

# Engineering databases

Autor(en): **Werff, Klaas van der**

Objektyp: **Article**

Zeitschrift: **IABSE reports = Rapports AIPC = IVBH Berichte**

Band (Jahr): **40 (1982)**

PDF erstellt am: **17.09.2024**

Persistenter Link: <https://doi.org/10.5169/seals-30901>

## **Nutzungsbedingungen**

Die ETH-Bibliothek ist Anbieterin der digitalisierten Zeitschriften. Sie besitzt keine Urheberrechte an den Inhalten der Zeitschriften. Die Rechte liegen in der Regel bei den Herausgebern.

Die auf der Plattform e-periodica veröffentlichten Dokumente stehen für nicht-kommerzielle Zwecke in Lehre und Forschung sowie für die private Nutzung frei zur Verfügung. Einzelne Dateien oder Ausdrucke aus diesem Angebot können zusammen mit diesen Nutzungsbedingungen und den korrekten Herkunftsbezeichnungen weitergegeben werden.

Das Veröffentlichen von Bildern in Print- und Online-Publikationen ist nur mit vorheriger Genehmigung der Rechteinhaber erlaubt. Die systematische Speicherung von Teilen des elektronischen Angebots auf anderen Servern bedarf ebenfalls des schriftlichen Einverständnisses der Rechteinhaber.

## **Haftungsausschluss**

Alle Angaben erfolgen ohne Gewähr für Vollständigkeit oder Richtigkeit. Es wird keine Haftung übernommen für Schäden durch die Verwendung von Informationen aus diesem Online-Angebot oder durch das Fehlen von Informationen. Dies gilt auch für Inhalte Dritter, die über dieses Angebot zugänglich sind.

## Engineering Databases

Bases de données pour ingénieurs

Datenbanken im Ingenieurwesen

### Klaas van der WERFF

Dr. Ir.  
Univ. of Technology  
Delft, the Netherlands



Klaas van der Werff, born 1942, obtained his mechanical engineering degree at the University of Technology, Delft, the Netherlands. As a senior researcher at the Delft TU he was involved in the finite element formulation of the kinematics and dynamics of mechanisms. Present work is directed to CAD.

### SUMMARY

To investigate the applicability of Database Management System (DBMS's) for engineering purposes, some prototype-implementations have been performed. These implementations were based on the use of a Codasyl-type DBMS and a relational type DBMS. The applications involved include storage and manipulation of a finite element network. One of the major conclusions from the study is, that due to the fact engineering applications have to keep record of part-subpart relationships, geometric relationships as well as computational relationships, DBMS's are of limited use in an engineering environment. In particular, control structures in the application software tend to be too complex.

### RESUME

Afin d'examiner les possibilités d'application pour l'ingénieur des systèmes de gestion de bases de données (SGBD) un certain nombre d'expériences pilotes ont été réalisées. Elles ont mis en œuvre des systèmes répondant aux normes CODASYL et de SGBD relationnelles pour la manipulation de réseaux d'éléments finis. Les conclusions de l'étude conduisent à considérer que ces systèmes sont d'un intérêt limité dans l'environnement technique de l'ingénieur, dû au fait notamment que les relations à maintenir entre les structures et les sous-structures, sur le plan de la géométrie et du calcul, deviennent trop complexes par ce type de méthodes.

### ZUSAMMENFASSUNG

Um die Anwendbarkeit eines Datenbank-Management-Systems für Ingenieurzwecke zu untersuchen, wurden einige Prototypstudien durchgeführt. Diese Implementationen wurden auf einer «codasylen» und einer «relationalen» DBMS aufgebaut. Die Anwendungen schlossen die Speicherung und Manipulation eines finiten Element-Netzwerkes mit ein. Eine der hauptsächlichsten Schlussfolgerungen dieser Studie ist, dass DBMS in der Ingenieurumgebung beschränkt anwendbar sind, weil ingenieurmässige Anwendungen Teil-Unterteil-Abhängigkeiten, geometrische Abhängigkeiten und berechnete Abhängigkeiten aufzeichnen müssen. Im speziellen neigen Kontrollstrukturen in der Anwendungssoftware zu grosser Komplexität.



## 1. INTRODUCTION

Computer Aided Design is rapidly advancing. The application of databases is an important component in the development, particularly because the use of databases is a kind of integrating factor in order to come to computer aided engineering.

In order to get insight in the essentials of engineering data base management systems, in 1980 a project group has been formed in which 9 companies and institutions cooperated. This project group is one of the activities of CIAD, a dutch institute for computer applications in engineering practice. The results will be made available in the form of a final report.

The study is directed to the following topics:

- formulation of a set of demands which has to be met by an engineering application of DBMS's,
- investigation whether or not the DBMS's selected provide adequate facilities to meet the demands,
- providing a checklist which covers all relevant aspects in selecting a proper DBMS.

In addition the applicability of DBMS's for engineering purposes has been investigated. To this purpose some prototype implementations have been performed. This last aspect gets much attention in the present publication.

## 2. WORKING METHOD

The three major topics were treated by separate working groups. Another subgroup was asked to study a specific CODASYL-type DBMS and a relational type DBMS. Furthermore two characteristic engineering problems should be implemented. The experience was distributed among the whole project group.

The selection of the characteristic problems had to be made first, because this influenced the work of other subgroups also. The members of the project group were asked to formulate problems representative for their field of interest. The outcome of this enquiry made clear that a wide variety of engineering activities had to be covered. The DBMS should not only deal with a constant part for storage of standardized parts, design procedures and material properties, but also with structures, machines and installations composed from these elements. Furthermore the DBMS should be used for storage and retrieval of measured and calculated data in such a way that the origin of these data would be maintained. The DBMS should support typical engineering activities such as analysis and modification of complex structures. Various ways of presentation of the database contents must be possible, including graphical presentation of objects and measurement data.

## 3. REQUIREMENTS FOR AN EDBMS

For the definition of the programme of requirements for an engineering database management system it was decided that the starting point had to be the engineers view on the problem. First of all the requirements and design criteria were studied from a number of existing systems. This led to a first formulation of the requirements. Next the participants of the working group were asked to describe their engineering problems for which an EDBMS was thought to be a solution. In consecution these representative problems were confronted with the first draft of the programme of requirements mentioned before. After evaluation the programme of requirements was brought in a final form.

In general an EDBMS should meet at least the requirements for an administrative DBMS. Specific properties directed to engineering processes must then be added. These specific properties are determined by the character of the engineering process, including the whole cycle of problem definition, design, production and testing. Within the four phases indicated a number of characteristic

activities can be distinguished:

- conception
- analysis
- calculation
- decision
- administration
- documentation
- production
- checking
- testing

It must be noticed however that the engineering process contains many loops in which activities with a routine character such as analysis, are followed by activities with a creative character such as selection. The routine work should be done by the computer where the creative part is a human activity. The EDBMS should support this process.

