International Institute for Applied Systems Analysis
IIASA www.iiasa.ac.at

An Interactive System for Experimenting with Development Planning

Sokolov, V., Valasek, M. and Zimin, I.

IIASA Research Memorandum
December 1975

RM-75-69

AN INTERACTIVE SYSTEM FOR EXPERIMENTING WITH DEVELOPMENT PLANNING

V. Sokolov
M. Valasek
I. Zimin

December 1975

An Interactive System for Experimenting with Development Planning

V. Sokolov, M. Valasek, and I. Zimin

Abstract

An interactive system which is directed to serve as an instrument for resolving development planning problems is presented in the paper. Manuals for programmers and players and a few numerical examples are attached.

I. Introduction

The goal of this study is an interactive system which could serve as a flexible instrument for development planning (an ISDP) in complex environment involving different types of uncertainty and risks. A controlled system itself is defined at the level of an aggregated representation of a socio-economic system. A general context for the study is illustrated in Figure 1. There are two principle points in this approach: 1) inclusion of a decision maker as a natural component of the model, 2) and a possibility of controlling model complexity and interaction mechanisms.

The first point contributes to driving experimental conditions to real ones while the second one allows us to systematically process the results of experiments and build a consistent practical tool for planning. Thus, a systems analyst may conveniently change model parameters; model structure (by changing formal and informal links); and degree of uncertainty resulting from the lack of data about natural parameters, interactions between model elements.

Moreover, the systems analyst may easily affect structures and goals of experiments. A tentative list of experiments may include:

- studying principles of goal setting and goals' dynamics under conflicting resource requirements;
- analyzing external strategies of attaining the goals;
- testing mechanisms which generate conflict situations and finding efficient ways to resolve conflicts;

