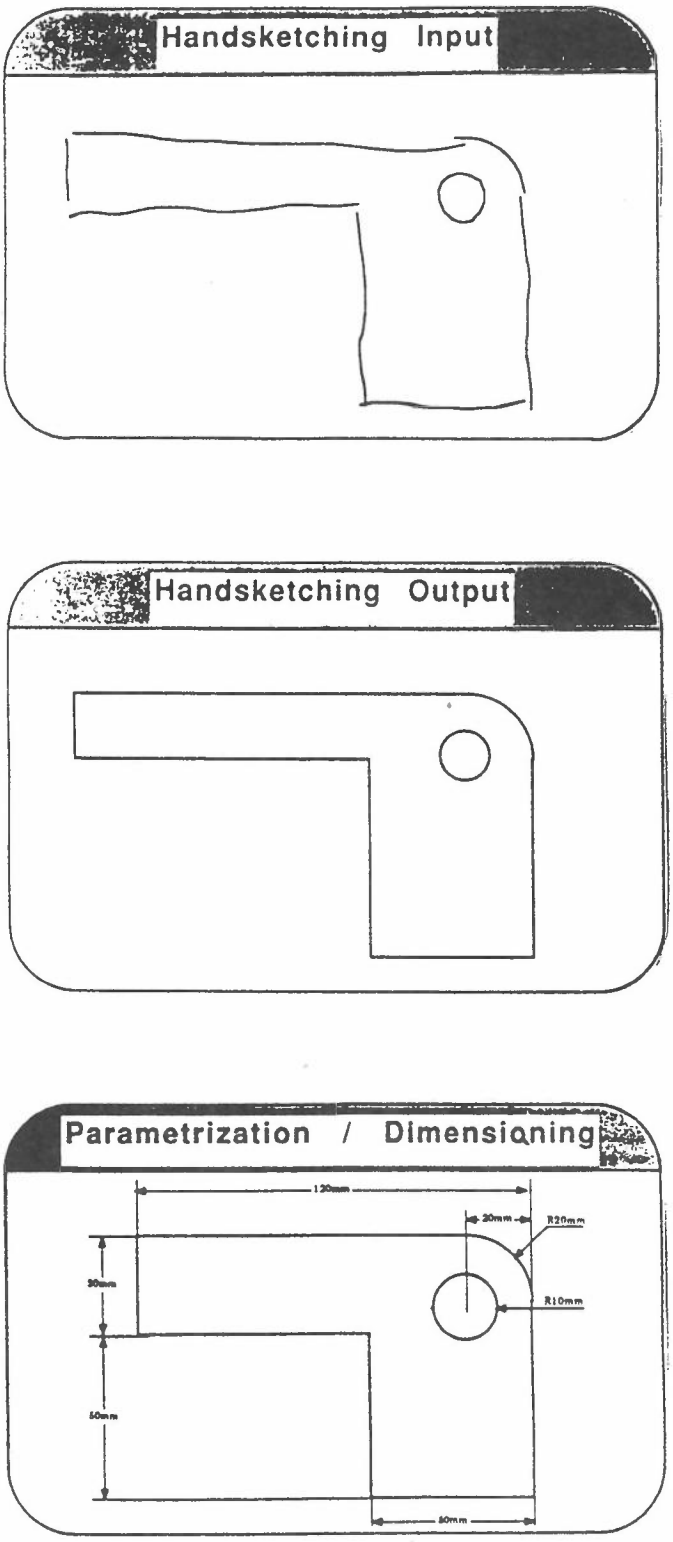
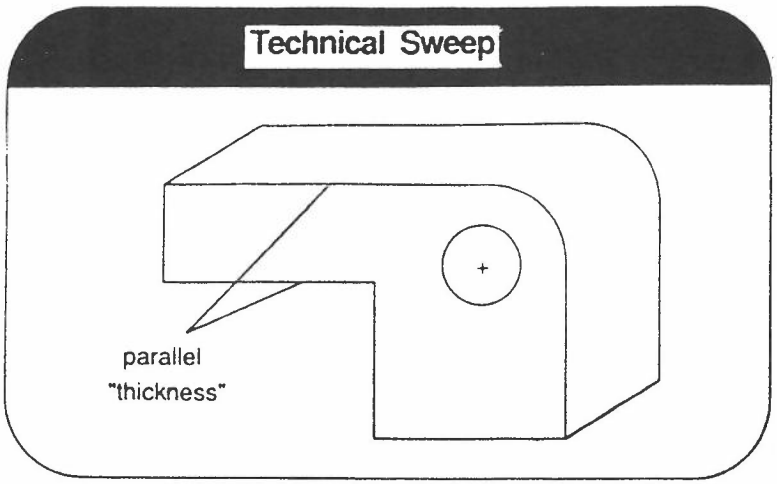
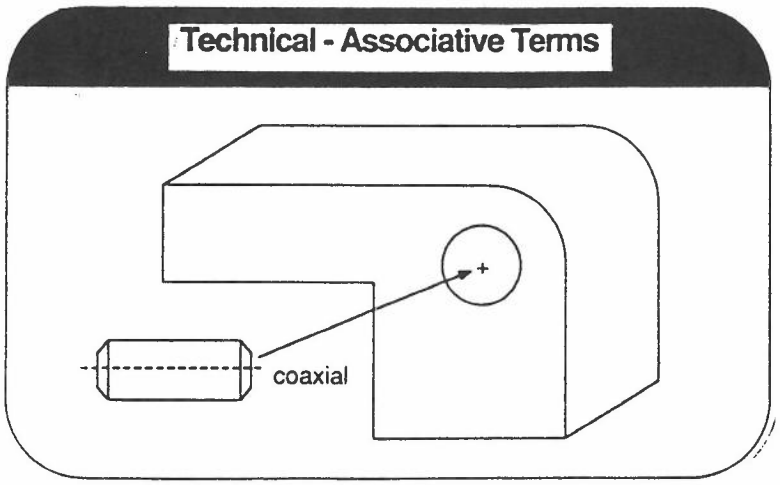