The requirements to be imposed to a EDBMS depend very much on the character of the engineering process. These requirements can be grouped as follows.

### -1. general requirements

A number of general requirements have to be satisfied, such as security, concurrent use possibilities and maintenance.

### -2. user friendliness

Both the engineer end user as well as the application programmer have their own ideas about database systems.

- They are not interested in the internal structure of the system.
- They expect the computer to be fast in jobs they can do fast.
- They want to see something happening after a minimum of input.
- They don't worry about the availability of computer facilities.
- They must be protected against themselves.
- They want to sophisticate their system use as they grow familiar with the system.
- They want to be warned for possible catastrophes as a result of their own action.
- They are allergic for manuals and slow in understanding the meaning of error reports.
- They want to use the system casually with a minimum of preparation.

### -3. data

Which data is to be stored in depends on the function of the engineering processes for which the database is used. In almost every case the ultimate function is a description of an installation consisting of a large set of elements, groups of elements and their mutual relations. Elements and groups of elements have the meaning of the all necessary information varying from a single parameter, via groups of function values to reference to an external file. The relations mentioned have the meaning of defining the functional relationship between the elements. They are essential for the changing configuration during the design process. Without further comments a number of requirements originating from the data is given below.

- elements, groups of elements, mutual relations must be handled
- these data are often fuzzy
- storage and retrieval of criteria and design rules
- the data structure has in principle a dynamic character
- stepwise refinement of structure definition and variables must be possible
- the origin of the data must be administrated
- two types of data: dynamic project data (product database) and static data (constant database) with library structure
- topological and geometrical description of a spatial structure of installation
- separate storage of graphics structures
- coupling between graphics and functional structure.

#### -4. aspects of use

The EDBMS should allow multiple ways of use such as:

- multiple use of data
- user friendly interface with analysis programmes
- very general graphical-alpha numerical report facilities
- interactive use possible (low response times)
- hardware independant.

#### 4. PRACTICAL EXPERIENCE IN APPLYING GENERAL DBMS's

The primary objective of research was answering the question: "what are the possibilities and problems when using administrative DBMS's for engineering purposes?". In order to answer this question the following procedure has been applied.

- inventory of that specific engineering problems in which a DBMS approach might be helpful within the reach of the members of the projectgroup.
- definition of two characteristic problems based on these engineering problems.
- design of a general conceptional scheme covering these two problems.
- implementation of this scheme on a network DBMS, a relational DBMS and also on an 'Integrated Engineering System'.
- evaluation.

The inventory of specific engineering problems has already been described in chapter 2.

##### 4.1. Selection of two characteristic problems

Two problems were selected for being characteristic problems:

- a) channel plate problem
- b) finite element structure

These problems were chosen for the following reasons.

The channel plate problem, which will be discussed in detail later on, is a typical example of an assembly of predefined parts. The parts have different connections with various physical and geometrical properties. Apart from the geometrical position in the assembly, also the logical connections between the parts are essential. The database should contain all information necessary for graphical display and for engineering calculations and storage. The finite element structure was chosen for two reasons. First of all a finite element model has the possibility of processing a large amount of data with a simple structure. Second we have the possibility to work with different levels in the database e.g. a ship structure divided into different sections.

##### 4.2. Design of the general conceptual database

As a first demand it was stated that both characteristic problems should be stored in the same database. For that reason a general conceptual scheme had to be developed to meet this demand. It is to be considered that the scheme cannot result in an optimal result for the individual problems.

Much attention was given to the problem of modeling the hierarchial structure present in all engineering problems. It is common use to structure a design according to the scheme in Fig. 1.

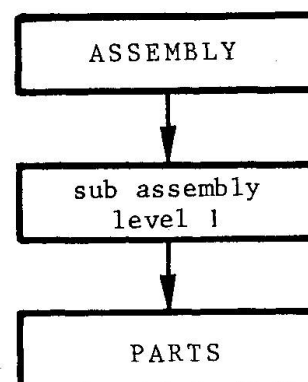


Fig. 1. Hierarchial structure.

The disadvantage of such a model is that the number of hierarchical levels is undefined beforehand. Furthermore a sub-assembly can appear at different levels in assemblies. For this reason an approach was chosen according Fig. 2, which is much like commonly used 'Bill of materials model'.

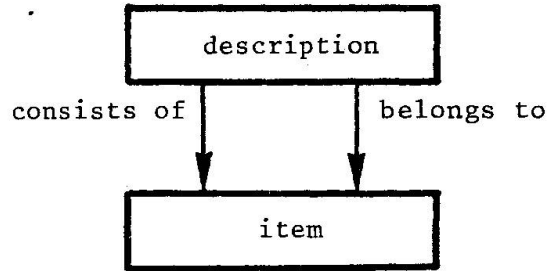


Fig. 2. Alternative model allowing hierarchical structure.

It is shown in the occurrence diagram Fig. 3 how a hierarchical structure can be modelled according to the bill of materials structure of Fig. 2.

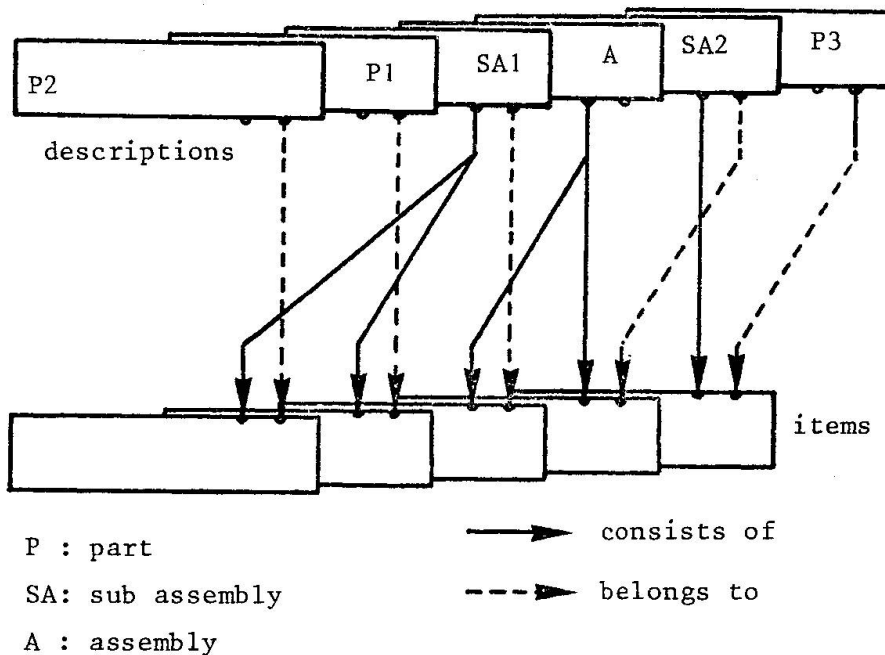


Fig. 3. Occurance diagram of assembly.