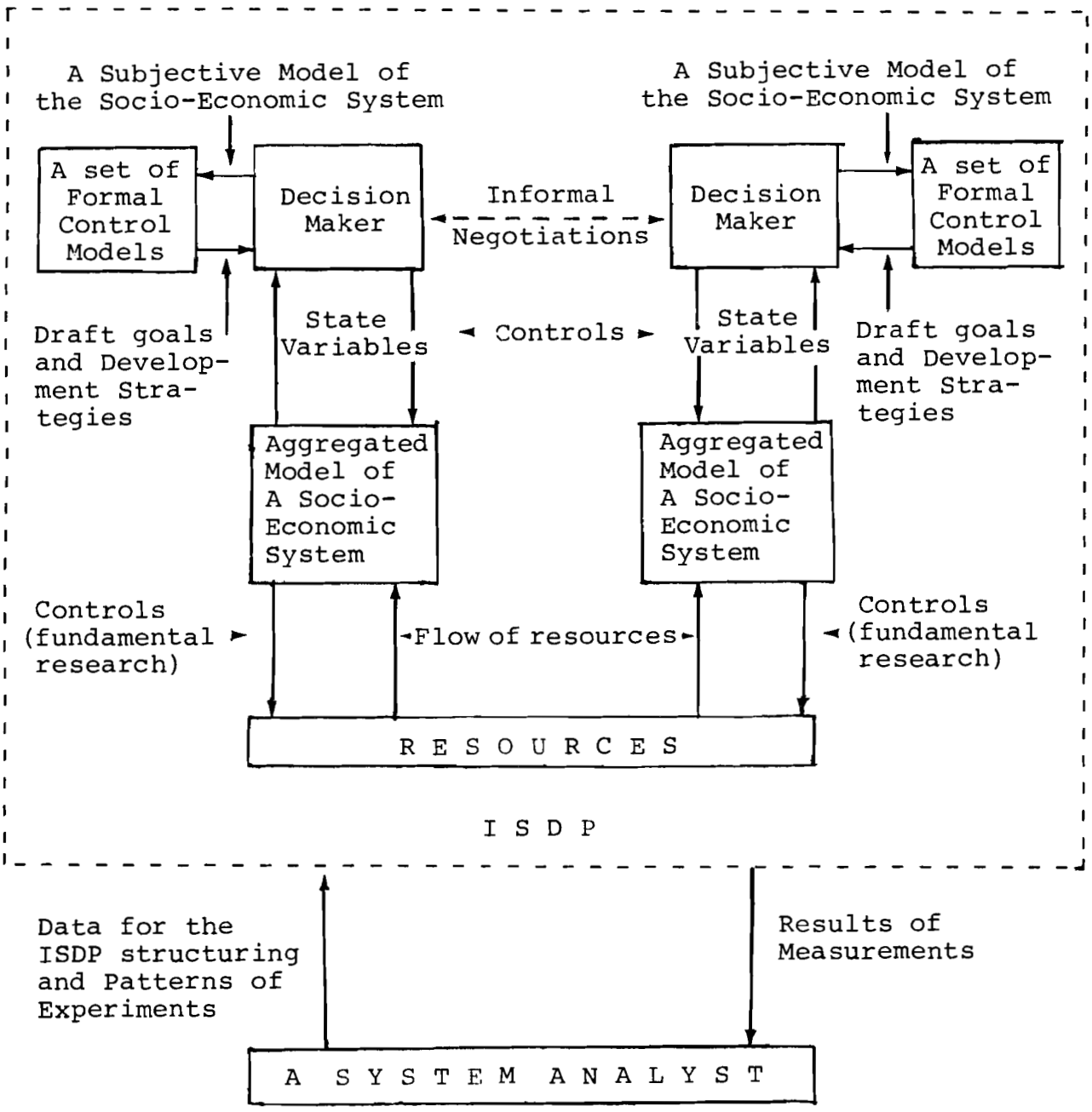

Figure 1. A flow diagram of an interactive system for studying development planning.

- specifying the operational information area of a decision maker which contains the most relevant data and procedures for decision making, etc.

ISDP basically assumes a gaming philosophy developed elsewhere [2-4] and evolves it through several consecutive stages. The first stage, presented in [5], is concerned with a formulation of basic principles and system structuring. The second stage, presented in this paper, deals with the computer implementation of ISDP.

II. Computer Implementation of ISDP

ISDP was implemented by the PDP 11/45 computer at the IIASA under the time-sharing UNIX operating system. Both programming and use are performed interactively through one of seven available terminals.

The programming system describing ISDP was written in Fortran. Fortran on the PDP 11/45 has the advantage of being fast and having powerful input/output statements. It can read and write files as well as type and receive data on terminal typewriters or displays.

Next, let us describe the ISDP programming system structure, and its generation and operation. ISDP consists of three parts intended for three different types of users:

1) a system programmer;

2) a game organizer (supervisor);

3) players.

If you are a player, there is no need to read the first two sections because players need not have any knowledge of the UNIX system or even of the programming itself.

2.1 Remarks for the System Programmer

The programming system operates within the hardware environment as shown in Figure 2. The system consists of sixteen disk files with a total size of 115 standard blocks. During the work with ISDP, all files must be available on line. The list of disk files with the respective sizes is as follows:

AY	13
COPY.F	13
DATA	5
GEN.F	2
GO	28
INS	4
LIST.F	2
P.F.	1

P.O.	1
PLOT	24
PLOT.F	4
PROG.F	13
S.F	1
S.O.	2
T.F	1
T.O	1

where

AY = the binary permanent disk file which keeps all A, Y, Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, and B values;[1]

COPY.F = the copy of the main program (PROG.F) before the last change of it (for verification only);

DATA = the input disk file containing all texts printed by the main program and values of Z-coefficients (this information could be easily updated using the text editor);

GEN.F = the disk file which contains a source program necessary to start the work with the system (the creation of AY-file, setting keywords and initial values of A);

GO = the compiled and linked (executable) version of the main program;

INS = the file containing information for users;

LIST.F = the source program which prints all values of A, Y, B on the line printer (after finishing the last step);

P.F = the source program of the function POS;

P.O = the compiled function POS;

PLOT = the compiled program PLOT.F which plots histograms on the CALCOMP-plotter after the last step;

PLOT.F = the source program for plotting results;

PROG.F = the source form of the main program (last version);

---

[1] Explanation of all variables is given below.

S.F = the source program of the function SUM;

S.O = the compiled program of the function SUM;

T.F = the source program of the subroutine TEXT;

T.O = the compiled subroutine TEXT.