Figure 5.4-1: Design Sequence (1)





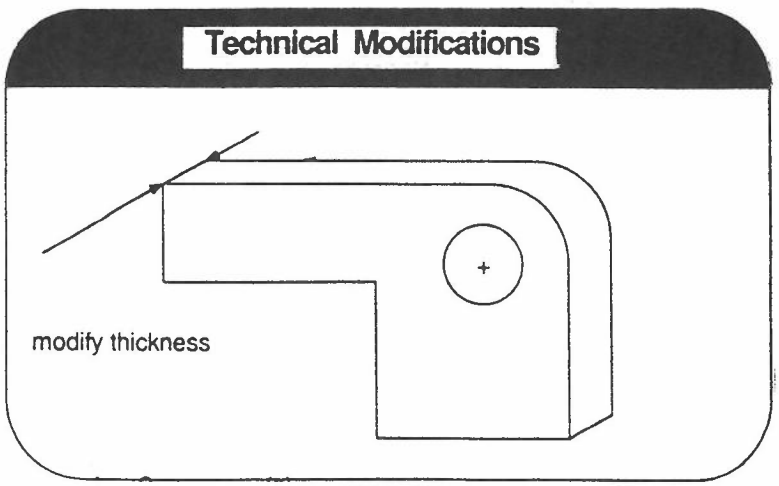


Figure 5.4-2: Design Sequence (2)

5.4 Outlook

Features

Further research is planned to be done in that field by Cranfield Institute of Technology/UK using the ROMULUS Modeller. A report on the ongoing work is in preparation.

Demonstration

As a final presentation of the work performed in the CAD*I project a HIS and a DTT-Modeller demonstration to a wider public is planned (e.g. during the ESPRIT Conference '89). Figure 5.4-1 and figure 5.4-2 describe a possible scene of the demonstration.

5.5 References

/1/ I. Bey, J.Leuridan, ed.
CAD*I CAD Interfaces, Status Report 3, KfK-PFT 132, March 1987

/2/ I. Bey, J. Leuridan, ed.
CAD*I CAD Interfaces, Status Report 4, KfK-PFT 139, March 1988

/3/ Grabowski, H., Anderl, R., Rude, S.: Technical Term Dictionary with Related Geometric and Technical Semantics, WG5 Report, ESPRIT CAD-Interfaces (CAD*I), KfK 1986

/4/ Pratt, M.J.: Form Features and their Applications in Solid Modelling; Course Notes Vol. 26, Advanced Topics in Solid Modelling, SIGRAPH 87 Conf., Anaheim, California

/5/ Pratt, M.J.: The Form Feature Concept for Advanced Modelling, 3th Int. Workshop on CAD-Interface, Lyngby/Denmark, 1988

/6/ Acar, B.S., Case K., Bennaton, J., Hart, N.: Evaluation of New Approach to Design and Process Planning; in Proc. 4th Conf. on UK Research in Advanced Manufacture, London, 1o-11 Dec. 1986, pp 131-135

/7/ Acar, B.S., Case, K.: Learning Studies in the Use of CAD Systems; in preparation

/8/ Pratt, M.J.: Conceptual Design of a Feature oriented Solid Modeller; Internal Report in preparation. General Electric Corporate Research & Development Center, Schenectady, N.Y.