In the proposed scheme we have two types of records: the description records and the item records. The description record contains a formal description of an assembly, sub-assembly or part. The item record represents an instance of a sub-assembly or part according to the description it belongs to. The item record contains information about the geometrical position of the item as a part of the (sub) assembly to which it belongs. A sub assembly can thus be regarded as a part of the main assembly. Item records always belong to a single assembly.

The method will be illustrated by the problem of a bicycle. We consider the following assembly of a bicycle, Fig. 4.

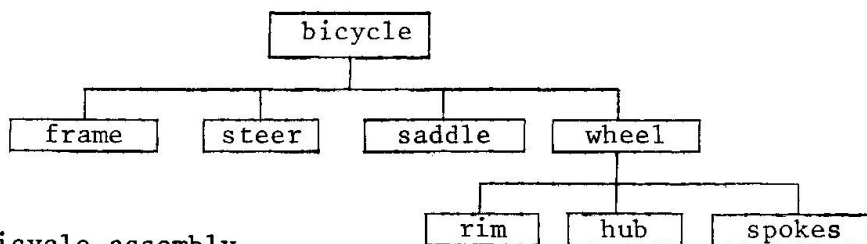


Fig. 4. Bicycle assembly.



In Fig. 5 the model of the bicycle is shown at occurrence level.

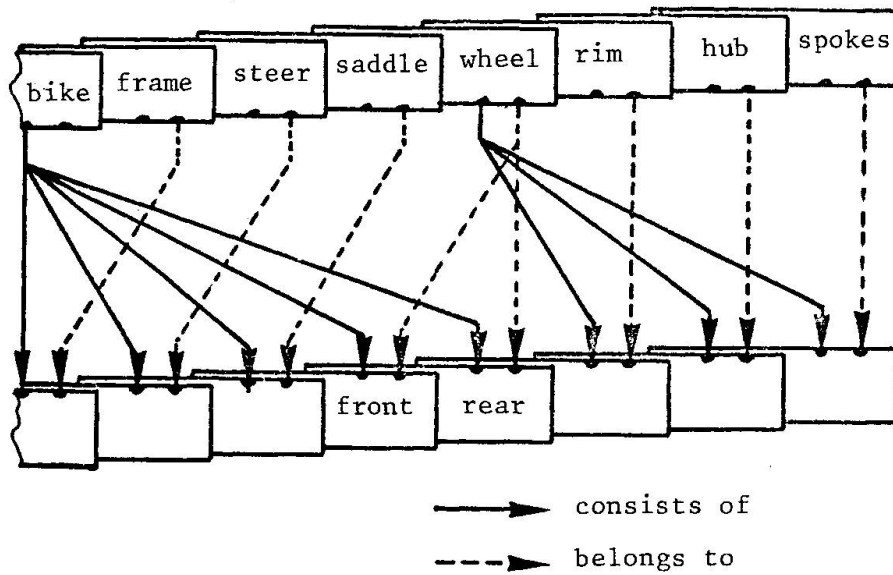


Fig. 5. Occurance diagram of bicycle assembly.

In order to be able to make design calculations the model of a structure or component must contain the necessary physical connections between the parts. This means that for each part connections points have to be defined. To set up the physical connections we need link records between the connection points (Fig. 6).

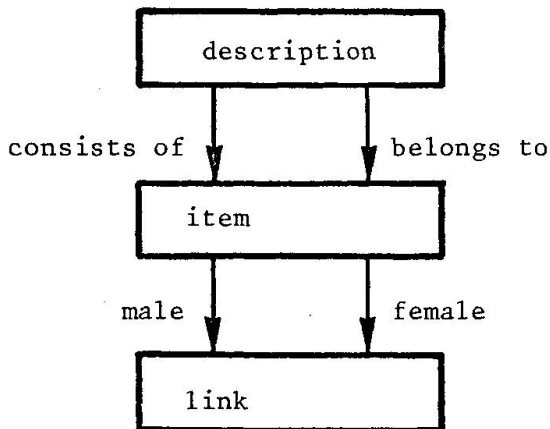


Fig. 6. Extended scheme showing link record.

The link record is related with the two connected items by means of two sets named male and female. This implies that linking of two items has a direction aspect. We did not try to overcome this inconsistency which is typically due to the demands of the database system.

The connection points of each item are assigned an identification. The link record contains the identification of the linked connection points.

In Fig. 7 this process is shown for the example of the bicycle. As an example a link record is shown which describes that connection point # 3 of the frame (item No. 1) is connected with connection point # 1 of the wheel (item No. 4).

