Zeitschrift: IABSE reports of the working commissions = Rapports des commissions de travail AIPC = IVBH Berichte der Arbeitskommissionen

Band: 31 (1978)

Artikel: "BC: bridge construction" a case history

Autor: Dutertre, J.C. / Ayral, A.

DOI: https://doi.org/10.5169/seals-24929

IABSE
AIPC
IVBH

COLLOQUIUM on:
"INTERFACE BETWEEN COMPUTING AND DESIGN IN STRUCTURAL ENGINEERING"
August 30, 31 - September 1, 1978 - ISMES - BERGAMO (ITALY)

"BC: Bridge Construction" a Case History
"BC: Construction de ponts": une étude de cas
"BC: Bau einer Brücke" ein Studienfall

J.C. DUTERTRE
PhD., Manager Elect. Data Process.
Europe Etudes Gecti
Clichy, France

A.J. AYRAL
System Analyst
Europe Etudes Gecti
Clichy, France

Summary
The general methodology followed to develop a large program is presented. This methodology is aimed at standardizing the communication between the engineers and the program. It is based on a simple in-house package which facilitates the programmer' s work. A high quality documentation, yet limited to the minimum, is produced.

Résumé
Cet exposé a pour but de présenter la méthodologie générale suivie lors du développement d' un grand programme. Cette méthodologie est basée sur la standardisation de la communication entre l' ingénieur et le programme. Elle est basée sur un package simple-maison qui facilite le travail du programmeur. La documentation produite est limitée au minimum mais elle est de bonne qualité.

Zusammenfassung
Die generelle Methodenlehre wird für die Entwicklung eines grossen Computerprogramms dargestellt. Diese Methode soll die Kommunikation zwischen dem Ingenieur und dem Programm vereinfachen. Sie ist auf einem eigenen "Package" begründet und erleichtert die Arbeit des Programmierers. Die Dokumentation ist aufs Minimum beschränkt, aber doch von hoher Qualität.

1. GENERAL

1.1. The Firm

Europe Etudes Gecti is a Civil Engineering consulting firm which essential specialization is prestressed concrete. It employs approximatly 300 persons in France, scattered in nine locations. It has five subsidiaries in different countries.

EEG has been involved in the design of many outstanding prestressed concrete structures (offshore platforms, Montreal olympics....) and is currently involved in the design of the Key West bridges in the US and of F9 freeway in Australia.

1.2. Data Processing

The duty of the EDP (electronic data processing) team is to make programs available to the engineers. The EDP team doesn't use the programs itself but for testing purposes. The programs results are placed under the final total responsibility of the engineers.

The service provides user's manuals which are the communication tool between the machine and the users. Training sessions are also organized.

Programs are only developped when they are not available somewhere else. This implies that the service has access to the most important computers available. This is done with the use of CADUCEE which is a high quality data processing network.

2. THE CASE

A simulation program for the verification of prestressed concrete linear bridges had to be developped. In house, programs were already available but did not meet new requirements. The problem related with this type of bridges is that each individual construction phase must be checked. A three span bridge for example may require up to one hundred computations, each done with a structure slightly different from the previous one.

The existing "chain" of programs was very limited both in the methods of construction it could modelized and in the mathematics it was based on. In addition these old programs were very difficult to use by the engineers and even more difficult to either maintain or modify.

Beside the strength of materials and mathematical requirements, which are not the topics of this conference, the objectives were :

- A verification program which could be used for design.
- An engineering oriented program
  - . user's oriented input language
  - . high quality output
- Stand alone user's manual
- An international target
  - . easily adaptable to any bridge regulation
  - . an output translatable into different languages
- A good quality in house documentation.
- A program which has to run on both IBM and CDC.

The final product is a program called "BC:BRIDGE CONSTRUCTION".

The methodology presented in this paper is the result of an evolution which started over four years ago. The large project represented by the building of BC enabled to standardize the approach.