/9/ Fifth Survey of Machine Tools and Production Equipment, Metal Working Production, London 1984

/1o/ ISO: Testing Draft; IPIM (Washington) Part 1, 2, and 3 (N215, N215 and N216); ISO TC184/SC4/WG1 1988

6. Working Group 6: FEM Model Description

6.1 Introduction

This report contains a summary of the last period of WG6 activities. A detailed description of WG6 activities and achievements is presented in the Final Report "Specification for Exchange of Product Analysis Data" (Springer Verlag, Heidelberg).

6.2 Summary of Final Report

One major goal of ESPRIT Project 322: CAD Interfaces (CAD*I) is the development of techniques for the exchange of product analysis data.

The Final Report describes a set of specifications for the exchange of product analysis data. A syntax for representing the analyis data in a neutral file is presented; this syntax is formally described in Backus-Naur form. A reference model for the finite element (FE) analysis data is given which uses a formal data modelling language to define the fundamental data entities, their attributes and interrelations. Syntactical expressions (statements) for entities in these reference models are worked out. The overall internal structure of the neutral file, and the interaction between the neutral file and the computing environment via pre- and post-processors are presented.

The final part of this report presents an invvestigation into the problems of transferring geometry data between CAD and FE systems and proposes some initial heuristic rules.

6.3 Achievements

The achievements reported are:

1. A definition of the syntax for the files.

2. A reference model for the data entitites that are relevant in analytical product analysis and syntactical expressions for the data entities.

3. A lay-out of concepts on the overall structure of the files and interaction of the files with the computing environment via pre- and post-processors.

4. Investigation into heuristic rules for the transformation of CAD geometry to FE geometry.

6.4 Conclusion

The prime objective of the CAD*I project is to develop interfaces for transfer of product data between the different processes of the CAD environment. This environment is often considered to include processes, such as FE analyis and CAT and is therefore sometimes referred to by the term Computer Aided engineering (CAE). It is therefore essential that interfaces allow for transfer of both product definition data and product analysis data.

The work that is reported on in this paper concentrates on the aspect of exchange of product analysis data. With the aim of having a highly unified set of specifications for both product definition and analysis data, the work has not been performed independently from the Working Groups active in the area of specifications for exchange of product definition data but rather in close collaboration. The results of this essential interaction are reflected in various parts of this report, so that the results that are reported truly contribute to the prime objective of the CAD*I project.

7. Working Group 7: Dynamic Model Optimization

7.1 Introduction

Since this report should cover the last period of the WG7 activities, the technical report presented here will contain a general description of the global WG7 task and work and the table of contents of the final WG7 report. Publication of this final report is in preparation (Springer Verlag, Heidelberg).

7.2 Dynamic model optimization: aim and approaches

When designing a structure one must take into account more than only the primary task to be performed by the structure. The end product should execute that task safely, comfortably and under a wide variety of external conditions. To achieve this goal the design phases will consist of interactions between analytical model techniques (e.g. finite element models) and experimental test approaches (e.g. material tests, static and dynamic load tests).

In this work especially the vibratory behavior of a structure is investigated. A thorough dynamic analysis of complex structures must not only employ both theoretical and experimental analyses but also combine these two approaches to provide a unified description of the structure. Unfortunately reasonable correlation between analytical and test results is usually not achieved until repeated changes are made in the analytical model. Nevertheless, combining both approaches remains important since both techniques are valuable. The finite element method can correct for the incompleteness (e.g. not measured rotation degrees of freedom, limited number of mode shapes) of the experimental approach. Similarly the experimental dynamic analysis compensates for the idealizations made by the finite element analysis (e.g. joints, bearing). The objective of jointly applying analytical and experimental approaches in design for structural system optimization is to minimize dependence on the analyst's intuition in obtaining mathematical models once a specimen (e.g. prototype) of the structure exists. The need for accurate mathematical modeling from drawings will continue as a requirement in the design process. Similarly important is the evolution in measurement techniques towards higher accurancy and

completeness. A secondary benefit is that experience with model optimization is likely to contribute to improved intuitive modeling and measurements.