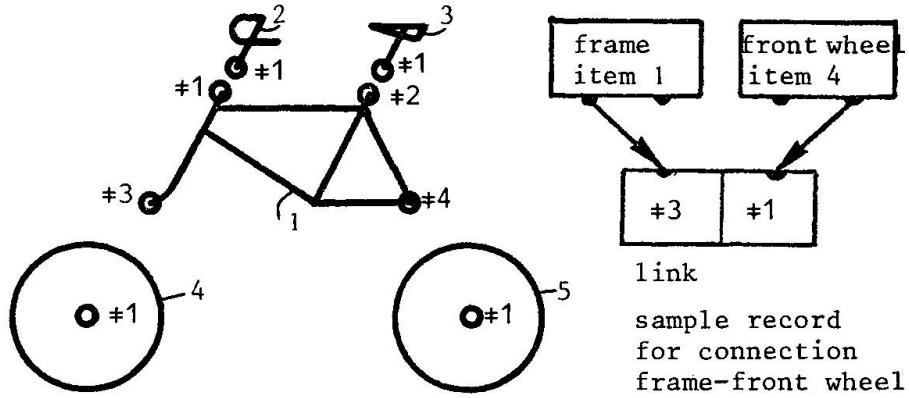
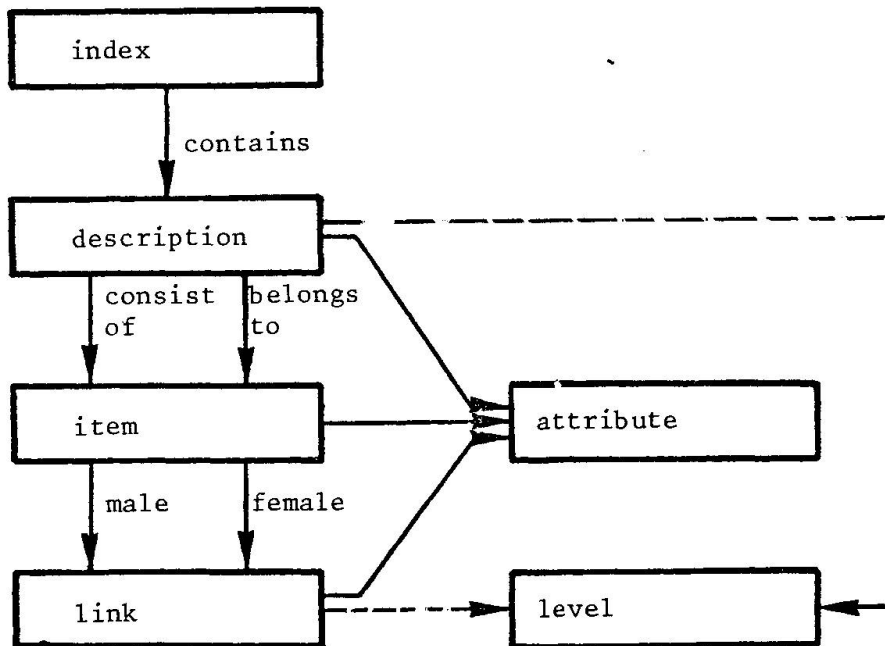


Fig. 7. Physical connection by link record (example).

An extra facility was designed in order to be able to distinguish connections on different hierarchical levels. However this facility was not used in practice.

In order to bring some structure in the heap of description records an index function was used. The index allowed us grouping of selected records. This was used for building a catalogue of standard components and a catalogue of projects like the ship structure and the channel plate designs. The complete scheme as it was used is shown in Fig. 8.







### 4.3. The channel plate problem

#### 4.3.1. Channelplates

In designing logical circuits with pneumatic logical components, one solution is to use so called channel plates. The application of channel plates is similar to the use of printed circuits in electronics.

The channel plate integrates the functions of mounting components to a base plate and interconnection of the components.

When covered with the base plate, the milled grooves in the channel plate form the connections between the connection pins of the logic components. When necessary multiple layers can be used (Fig. 9).

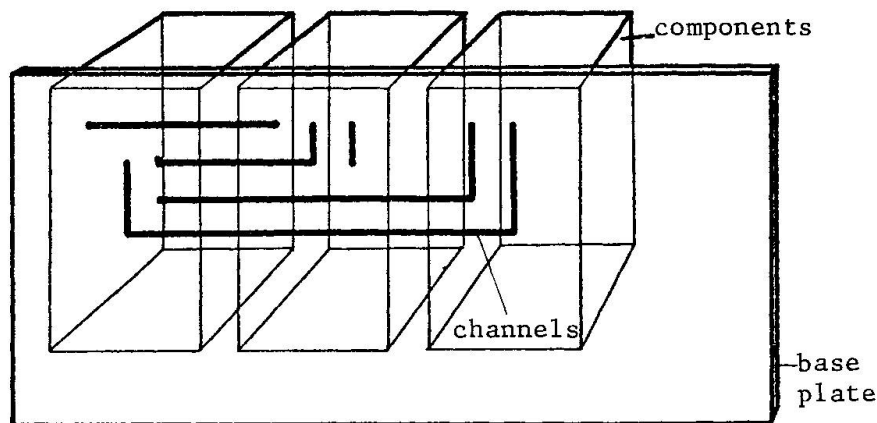


Fig. 9. Channel plate.

Computer Aided Design of the channel layout and of the manufacturing with NC tools has been a subject of study at the Delft University of Technology. The starting point of the design is an existing scheme showing the components used and their interconnections.

An important aspect of the computerized design of channel plates is the method of storage of the data describing the problem. In this report we consider the possibility of storing the data in a data base using the conceptual scheme according to Fig. 8.

The following data have to be considered:

- a catalogue of logic components containing the identification and dimensions of the component and its connection pins,
- a parts list containing the component names and part numbers used in a specific circuit,
- a connection table containing the complete specification of the pins which are to be connected,
- the placing of the parts on the channel plate containing the positions of each component.

4.3.2. The logic component catalogue

For the purpose of channel plate layout design the information shown in Fig. 10 is essential.

component identification	VZ0-3-PK-3		
dimensions	26	68	
pinname and position	A	13.	10.
	R	13.	18.
	P	5.	26.
	Z	21.	26.
	6	5.	42.

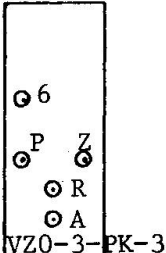


Fig. 10. Component description.

A specific logic element description will be accessed either by its component identification or by the fact that a component is a part of a specific circuit. The catalogue is realized using the indicated part of the conceptual scheme (Fig. 11).

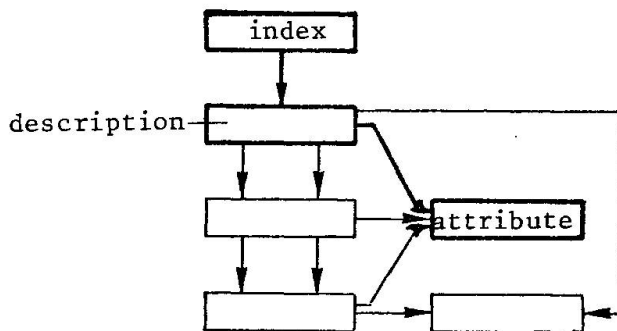


Fig. 11. Catalogue function.

An example of a catalogue containing two logic components is shown in Fig. 12.