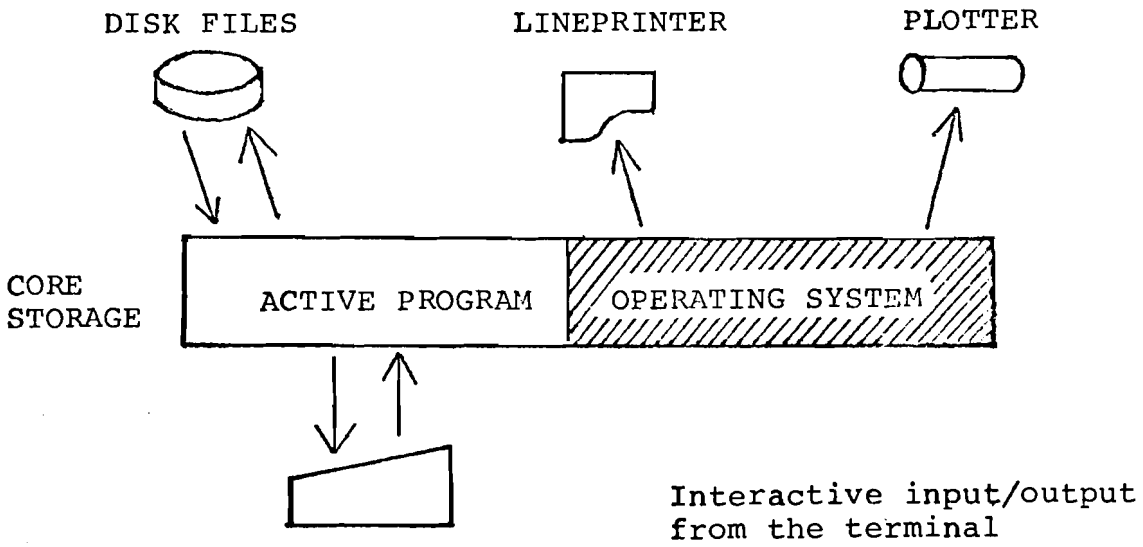

Figure 2. System hardware configuration.

All source files can be easily updated by using the text editor. Editorial functions and operating system commands are described in [1,6]. The program operator must dump the whole disk library on a magnetic tape after any change in the program and at certain time intervals. The following remarks about the compilation of programs are essential:

- Any file that will be compiled by the PDP-Fortran compiler must have a file name with an ".f" ending.

- To compile a Fortran program that does not call any subroutines the command FØ (completed by the system as FØRTRAN) should be followed by the source file name, for example, FORTRAN GEN.F. If the compilation was done successfully without errors, the compiler will automatically link the program, putting the completed linked module into an "A.OUT" file. After this, it is possible to run such a program generating initial values of A, Y, and B arrays by typing RUN A.OUT on the terminal.

To save the space on the user disk, it is better to delete file A.OUT after its execution by typing DELETE A.OUT; it can be compiled later when necessary.

- The main program PROG.F requires subroutines SUM, POS and TEXT; they are written on files S.F, P.F, and T.F. Therefore the compilation command for the main program is followed by "-c": FORTRAN PROG.F -C. This commands the compiler to compile the main programs, but not to link them. After this compilation the main program is written on a temporary file called PRØG.Ø. The linking command is FORTRAN PROG.O T.O P.O S.O. The completed linked module will be called A.OUT again and renamed as "GO" later by RENAME command. RUN GO is the command activating the execution of the current session with ISDP.

In the Appendix you can find a source text of all programs, subroutines, and data files (with comments). The meaning of array names and variables used in the main program is as follows:

```
A          = array of current phase variable values;
A(1,k,i)   = production output;
A(2,k,i)   = capital stock;
A(3,k,i)   = stock of resources;
A(4,k,i)   = amount of population;
A(5,k,i)   = accumulated knowledge;
A(6,k,i)   = total production funds;
A(7,k,i)   = pollution;
A(8,k,i)   = accumulated fundamental research investments;
A(9,k,i)   = standard of living;
k          = number of a time-step;
i          = number of a player.

Y          = array of current control variable values;
Y(1,k,i)   = investment into expansion of production sector;
Y(2,k,i)   = consumption per unit of population;
Y(3,k,i)   = protection investment;
Y(4,k,i)   = applied research investment;
Y(5,k,i)   = production resources;
Y(6,k,i)   = effective production funds;
Y(7,k,i)   = fundamental research investment;
Y(8,k,i)   - Y(13,k,i) = natural resource purchasing from
               the first to the sixth layers correspondingly;

Y1(k,i,j) = capital investment negotiated from the i-th
              element to the j-th element at time-step k;
Y2(k,i,j) = capital investment negotiated from the j-th
              element to the i-th element;
Y3(k,i,j) = capital out-flow from the i-th element to the
              j-th element owing to the purchase of resources;
Y4(k,i,j) = capital in-flow from the j-th element to the
              i-th element owing to the sale of resources;
```