3. THE METHODOLOGY FOLLOWED

The final objective is to obtain a complete program which satisfies the needs of the users.
A program can be considered as complete when :

- it can be used without a direct contact between the engineer and the programmer
- documentation is provided to satisfy
  . the client who is going to look at the output
  . the user
  . the programmer who will have to maintain the coding
  . the technical management who is going to request future changes

The final documentation is standard at EEG. It is illustrated on figure 1.
It contains :

- The USER'S MANUAL which is itself split into two basic levels
  . the general and client oriented level (Presentation, Hypothesis and conventions, presentation of results). This part can be referred to as the functionnal specification of the program.
  . the user's level (Data input, methodology, running the program, tests and examples, error messages).

- The EXTERNAL SPECIFICATION which is a complete desciption of the program and associated information. It contains :
  . A description of the overall program structure
  . The mathematics on which the program is based
  . The data base definition (either in core or on disk)
  . The functional description of each module and the detailed algorithm when necessary
  . The graphs representing the user's oriented input language.

- The INTERNAL DOCUMENTATION which is directly included in the FORTRAN coding.
This list contains from 30 to 50% of comments among the final 70000 cards of BC.

- The TESTS DOCUMENTATION where the main tests are logged together with their results.

3.1. The design phase

The programming of an application like BC is a permanent iteration process which converges towards finer and finer documentation. It is basically a topdown analysis which links with some existing utility modules or sets of modules.

The initial request usually comes either from the technical Management or from the Data processing itself.

The user's manual is used to express the functional requirements. It is written by the EDP team. Only a skeleton of the manual is first written, it will gradually "grow" into the final version.