Dynamic model optimization or updating techniques will use the results of dynamic tests (time histories, frequency response curves, measured resonance frequencies and mode shapes) in order to modify parts of the finite element model matrices, such that the updated model describes the actually measured dynamic behavior. The ultimate result of those techniques is a matrix model which is more reliable for use in further analysis techniques: prediction of the effect of modifications, system synthesis, noise analysis, sensitivity analysis, structure monitoring, design optimization ... Since such techniques can be applied on an improved model, less verification steps with the real world structure will be necessary. Such reduction of the number of prototypes in the (re)design phase represents a significant cost decrease. Typical applications are the prediction of the dynamic effect of weight reduction in automobile cars or aircrafts, the location of major defects in oil platforms from changing model parameters, improved models for control, or tuning of a machine tool towards a desired dynamic behavior in order to improve manufacturing accuracy.

A model optimization case evolves over several steps. The first phase can be called the **compatibility** phase. It contains firstly the practical interface between finite element program and experimental modal analysis package on one side, and the optimization package on the other side. Analytical and experimental description probably use different coordinate systems. The second part of the compatibility phase consists of matching those coordinate systems.

The following step in an optimization case is the **correlation** phase. It consists of techniques for comparing the mathematical modal data with the experimental modal analysis results. It establishes to what extent mathematical model and test are describing the same physical structure. In order to get meaningful results from the optimization the original descrepancy between model and test should generally be limited. This phase also takes care of dimensional differences between experimentally and analytically obtained mode shape vectors.

If sufficient correlation exists the **optimization**, or **updating**, itself can start: experimental and mathematical results will be brought into agreement, primarily by

modifying parts of the matrix model, based upon test results.

In this work two updating approaches will be discussed: a frequency domain full size method and a time domain reduced size method. The major aim of both approaches is identical: create an analytical model that describes the measured dynamics as close as possible. The background and the approach itself however are different. The **time domain** method will update the analytical model based upon time responses. This approach will furthermore first reduce the size of the original finite element model, by taking into account only the lower order modes. The result of this technique is a small size corrected model especially suitable for control purposes. The **frequency domain** method, however, will update the full size finite element model, based upon the improved agreement with the measured resonance frequencies and mode shapes. The major aim of preserving the full size finite element model is to improve the interpretability of the model changes. This feature generally justifies the larger amount of computer effort needed.

7.3 Contents of the final WG7 report

CORRELATION & UPDATING:

1 Background information:

Definition of correlation problem
Definition of updating problem
Approaches

2 Correlation analysis:

2.1 Introduction
2.2 Orthogonality check
2.3 Modal Assurance criterion
2.4 Coordinate modal assurance criterion
2.5 Completion/smoothing of measured modes

2.5.1 Problem definition
2.5.2 MAC-MSF method
2.5.3 Orthogonal projection method

3 Updating approaches:

3.1 Time domain, reduced size approach
3.1.0 Introduction
3.1.1 Formulation of the finite element model
3.1.2 Reformulation into state space formulation
3.1.3 Calculation of the lowest eigenvalues
3.1.4 Establish a reduced time discrete state space model
3.1.5 Establishing the FE-matrices and recycle
3.1.6 Parameter estimation in reduced state space model
3.1.7 Simulation using the reduced state space model
3.1.8 Examples

3.2 Frequency domain, full size technique
3.2.0 Introduction
3.2.1 Sensitivity analysis
3.2.2 Iterative updating
3.2.3 Example

4 Program development

5 Future developments

7 Conclusions

Appendices

References.

8. Working Group 8: Experimental Dynamic Structural Analysis

8.1 Synopsis of WG8 Final Report

A task group of the ESPRIT Project No 322, CAD Interfaces, has studied methods for integrating FEM techniques and testing methods for structural dynamic analysis. The following presents a summary of the final report. Publication of this final report is in preparation (Springer Verlag, Heidelberg).

The methods of integration refer to the development of a synergetic data platform: a coherent set of data including data from FEM and test, correlation analysis, and FEM model updating. The latter has the objective to improve the mathematical analysis models of a product as efficiently as possible using test data from prototypes. The modified analysis models are then available for more accurate predictions of operating behaviour that can not easily be simulated on the available prototypes.