Y5, Y6	= similar to Y3, Y4 variables concerning the knowledge of purchasing and selling respectively;
Y7(k,i,j)	= pollution exchange between the i-th and the j-th elements;
Z	= array of structural coefficients in the model;
B	= data matrix about the nature;
B(1,i)	= amount of resources in the i-th layer;
B(2,i)	= price of the i-th layer resource;
B(3,i)	= information threshold for entering the i-th layer;
B(4,i)	= capital threshold for entering the i-th layer;
K1,K2,K3	= keywords;
K11,K22,K33	= number of steps already done by the players;
T	= array containing all texts printed on the terminal.

2.2 Remarks for the Supervisor

To prepare to use ISDP one has to generate all the necessary files and programs in the core memory of the computer. The following sequence of instructions should be used for doing this:

```
Fortran (files) GEN.F
Run (Program) A.OUT
Delete (files) A.OUT.
```

After this, the work may be started with the player's command, Run (program) GO. Besides initiation of ISDP, a supervisor should perform the following functions:

- instruct all players how to operate the terminal and carry out sessions with ISDP;

- provide a proper sequence of player's moves during a session. Only one user may operate the terminal at one time, and the sequence of operation is either defined by the respective code words or may be established at random by a supervisor;

- return (if necessary) the whole system to one of its previous states by the following sequence of instructions:

```
Fortran (files) Change.F
Run (Program) A.OUT.
```

After this the computer asks the number of steps you want to start again with. You have to print a desired step and then Delete (files) A.OUT. After this instruction, the system may be initiated again by Run (Program) GO with initial conditions as given by the results of the given step:

- generate the disk file "AY" before the first step, set up new initial values of A and/or Z, and change keywords;

- check after each complete step if all players' actions were correct;

- print and/or plot results;

- publish results of experiments and carry out debriefing sessions;

- log out the terminal after using ISDP by typing "BYE" statement and thus confirming it.

2.3 Remarks for the Player

To activate ISDP one has to type RUN GO on the terminal keyboard. All command names are abbreviated; the full type-out will be completed by the computer as follows: RUN (PROGRAM) GO. After activating the terminal one has to set a current step number and a keyword. Following the keyword specification the computer displays initial conditions of phase variables related to specific players. The current values of phase variables are exemplified in the left part of the terminal screen while the values from a previous step are given in the right part to see trends of development. The following values are used as phase variables:

production output
capital stock
amount of resources
population
accumulated knowledge
total production funds
pollution
investments to discover layers
life standard.

A work with the system is essentially a selection of values of control variables from the following list:

0 help
1 negotiate cap in from unit a
2 negotiate cap in from unit b
3 negotiate cap out to unit a
4 negotiate cap out to unit b
5 negotiate res cap out to unit a
6 negotiate res cap out to unit b
7 negotiate res cap in from unit a
8 negotiate res cap in from unit b
9 negotiate kno cap out to unit a
10 negotiate kno cap out to unit b

```
11 negotiate kno cap in from unit a
12 negotiate kno cap in from unit b
13 define fund investments
14 define consumption
15 protection investment
16 knowledge investment
17 production's resources
18 determ pollution out to unit a
19 determ pollution out to unit b
20 define effective funds
21 fundamental research investment
22 nat. res. purchase from layer 1
23 nat. res. purchase from layer 2
24 nat. res. purchase from layer 3
25 nat. res. purchase from layer 4
26 nat. res. purchase from layer 5
27 nat. res. purchase from layer 6
28 avail. layers of nat. resources.
```

The selection process is governed by the block diagram of Figure 3. All control (input) variables are set to zero before each step. The computer runs the whole system after changing at least one of the input variables. The first run is done automatically with all input variables equal to zero. A player can input all decisions (controls) in a random sequence and change the decision later on, if necessary, and if he has not finished his current step. After finishing the step and typing the "END" statement modifications of control variables cannot be made. While working with a computer one should keep in mind that a unit number, a step number, a keyword number, and variable number are integers which do not require setting a decimal point. All other variables are real. The program expects the decimal point by typing it, for example:

```
KNOWLEDGE INVESTMENT
10.0.
```

After typing 0 (zero) as a name of a control variable, useful support information will appear on the terminal screen which identifies some critical factors that you should take into consideration more carefully in further planning. To finish the step type END to answer the question "IS IT OK?" otherwise type either OK (carriage return only) and go on with selecting other control variables, or NO if you intend to change the last input value.