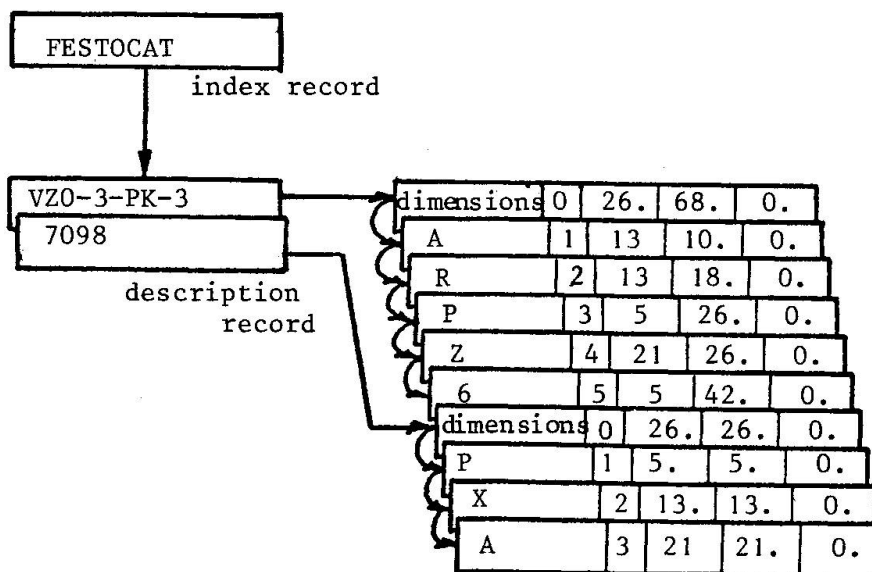


Fig. 12. Logic component data (example).



### 4.3.3. The parts list

It was decided that connections should be treated similar to connecting pipes in a chemical plant.

In the parts list however connections and components should be treated separately. Therefore the part with the description 'channel plate' is composed of two 'parts', the 'component list' and the 'connection list' as shown in Fig. 13.

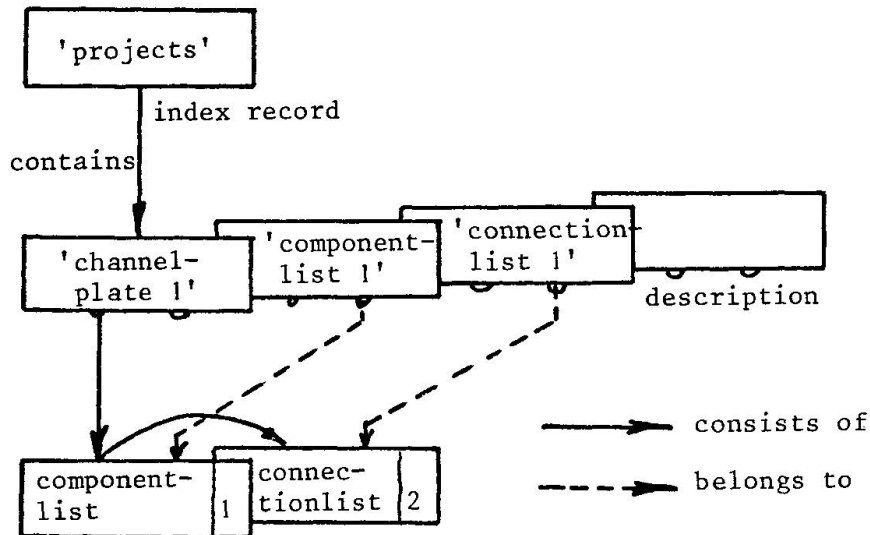


Fig. 13. Channel plate divided into component list and connection list.

Of course the description 'channel plate' is related to the index. The channel plate can be accessed by means of a direct entry on its descriptive name or by looking it up in the index under an entry called 'projects'. It has to be considered that the description records such as componentlist 1 and connection list 1 are strongly related to 'channelplate 1'.

Normally these descriptions will not be used in other channelplates. These records should always be accessed via the record "channelplate".

The logic components used in a specific channel plate are to be selected from the catalogue. An item record is created for them in which the part numbers are stored. The item records are connected with the description record 'component-list 1' and with the selected description record. This process is shown in Fig. 14.

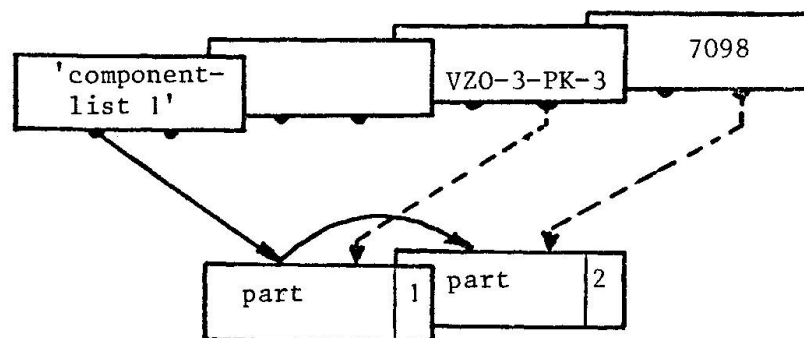


Fig. 14. Creating the componentslist.

During the generation of this part of the database it has to be checked whether a component name given by the designer is really present in the component catalogue.

The placing of the components on the channelplate is reached by the attachment of an attribute record to each part containing the xy-coordinates of that particular part.

For each channel connecting a number of pins a list is given of the pins that are to be connected by that channel, hence a set of duplets (componentname, pinname). Before a channel can be incorporated into the database a check is made whether the connections are possible or not. The componentnames should be present and they must have the specified pin. Then for each channel an item record is stored. Next for each connected pin a connection record is stored and linked to the indicated component (Fig. 15).

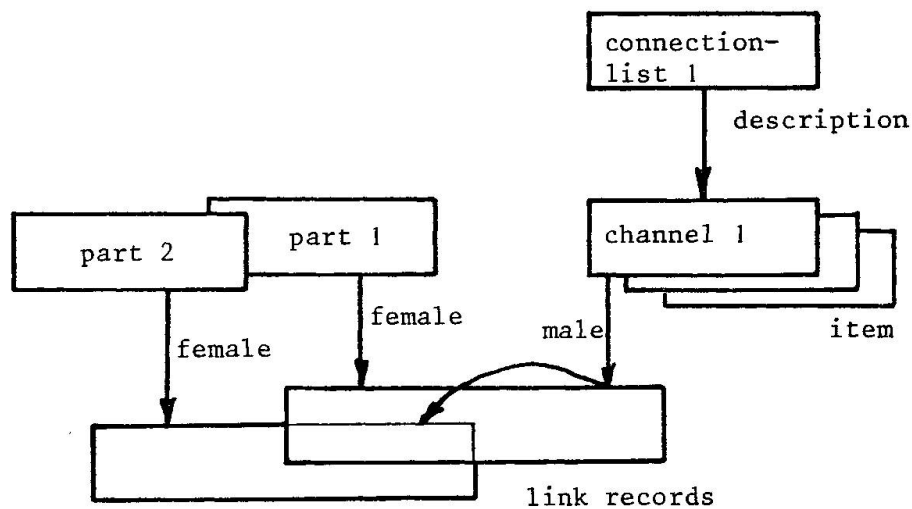


Fig. 15. Storage of connections.