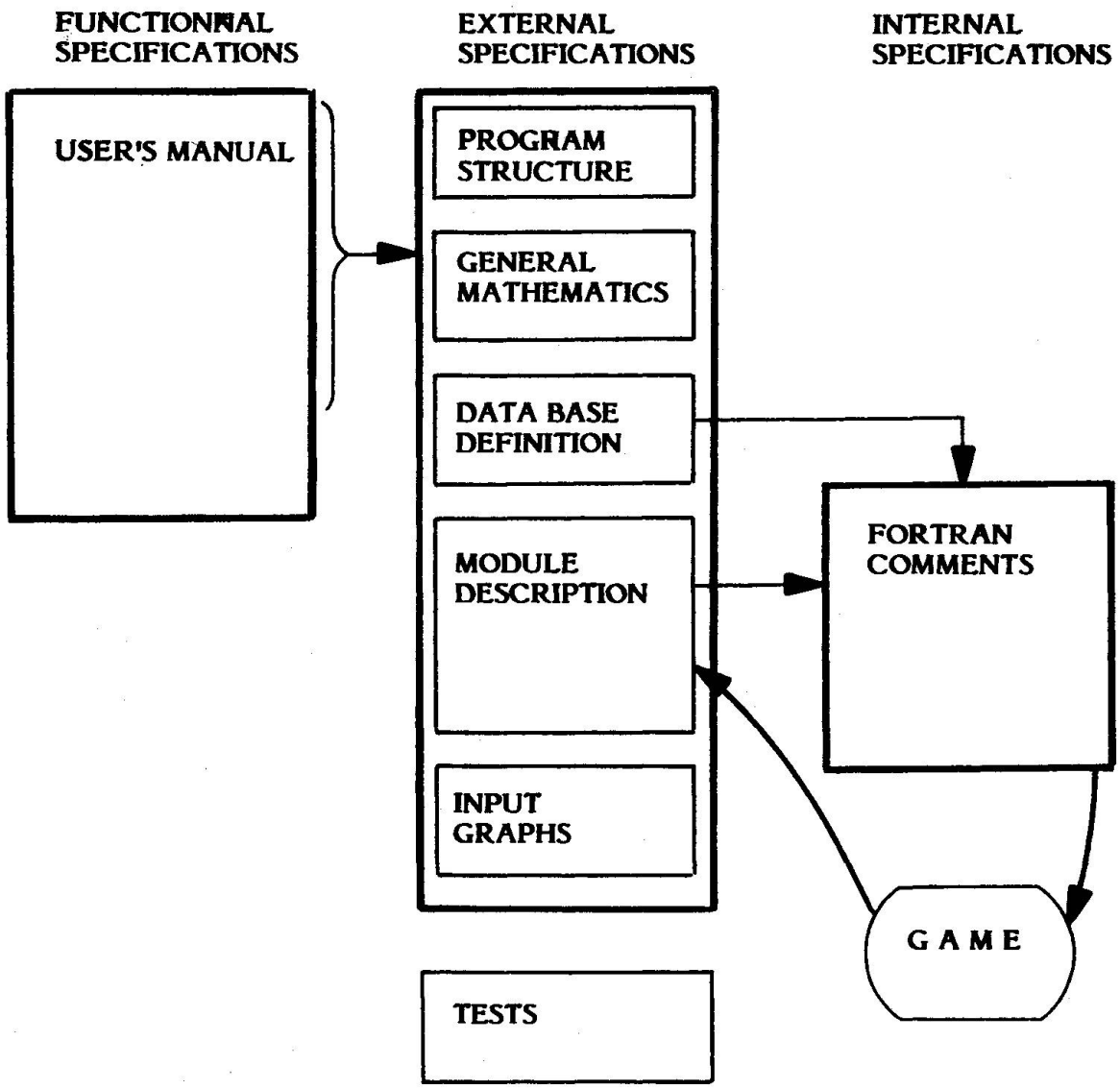


Figure 1

Structure of the Documentation

The initial manual only contents the following section :

- presentation
- hypotheses and conventions
- presentation of results
- data input (very brief)

After a few iterations between technical management and EDP team, the external specificationsstart growing, beginning with the program structure.
As soon as possible cost estimates are made and "Go ahead" or "kill" decisions are taken.

3.2. The GO AHEAD phase

Once the GO AHEAD has been granted, the external specifications are detailed

- the data base is defined
- the graphs of the user's oriented language are established
- the modules are described

Because a top down approach is followed, all the modules need not to be defined before programming begins.

3.3. The Programming phase

The programming is done in a very standard FORTRAN IV as the program must be able to run on both IBM and CDC machines.

Structured "egoless" programming is used. It is based on very strict written rules which were discussed by the team before being applied. Figure 2 shows an example of the coding produced.

The basic rules which are followed can be summarized as :

- it must be possible to follow the basic logic of a module by simply reading the comments
- the text is indented as requested by branches, computed GOTO's or DO LLOP's.
- the statement numbers normally contain five digits
    . two for the chapter number
    . two for the section number
    . one for the part number. A zero part number represents a "clean" instruction and a non zero one an instruction which is implied in a surrounding flow.
- variable names are formed by using two letter codes common to the entire program.

The functional description of each module is placed as comments at the very beginning of the list following the "SUBROUTINE" or "FUNCTION" statement.

One of the difficulty found is to keep the initial functional hand written description of the module up to date while some functions are slightly modified. To overcome this problem, it was decided to throw away the initial hand written functional description and replace it by that contained in the FORTRAN source. This new functional description is obtained automatically by a specific program.

```
      SUBROUTINE KPRT1
C
C PURPOSE
C -------
C     PRINT A PHASE RESULT
C
C INPUT
C -----
C     PRINTING CODES FOR EACH OF THE LOADCASES
C     STARTING AT ADDRESS  IDPRT WITH NBPRT  VALUES
C     FOR EACH LOADCASE WHICH ARE IN ORDER
C              1 !≡ DEAD LOAD
C              2 !≡ SURCHARGES
C              3 !≡ TOTAL OF ALL LOADS
C              .
C              .
C        NBLDPR !≡ PRESTRESSING TOTAL
C        NBLDMX !≡ MAXIMUM NUMBER OF LOADINGS IN R.F.
C
C     PRINTING CODES ARE IN ORDER
C              0 !≡ 1 SOMETHING TO PRINT
C                   0 NOTHING TO PRINT
C              1 !≡ STATE OF THE STRUCTURE
C              2 !≡ DISPLACEMENT OF THE NODES
C              3 !≡ REACTIONS AND ELASTIC SUPPORTS
C              4 !≡ INTERNAL FORCES


C
02024            CONTINUE
C                **** THIS COORDINATE IS SLAVED
                 NDI=NRK
C                .... IS I NODE ACTIVE ?
                 IDNDI=IADNO(NDI)
                 IF(IDNDI)02025,02025,02026
C
02025                 CONTINUE
C                     .... I NODE IS NOT ACTIVE
C                     .... NUMBER THIS COORDINATE WITH K
                      NBCO=NBCO+1
                      IBUF(IDNRK+3)=NBCO
                      GOTO 02020
C
02026                 CONTINUE
C                     .... NODE I IS ACTIVE AND WILL BE NUMBERED
C                     .... AT THE END OF THE LOOP
                      IWAIT=IWAIT+1
                      CALL GETVAR(IDW,2)
                      IBUF(IDW)=IDNRK+3
                      IBUF(IDW+1)=IDNDI+KOORP1+3
02020       CONTINUE
02000  CONTINUE
C
C      **** FINAL LOOP TO NUMBER THE LEFT OVER SLAVE COORDINATES
```