One session of operation with ISDP includes ten steps which correspond to one, five, or ten years of real time and three players at the current implementation. However, these figures can be changed without any difficulties as the program has a module structure and is very easy to modify. Figures 4 and 5 illustrate the results of one session with ISDP when model coefficients were set as shown in Figure 6 and players were assigned the same goals to provide for the monotonic growth over the whole set of phase variables.

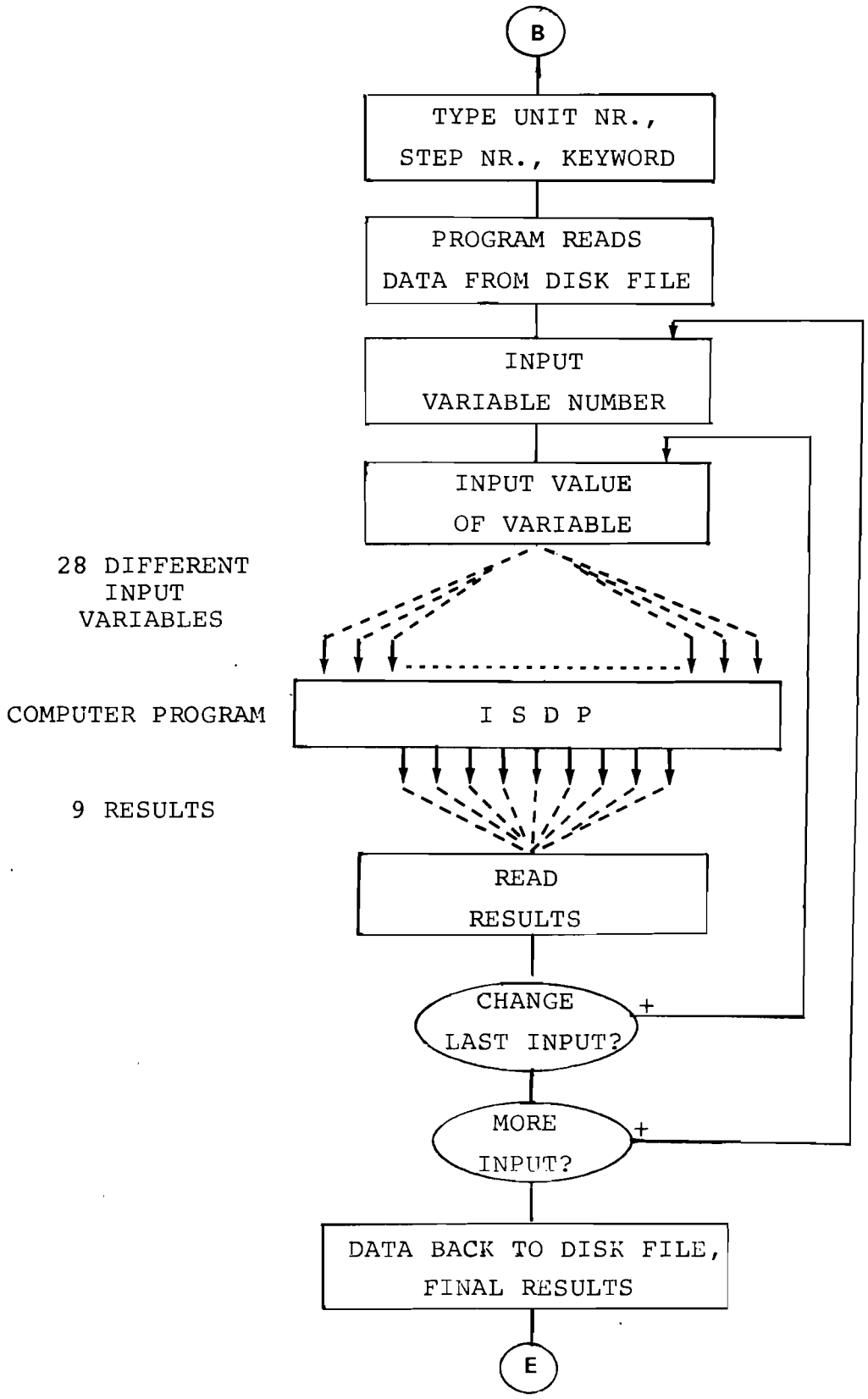

Figure 3. A block diagram of selecting control variables.