In the connection record the following information is stored:

- a unique number
- the pen name
- the connection number.

Geometrical properties of the channel are stored in an attribute record which is linked to the channel record.

This description does not include some very specific problems related to channelplate problems.

These problems could be solved within the proposed conceptual scheme without many difficulties.

#### 4.4. The finite element problem

##### 4.4.1. Finite element structure

The finite element method is a method to describe the mechanical behaviour of engineering structures.

A structure is divided into a number of finite elements of a simple geometric form. The mechanical properties of the individual elements are known.

Computer programs are available to calculate the response of the structure due to the applied forces. In this context the finite element structure is defined as the complete topological and geometrical description of the structure.



The topology describes the way the structure is divided into elements. This results in a list of the used elements and their relation to a list of nodal points. The geometrical description contains the actual values of the coordinates of the nodal points.

The topology and the geometry are input for various calculations in the design process. The storage of the finite element structure in a database has many advantages. For example the database could be used for modification and control of the configuration.

For large structures it could be necessary to divide the structure into a number of different parts. A ship structure can be split up into different sections and each section is divided into elements. See Fig. 16.

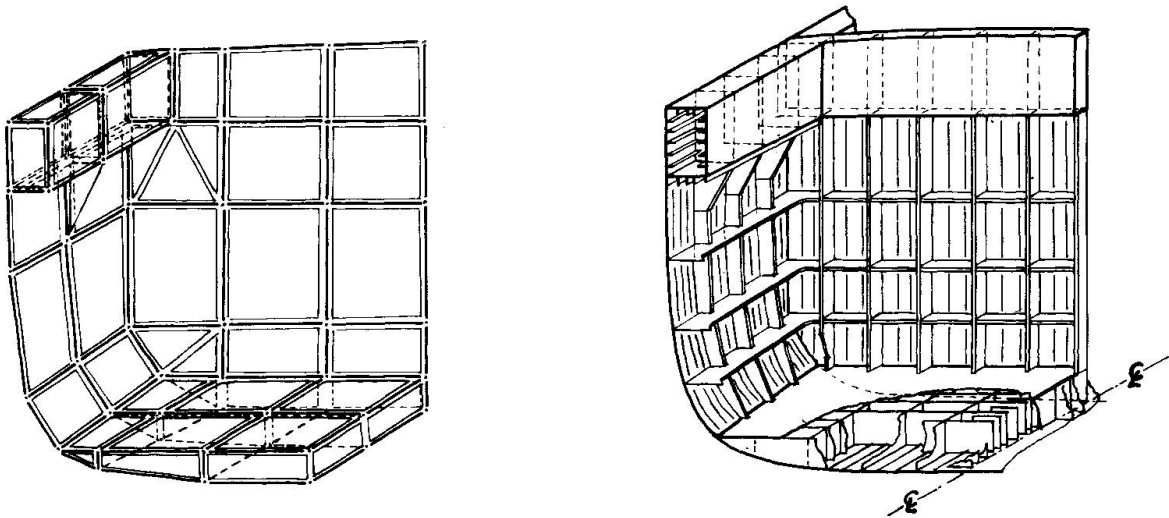


Fig. 16. Ship structure.

In the following paragraphs it will be explained how the topological and geometrical data can be stored in a database according the conceptual scheme.

4.4.2. Implementation of a ship structure into the database

First a description record with the ship's name is stored. Being a project this record is linked to the index record 'projects'. The ship consists of two parts, each being a separate section of the ship. See Fig. 17.

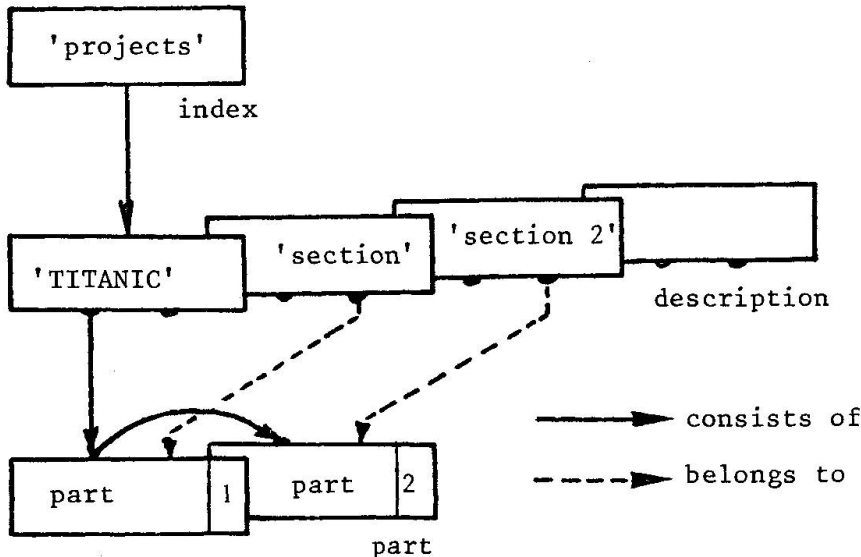


Fig. 17. Ship divided into sections.

Next each section will be treated. To each section the elements and coordinates are assigned in a way similar to the channel plate problem. This results in a list of nodal points and an list of elements for each section. The coordinate values are connected to the nodes as attribute records.

Each element record, containing the element number, is connected to the specific nodes which are part of the element.

The connection is accomplished by means of connection records.

The assembling of two sections has not been implemented. But the conceptual scheme does allow such an assembling.

In both problems it was needed to structure the parts belonging to a description. The way it was implemented, by means of special lists, see Fig. 18, was really cumbersome. Other solutions, requiring a redefinition of the conceptual scheme, are also possible.

The connection of similar parts should be mutual, i.e. the connection from part A to part B should be equivalent to the connection from part B to part A. But a good solution for this problem has not yet been found.