The study of these problems is really only meaningful provided that sufficient accurate data can be obtained from prototype testing. The set of experimental test techniques to extract dynamic model parameters from structures is also referred to as experimental modal analysis. A work group of the project team was defined to study and improve methods for modal analysis : Working Group 8 (WG8).

Two classes of methods are in use to perform modal analysis: phase resonance testing and phase separation testing. The former methods actually allow to measure the dynamic parameters (frequency, damping, mode shapes) one by one using multiple input sine testing with force appropriation. These methods are the oldest, rather hardware intensive, and today really only in use for aircraft and aerospace applications. On the other hand, the phase separation methods are based on the analysis of response data of the structure, from broadband or smallband excitation, or from predefined initial conditions. The response data normally contains information on several dynamic parameters simultaneously. An additional analysis procedure is needed to identify estimates for all the dynamic parameters that are observable from the data. This class of methods has become very popular with the advent of flexible digital signal processing systems - e.g. Fourier analysers - that allow to acquire conveniently the required data. Those methods are therefore now in common use for a large

variety of industrial applications, both in prototype identification and trouble shooting . The quality of the results is however much dependent on the particular measurement technique and signal processing algorithms used when acquiring the data, and of the analysis algorithms used to estimate the dynamic parameters.

WG 8 has researched both the measurement and analysis aspect of the phase separation testing methods. The contributions of WG 8 in this field are described in this report, not as individual items, but structured so as to represent a selfcontained presentation on some important phase separation testing techniques.

For the measurement aspect, the study has concentrated on the further development of multiple input estimation techniques of frequency response functions. Contributions are specifically in the area of better techniques to judge correlation between the excitation forces for optimal measurement quality, and the application of the Total Least Squares method for identifying frequency response functions from measured input and output spectra /1-6/. These correlation analysis techniques formed the basis for spin-off's developments to identify noise sources in operating environments /7/.

On the analysis aspect, the study has concentrated on the development of global modal parameter estimation techniques. These methods aim at identifying all structural dynamic parameters by analyzing simultaneously all available measurement data; they use at best the redundancy in the data, and allow to minimize operator interaction. Contributions have been done primarily by developing practical methods that analyse data in the frequency domain. Those techniques allow to identify the characteristic equation - and therefore the dynamic parameters - of the underlying continuous model, from frequency data, possibly with uneven frequency resolution /8,9/. This stands in contrast with the more popular techniques that analyze data in the time domain, and bear inherent limitations, as they identify a discrete system model on evenly spaced time data.

Related to the analysis, WG8 has also developed methods that allow easier execution of the complex analyses, and better validation of the analysis results /10,11/.

WG8 has also developed an experimental modal analysis technique that uses multiple input sine excitation, but not with force appropriation (as for the phase resonance testing). This technique is called "Stepped Sine Testing" /12-14/. The data measured with this method essentially

describe several dynamic characteristics simultaneously, and should be analyzed using the analysis methods used with phase separation testing. This technique combines the advantages of traditional phase resonance testing (harmonic excitation high signal-to-noise ratios, avoiding signal processing errors) and modern phase separation testing (measurement time, identification of more modes). As the frequency steps for this technique can be made a function of the information contents of the data, compressed data sets with unequally spaced frequency steps can be measured quite fast.

The working group leader of WG8, LMS International-Belgium, has implemented several of the new developments of WG8 in its new Computer Aided Dynamic Analysis system for UNIX workstations, the CADA-X system. The system is configured around a state of the art engineering computer workstation, and a modular front-end for data sampling and generation of excitation signals. The CADA-X software library includes modules for structural dynamic analysis. In particular the CADA TEST/FOURIER MONITOR module, a general purpose Fourier analyzer, and the CADA MODAL/ANALYSIS module, an advanced modal analysis package have been enhanced to include the research contributions of WG8. The CADA-X system is actively marketed on a world wide basis.

8.2 References

/1/ J. Leuridan, and B. Rost (1985)
"A Comparison of Least Squares and Total Least Squares for Multiple Input Estimation of Frequency Response Functions", ASME Paper No. 85-DET-105, 6 pp.