Figure 2

Egoless FORTRAN

3.4. GAME : An in house package

One of the problem of communication between engineers and the programs is that of the standardization of both the input and the user's manual. Using a new program requires a time for discovery.

This discovery goes through the following phases :
- global look at the manual
- detail look at the hypothesis
- understanding of the input
- trying to use the program

It is highly important to ease the task of the engineer by simplifying this discovery. This can be achieved if :
- all user's manual have the same structure
- all user's manuals have the same presentation
- all programs use the same user oriented input language

This is one of the goal of the in-house package GAME (Graph - Analyzer - Memory - Editor).

GAME comprizes four main functions (see figure 3) :
- a syntaxic language analyzer
- an output formater and an editor
- a memory manager
- a specification regenerator

These basic functions can be found in large systems like GENESYS or ICES. However GAME is a very basic simple tool, which runs on IBM and CDC, and which does not require a complex environment.

3.4.1. The syntaxic language analyzer.

This analyzer is made of two parts

- a table builder program which takes the programmer's description of the requested input language to build the necessary FORTRAN tables. These tables are included in the program being developped

- a series of routines which analyze user's input data (see figure 4) according to the above tables and returns to the programmer the decoded informations. A series of basic functions are also included in the language and are therefore common to all programs. (editing functions, setting program parameter functions).

3.4.2. The output formater and editor.

This is a tool to the programmer. It consists of two parts :

- a program which takes as input all the formats containing some alphabetical informations to create a format file.
- a series of routines which are called instead of the standard write. Figure 5 shows a typical output together with the summary of the job which is automatically generated.