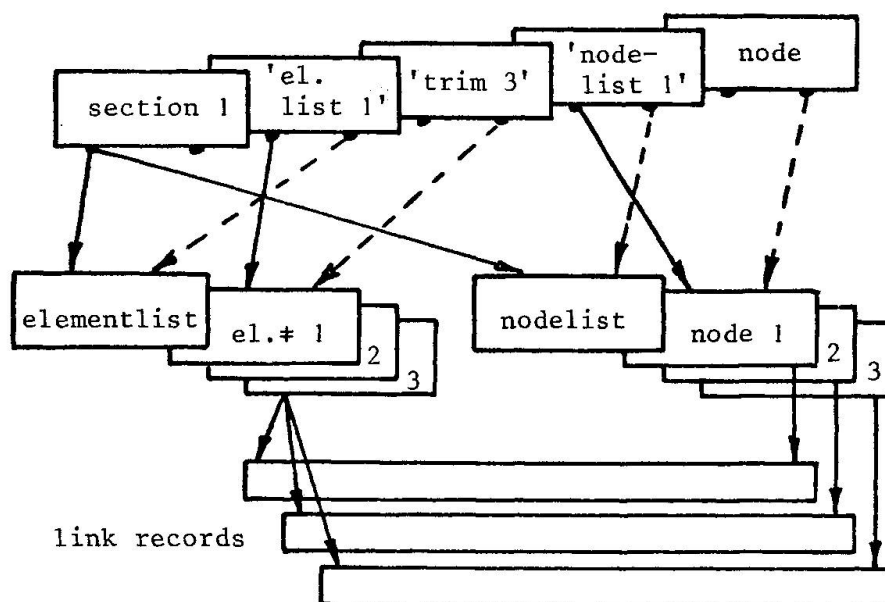


Fig. 18. Finite element division.

#### 4.5. Implementation with DMS 1100

The channel plate problem and the ship finite element model were implemented with the DMS 1100 DBMS installed at a Univac 1100 series computer installed at the Dienst Informatieverwerking Rijkswaterstaat. DMS 1100 is a network type DBMS supporting several host languages and a query language.

First the conceptual scheme was discussed and its final form described in detail. As the conceptual scheme is the basis of the whole project, much attention was paid to this part. With the help of a DMS 1100 specialist the physical implementation of the conceptual scheme was relatively simple. The implementation group existed of four members including a DMS 1100 specialist.

The facilities like automatic rollback, back up etc. were found to be very useful.

The performance of the system is difficult to measure. With a size of approximate 200 interconnected elements for each problem, a typical DMS 1100 retrieval of all elements took a time of 5-10 seconds.

The more simple FORTRAN solution with a sequential and a direct access file took about twice the amount of time.



It is felt that in many engineering applications this performance will be acceptable.

The whole implementation including the channel plate problem and the finite element model into DMS 1100 took about 200 hours.

Our experience with the implementation led to the following remarks:

- If the conceptual scheme is well chosen, there is no need for intermediate changes of the conceptual scheme.
- It is important to choose a conceptual scheme that anticipates to future complexity, because most network DBMS'es do not allow a dynamic change of the implemented scheme.
- The programmes for storage and retrieval are quite complicated. In a production environment it is recommended to develop standard programs.

To illustrate the last omission the programme part for the input of the channels will be discussed. It will be assumed that the individual elements have been checked and stored. It is of great importance that the database is always consistent and that it does not contain erroneous or incomplete information. First the database is coupled to the user programme. When the channelplate in question is found via the INDEX, an ITEM record 'channellist' is stored and automatically linked.

An unique description record 'chanlis 1' is stored as the head of the channels-list and a connection is made with the 'channellist', as follows:

```

FETCH      SYSTEM
FETCH      INDEX      'projects'
FETCH      DESCRIPTION 'channelplate'
STORE      ITEM       'channellist'
STORE      DESCRIPTION 'chanlis 1'
FETCH      ITEM       CURRENT, RETAIN
CONNECT    ITEM       SET = USES

```

Attention is asked for first the aspect of "navigating" through the database, and second the currency problem. For each record or link type there is always one specific record or link current in use. Most activities applied to the database change this currency. The CONNECT command above establishes a link between the current DESCRIPTION and ITEM records. Before the CONNECT command is issued the intended ITEM record is fetched, while retaining the current DESCRIPTION record. A single channel is then read as a set {(itemnumber, pinname)}. Before the channel is stored in the database it is checked whether the itemnumber appears in the partslist of the channelplate and whether that component has indeed a pinname as required. Furthermore it is to be checked that this pin has not been used before. Such a check does require some navigation through the database.

```

FETCH      DESCRIPTION 'elementslist'
FETCH      ITEM       itemnumber, if present then
FETCH      OWNER OF SET = USES (establishes DESCRIPTION record)
FETCH      ATTRIBUTE  pinname, via set = omscatter; if pin exists then
FETCH      CONNECTION pinname, via set = female. Not found means unused pin.

```

When all these conditions are satisfied, the data can be stored.

#### 4.6. Implementation with ORACLE

ORACLE is a typical relational DBMS. At the time that this paper was written the ORACLE implementation was not finished yet. A lot of thinking work that had been done for the DMS-1100 implementation was also useful for the relational approach. The general scheme could easily be translated to a set of relations as required for ORACLE. This resulted in the following relations.

```

INDEXENTRIES { ENTRY },
DESCRIPTION { DESCRIPTIONNAME },
INDEX { ENTRY, DESCRIPTIONNAME },
ITEM { DESCRIPTIONNAME 1, DESCRIPTIONNAME 2, ITEMNO },
NODE { DESCRIPTIONNAME, NODENO, NODETYPE, x, y, z },
CONNECTION { DESCRIPTIONNAME, NODENO 1, ITEMNO, NODENO 2 },
DESCRIPTION ATTRIBUTE { DESCRIPTIONNAME, ATTRNM, ATTRNO, NVALUE, CVALUE }
ITEM ATTRIBUTE { DESCRIPTIONNAME, ITEMNO, ATTRNO, NVALUE, CVALUE }.

```

The cursive attributes form a key for the tuples. It is noticed that the system of defining connections between items has been changed with respect to the original approach. For an assembly a list of nodes is defined to which nodes, internal to the items, are attached by means of the connection relation. For this reason the connection relation has two attributes NODENO, the first node indicating the global node number, the second the local node number of the item to be connected.

As a result of our first experience the preliminary conclusion can be drawn that unless special precautions are made it is difficult to maintain the integrity of the database.

#### 4.7. Implementation with the ELPRO system

ELPRO is an acronym for Engineering- and List PROcessing system. ELPRO was developed by the Dutch HBG Company based on Becqué's thesis \*). ELPRO is supported by P & P Engineering Consultants B.V., Rotterdam.

Characteristics:

- System is written in FORTRAN with user callable subroutines.
- System and user data are stored in tables with a row and column structure. Information is stored and retrieved directly from the tables by user written programmes.
- ELPRO supports its own input and definition language for automatic generation of data.
- ELPRO is an open system in the sense that the user can define his own input and command language.
- ELPRO has a possibility to define relations between data and use these for storage and retrieval (Relational approach).
- The command language allows the use of user defined procedures including possibilities for jumps and loops.
- ELPRO has its own report writer, a query language for system maintenance and a logging facility.
- ELPRO is multi user and can be implemented on small computers.