/2/ J. Leuridan, B. Rost (1985)
"Multiple Input Estimation of Frequency Response Functions : Diagnostic Techniques for the Excitation", ASME Paper No. 85-DET-107, 5 pp

/3/ D. De Vis, J. Lipkens, U. Vandeurzen and J. Leuridan (1985)
"Application Study of Advanced Structural Analysis Techniques on a Motorbike Frame", Proc. of 10 th Int. Seminar on Modal Analysis, 30 Sept - 4 Oct 1985, K.U. Leuven, Leuven, 8pp.

/4/ J. Leuridan, D. De Vis, H. Van der Auweraer and F. Lembregts (1986)
"A Comparison of Some Frequency Response Function Measurement Techniques", Proc. of 4th Int. Modal Analysis Conf., pp. 908 - 918, 3 - 6 February 1986, Los Angeles, Ca.

/5/ J. Leuridan and H.. Van der Auweraer (1986)
"Comparison of Recent Developments for Estimating Frequency Response Functions in Structural Analysis", Proceedings of EUSIPCO - 86, Signal Processing III : Theories and Applications, pp 263 - 266, Elsevier Science Publishers B.V.

/6/ J. Leuridan (1986)
"The Use of Principal Inputs in Multiple Output Data Analysis", The International Journal of Analytical and Experimental Modal Analysis, Vol 1/3, pp. 1-8, Society of Experimental Mechanics Inc.

/7/ D. Otte, K. Fyfe, P. Sas and J. Leuridan (1988)
"Use of Principal Component Analysis for Dominant Noise Source Identification" Int. Conf. on Advances in the Control and Refinement of Vehicle Noise, 22-24 March 1988, Birmingham, UK, 8 pp.

/8/ J. Leuridan, J. Lipkens, H. Van der Auweraer and F. Lembregts(1986)
"Global Modal Parameter Estimation Methods : An Assessment of Time versus Frequency Domain Implementation", Proc. of the 4th Int. Modal Analysis Conf., pp 1586 - 1595, 3-6 February 1986, Los Angeles, Ca..

/9/ H.Van der Auweraer and J. Leuridan (1987)
"Multiple Input Orthogonal Polynomial Parameter Estimation", Journal of Mechancial Systems and Signal Processing, Vol 1/3, pp. 259-272, Academic Press.

/10/ F. Lembregts, R. Snoeys and J. Leuridan (1987)
"Application and Evaluation of Multiple Input Modal Parameter Estimation", Int. Journal of Analytical and Experimental Modal Analysis, Vol 2/1, pp 19-31, Society of Experimental Mechanics Inc.

/11/ J. Lipkens and J. Leuridan (1987)
"The Use of Modal Participation Factors to Determine Physical Poles", Proc. of 5th Int. Modal Analysis Conf., 6-9 April 1987, London, pp. 995-1002.

/12/ F.Lembregts, H. Van der Auweraer, P. Sas and J. Leuridan (1987)
"Integrated Stepped Sine Modal Analysis", Proc. of the 5th Int. Modal Analysis Conf., 6-9 April 1987, London, pp. 979 - 985.

/13/ F.Lembregts, P. Sas, H. Van der Auweraer and J. Leuridan (1987)
"Integrated Stepped Sine System for Modal Analysis", Journal of Mechanical Systems and Signal Processing Vol 1/4, pp. 415-424, Academic Press.

/14/ F.Lembregts, J. Leuridan, J. Lipkens and H. Van der Auweraer (1988)
"Comparison of Stepped-Sine and Broad Band Excitation to an Aircraft Frame", Proc. of the 6th Int. Modal Analysis Conf., 1-4 February 1988, Orlando, Florida, pp. 1706 - 1713.

9. Standardisation Activities

ESPRIT Project 322: CAD Interfaces (CAD*I) has been contributing to international standardisation efforts since the begin of project work to strengthen European influence in this area of vital importance for European industry.

A major goal is now reached with the elaboration of a draft proposal for the exchange of product model data, commonly known as "STEP", at ISO, the International Standards Organisation.

Important European contribution to the international standardisation efforts have been made by particular CAD*I contribution to the geometric specifications, the physical file format and the formal specification language used for STEP. The experimental geometric model exchanges between members of the CAD*I project using the closely related CAD*I neutral file specification have helped to validate some of the concepts before they were incorporated in STEP.

ISO has released to national standards bodies for review the draft proposal for a standard for the exchange of product model data. This proposal, commonly known as STEP, is the result of more than 4 years work by an ISO working group among them a group of CAD*I members who were specifically engaged with the solution of standardisation problems.

CAD*I work is going on to influence further international standardisation activities.