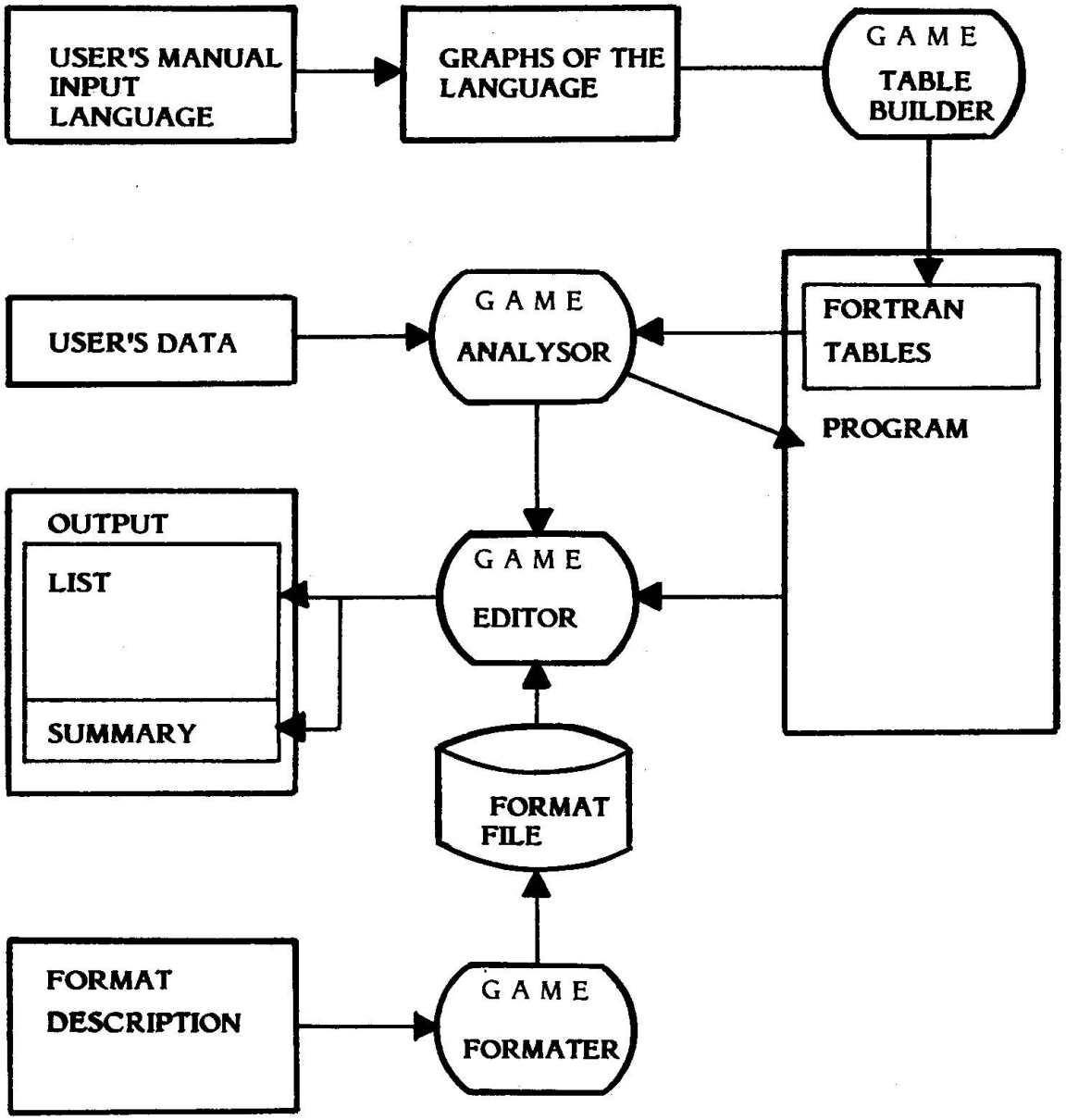


Figure 3
Functions of the package "GAME"

```
EUROPE - ETUDES                    PROGRAMME  BC                          APPENDIX -B-
CONTRACT  26203.00        A 35 OA 203
06/15/78                                                                  PAGE     1

  1 ...  PARA 84  0 / 88 2 / 27 3 / 163 2
  2 ...  CONTRACT 26203 01 / A 35 OA 203
  3 ...  TITLE EEG STRASBOURG A 35 OA 203 PONT SUR LE CANAL DU RHONE AU RHIN
  4 ...  TITLE DEFINITION S1
  5 ...  REPORT
  6 ...  DEFINITION
  7 ...  MATERIAL
  8 ...  CONCRETE 10/ BETON STANDARD
  9 ...  STEEL 1 / 8 T 15 CLASSE 3 TBR POUR CABLES D ENCORBELLEMENT
 10 ...     AREA 0.001112
 11 ...     GUTS 184900 JACK 157000
 12 ...     FRICTION 0.17 WOBBLE 0.0016 DRAW 0.005
 13 ...     RELAXATION INITIAL .55 10.3 .8 13.1
 14 ...  STEEL 2 / 8 T 15 CLASSE 3 TBR POUR CABLES DE CONTINUITE
 15 ...     AREA 0.001112
 16 ...    GUTS 184900 JACK 157000
 17 ...     FRICTION 0.17 WOBBLE 0.0016 DRAW 0.005
 18 ...     RELAXATION INITIAL .55 17.7 .8 18.2
 19 ...     RETURN
 20 ...  NODES 1 2 / 0 .50 -0.681          -.681
 21 ...     3 THRU 10 / 4.27 THRU 30.52 STEP 3.75 -0.681 -0.684 -0.697 -0.727
 22 ...  * -0.785 -0.882 -1.028 -1.240 / 11 / 37.52 -1.849
 23 ...    111 / 37.52 -1.849 / 211 / 37.52 -8.43
 24 ...     12 THRU 19 / 44.52 THRU 70.77 STEP 3.75 -1.236 -1.024 -0.876 -0.776
 25 ...  * -0.716 -0.682 -0.666 -0.663 / 20 /73.97 -0.661 /21 / 76.11 -0.661
 26 ...    22 THRU 29 /79.31 THRU 105.56 STEP 3.75    -0.663 -0.666 -0.682
 27 ...  * -0.716 -0.776 -0.876 -1.024 -1.236 / 30 / 112.56 -1.849
 28 ...    130 /112.56 -1.849 / 230 /112.56 -8.43
 29 ...    31 THRU 38 / 119.56 THRU 145.81 STEP 3.75 -1.24 -1.028 -0.882
 30 ...  * -0.785 -0.727 -0.697 -0.684 -0.681 /39/149.58 -0.681
 31 ...     40 /150.08 -0.681
 32 ...    SUPPORT V 1 40 H V R 211 230 RESTRAINT H V R 11 111 30 130
 33 ...     RETURN
 34 ...  SECTIONS
 35 ...     1 IB 19.341 AREA 8.476 C 1.849 CP 2.101 QI 5.89  B 1 RESAL PARTIEL
```

Figure 4

Example of user's input data

```
EUROPE - ETUDES                    PROGRAMME  BC                VERSION 1.3B
CONTRACT  26203.00        A 35 0A 203
06/15/78                                                        PAGE     3

          INPUT  DATA  -GENERAL-              (CONTINUED)
          ----------------------

          MATERIAL PROPERTIES   (CONTINUED)

          STEEL     1  I   8 T 15 CLASSE 3 TBR