#### Implementation of the finite element structure in ELPRO

In ELPRO each data item has to be collected and stored in twodimensional tables. To avoid fixed pointers every table often has to contain extra data. For the ship structure, a language was defined to read directly the datacards produced by the user. This resulted in the tables as shown in fig. 19.

---

\*) Becqué: LPR, an Integrated program and Data Management System for Engineering Applications (Thesis Delft, 1977).



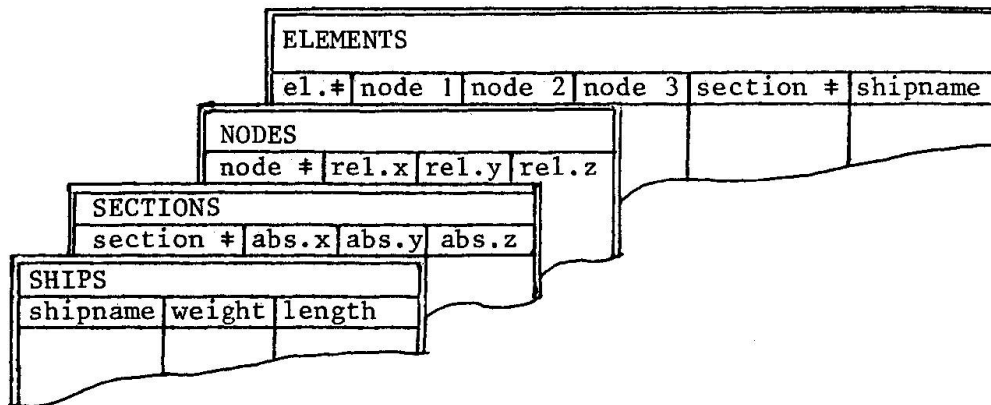


Fig. 19. Tables and relations for ship structure.

The implementation takes four steps.

- Table definition

Every item is given a name, sequence number, type, default value and a range. Tables are defined using the ELPRO definition language. No user programmes are necessary at this stage. The user uses his own problem oriented language. Data generation tools are available.

- Data entry

The tables are filled using the ELPRO input-programme. Each input record will be checked according to the table definition. At this stage the reportwriter is used to print the contents of the tables for checking purposes. Some changes in the definition or in the contents of the tables can be made.

- User programmes

With user written FORTRAN programmes communication with the database is possible via ELPRO Library subroutines.

- Running the test programme

The user can activate his programme by calling the module-command in the command table. The results are stored in work tables for examination or they are printed directly. Before running modifications can be made.

Implementation of the channelplate problem in ELPRO

In the same way as it was done with the ship, tables were defined and filled. Some minor changes of the data were necessary before they could be read, checked and stored in the tables.

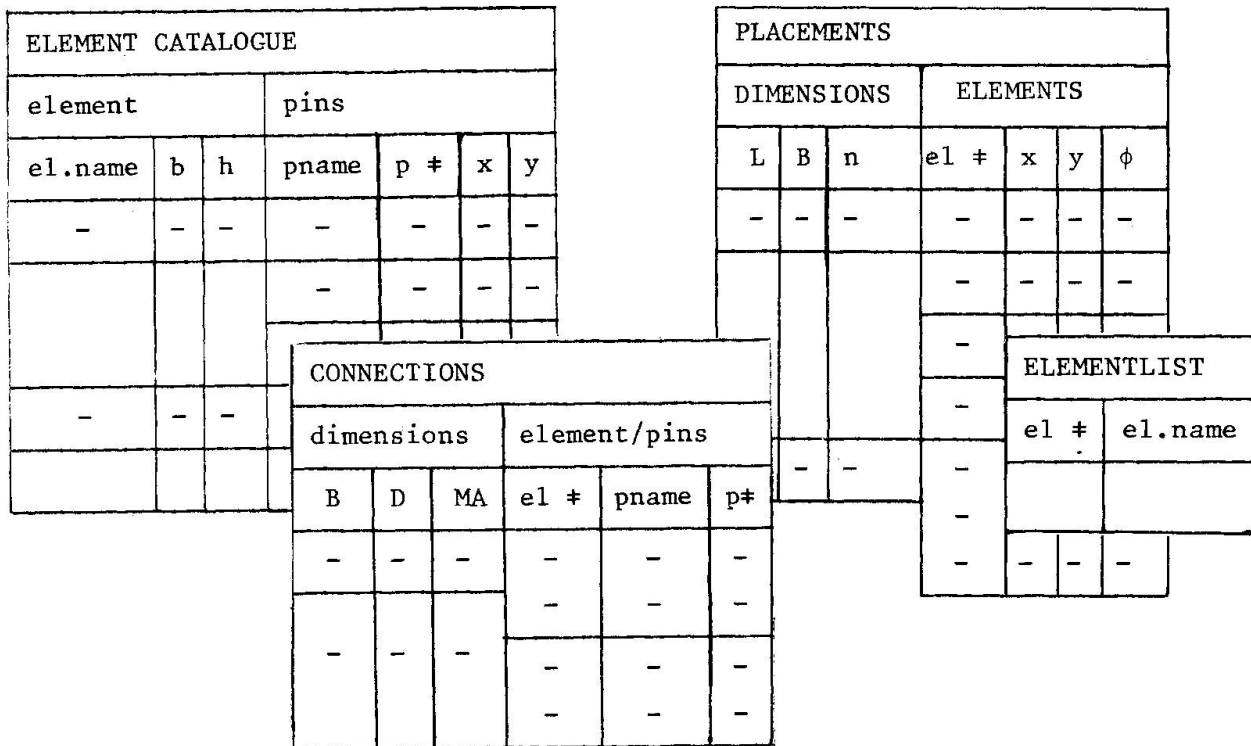


Fig. 20. Tables and relations for channel plate.

As shown in the figure 3 tables had to be divided into two parts. In the first part some general information is stored (overall dimensions) while in the second part more detailed information is given. The two parts have different definition rules (subtables). This division was necessary to read the user delivered information rows in an easy way. Other solutions are also possible. The implementation of the applications in ELPRO is much more simple compared with a CODASYL-type database. ELPRO was developed for an engineering environment. Therefore a number of facilities such as crash protection and integrity are less sophisticated. Execution times of ELPRO are better than with the CODASYL-type database.

## 5. CONCLUSION

Due to the fact that the project has not been finished yet it was not possible to give an integral result. A final report is expected to be available at the end of 1982 from the CIAD institute.

## 6. ACKNOWLEDGEMENTS

The preparation and presentation of this work was made possible by CIAD.

Leere Seite  
Blank page  
Page vide