                RELAXATION COEFFICIENTS

                RELATION RELAXATION/INIT.STRESS

                    INIT.STRESS              RELAXATION
                (PERCENT OF GUTS)    (PERCENT OF INIT.STRESS)

                        .550                      10.3
                        .800                      13.1
```

```
EUROPE - ETUDES                    PROGRAMME  BC              APPENDIX -A-
CONTRACT  26203.00        A 35 0A 203
06/15/78                                                        PAGE     1

          ---------------  S U M M A R Y  ---------------

     INPUT  DATA  -GENERAL-                                          1
     ------------------------------

          MATERIAL PROPERTIES                                        2
          COORDINATES OF THE NODES                                   5
          DATA OF SUPPORTS                                           6
          DATA OF INTERNAL RESTRAINTS                                6
          CROSS-SECTION GEOMETRIC PROPERTIES                         7
          CROSS-SECTION TEMPERATURE PROPERTIES                       7
          DATA OF ELEMENTS                                           8
          DATA OF SCAFFOLDS                                         10
          DATA OF UNIFORM LOADS                                     10
          DATA OF PRESTRESSING                                      11
                 STANDARD EXTREMITIES                               11
                 EXTR NB.    3                                      11
                 EXTR NB.    4                                      11
                 EXTR NB.    5                                      11
                 EXTR NB.    6                                      11
```

Figure 5

Typical output and summary

3.4.3. The memory manager . (see figure 6)

This is a set of simple subroutines which manages the in core memory. It prevents using the cumbersome FORTRAN rigid DIMENSION statement.

In addition, a standard memory organization enables the user to access program parameters at any time during his natural flow of data.

For example :
Normal flow of data
"CONSTRUCTION PHASE SEGMENT ASSEMBLE ALL"

Setting parameter 200 to 1 before assembling :
"CONSTRUCTION PHASE SEGMENT PARA 200 1 ASSEMBLE ALL"

Because this specific data input is handled directly by the package, it is common to all programs. Here again, discovery is made easier to the engineer.

3.4.4. The specification re-generator.

This is a very simple module which extracts and prints comments from a FORTRAN list. It does it with two options :

- prints the functional description comments only (i,e the comments place at the beginning of a module)

- prints the above comments plus all those found in the coding

The output from this program replaces the hand written functional specifications used initially. The new specifications are therefore always up to date as the programmer easily accepts to update the comments in the FORTRAN list at the same time as he modifies the coding.

4. PROGRAMMING THE REGULATIONS

One of the difficulty when programming traffic regulations for bridges is that they are subject to change and that they are different from one country to the other. This is always a source of problems to a programming team, more so when the user is thousands of kilometers away.

The decision was taken to implement in BC a user oriented traffic regulation language. It enables the user to describe the logic he requires. He does so using elementary orders which deal with variables and operators located at a very high level (trucks, influence lines, combinations, searches....).

Standard traffic regulations (AASHTØ, CPC) are available by default but they can either be modified or completely changed upon data input.

Just to give an idea, only approximately 70 statements are sufficient to represent the AASHTØ requirements.

This gives the user a possibility to gain confidence into the program in an area which is usually only accessed by the programmer.

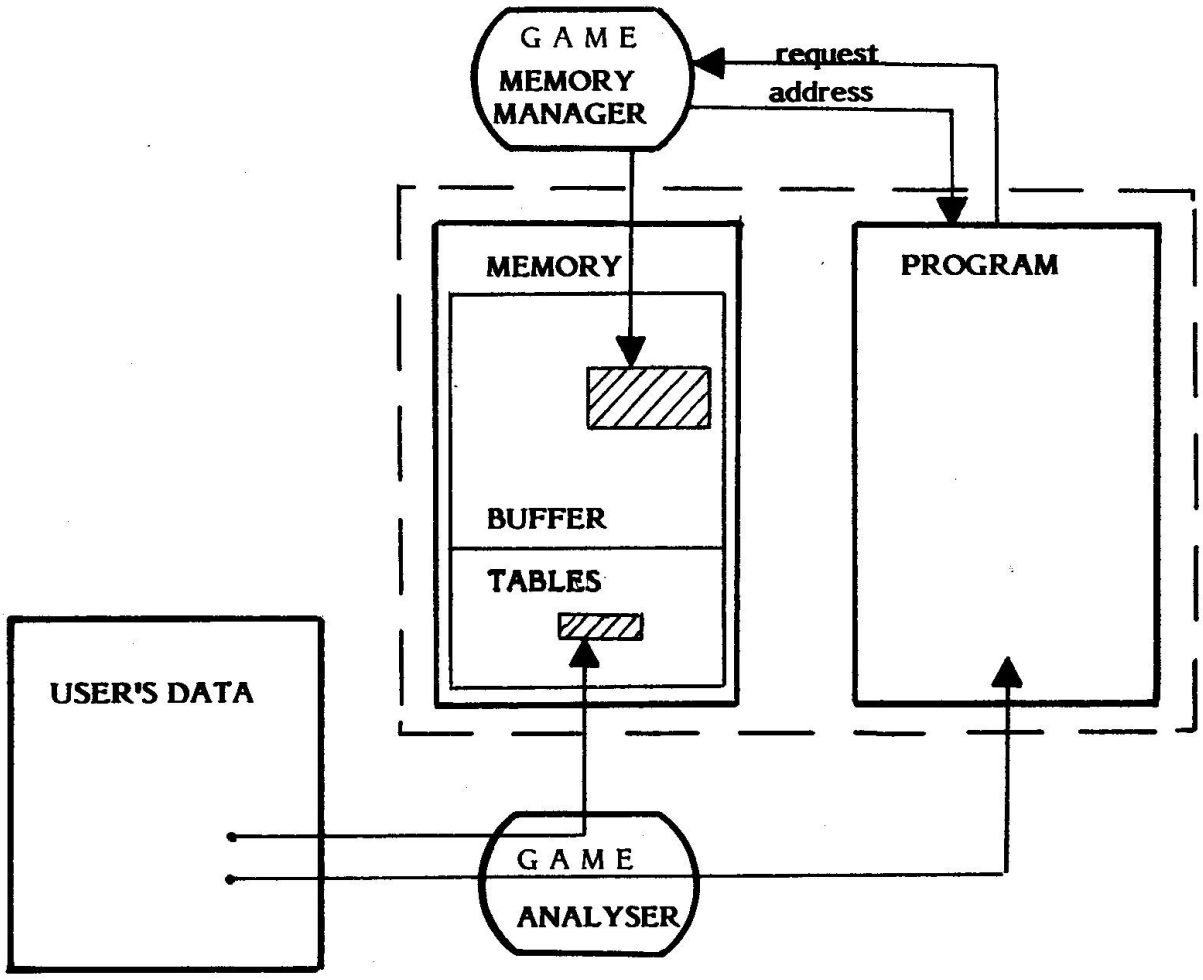


Figure 6
The memory manager

5. THE USER'S MANUALS

A full time secretary is in charge of updating, printing and dispatching the user's manual. Only one working copy is available to the EDP team. Upon correcting this copy, the programmer tags the page to be modified. Every so often, if there is no emergency,the secretary updates the original and goes ahead with the dispatching.

The entire user's manuals are kept on cassettes using a Xerox 800 machine.

The updates can be done at a chapter, a section, a part or a page level which is the smallest entity which can be updated.

The contents of the manual is standard. The following sections can be found :

- Presentation
- Hypothesis and conventions
- Presentation of results
- Data Input
- Methodology
- Running the program
- Tests and examples
- Error messages

Each page of the manual can be clearly located in its position. (See figure 7). A "control" sheet is sent together with each update thus giving the opportunity to the user to check the entire contents of his manual.

The mailing list of BC contains as of today 65 addresses.

6. CONCLUSION

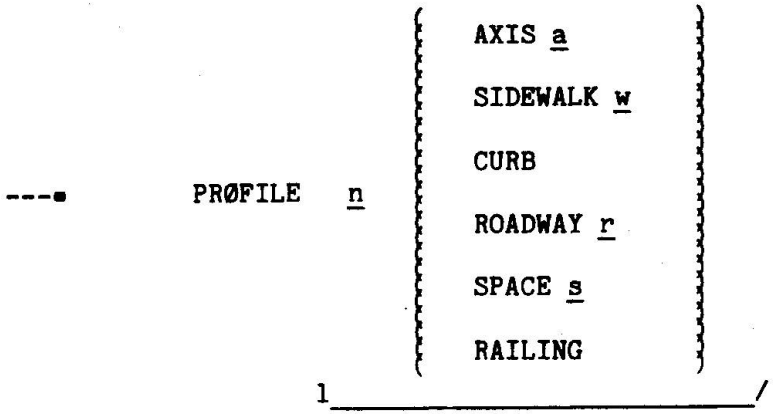
This paper was supposed to present a case history. Because EEG's EDP team has continuously aimed at standardizating it's methods,the resulting paper is more a presentation of the general methodology followed.

The key points of this methodology are :

- a standardization of the interface between program and engineer
- a high quality documentation
- a documentation which evoluates in order to be kept to a minimum.
- a simple in-house package which is used for data input, output, memory management and documentation generator.

E • E CALCUL SCIENTIFIQUE	CHAP : 20 - PROGRAMME B C	REF : 2.20.4.2
VOL : 2	SECT : 4 - Data Input	PAGE : 42
RED : JD	PART : 2 - Data base definition	DATE : 24.06.76
		VERS : 3,0

The following set of commands is available :



where :

PRØFILE n gives and I.D. number to the profile. This number will be used when defining spans/piers. Usually only one profile will be necessary for a given bridge.

AXIS a enables the user to position the profile with respect to the vertical x - y plane.

Figure 7
Frame of the user's manual