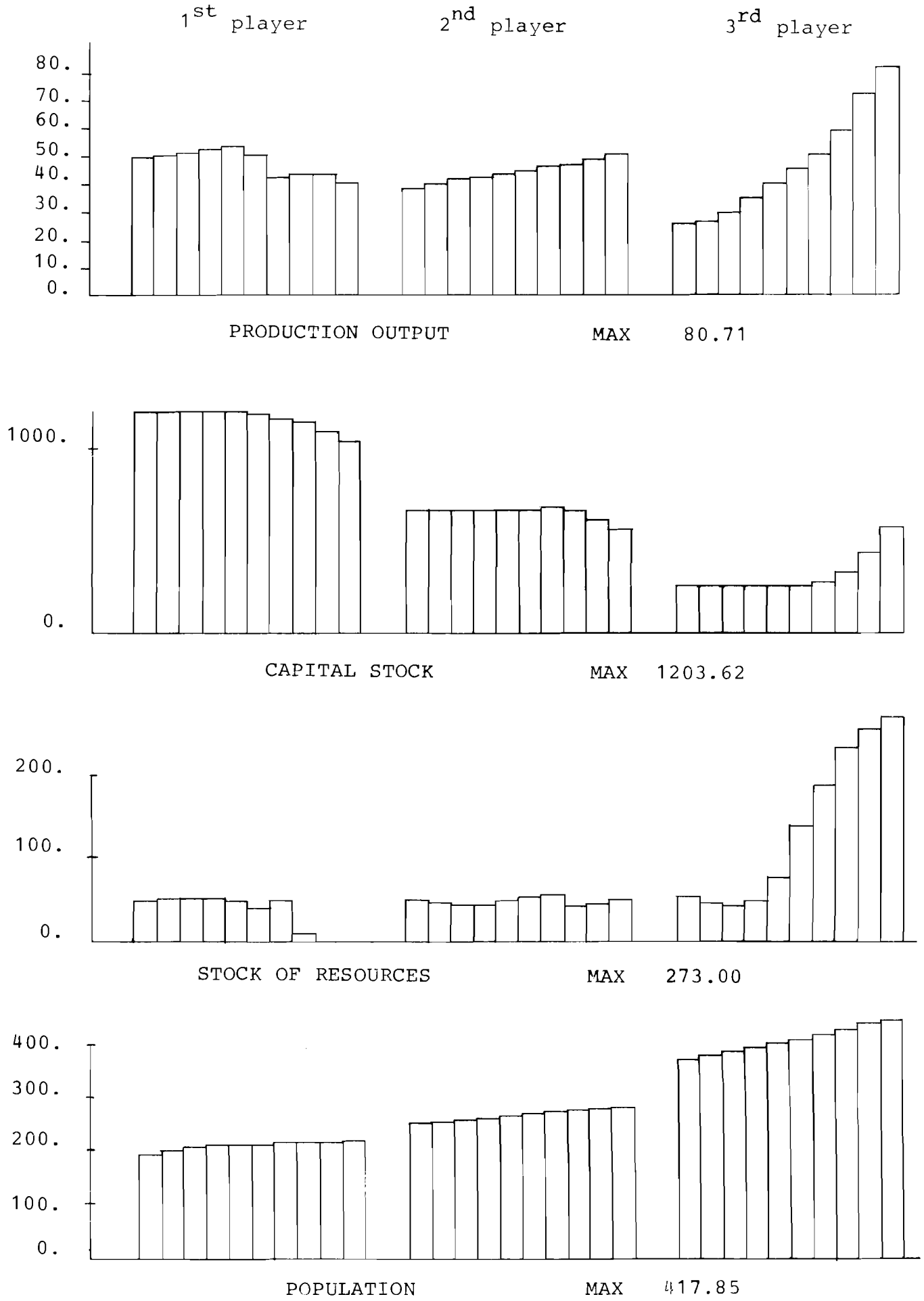
1st player
2nd player
3rd player
80.
70.
60.
50.
40.
30.
20.
10.
0.
PRODUCTION OUTPUT
MAX 80.71
1000.
0.
CAPITAL STOCK
MAX 1203.62
200.
100.
0.
STOCK OF RESOURCES
MAX 273.00
400.
300.
200.
100.
0.
POPULATION
MAX 417.85

Figure 4.

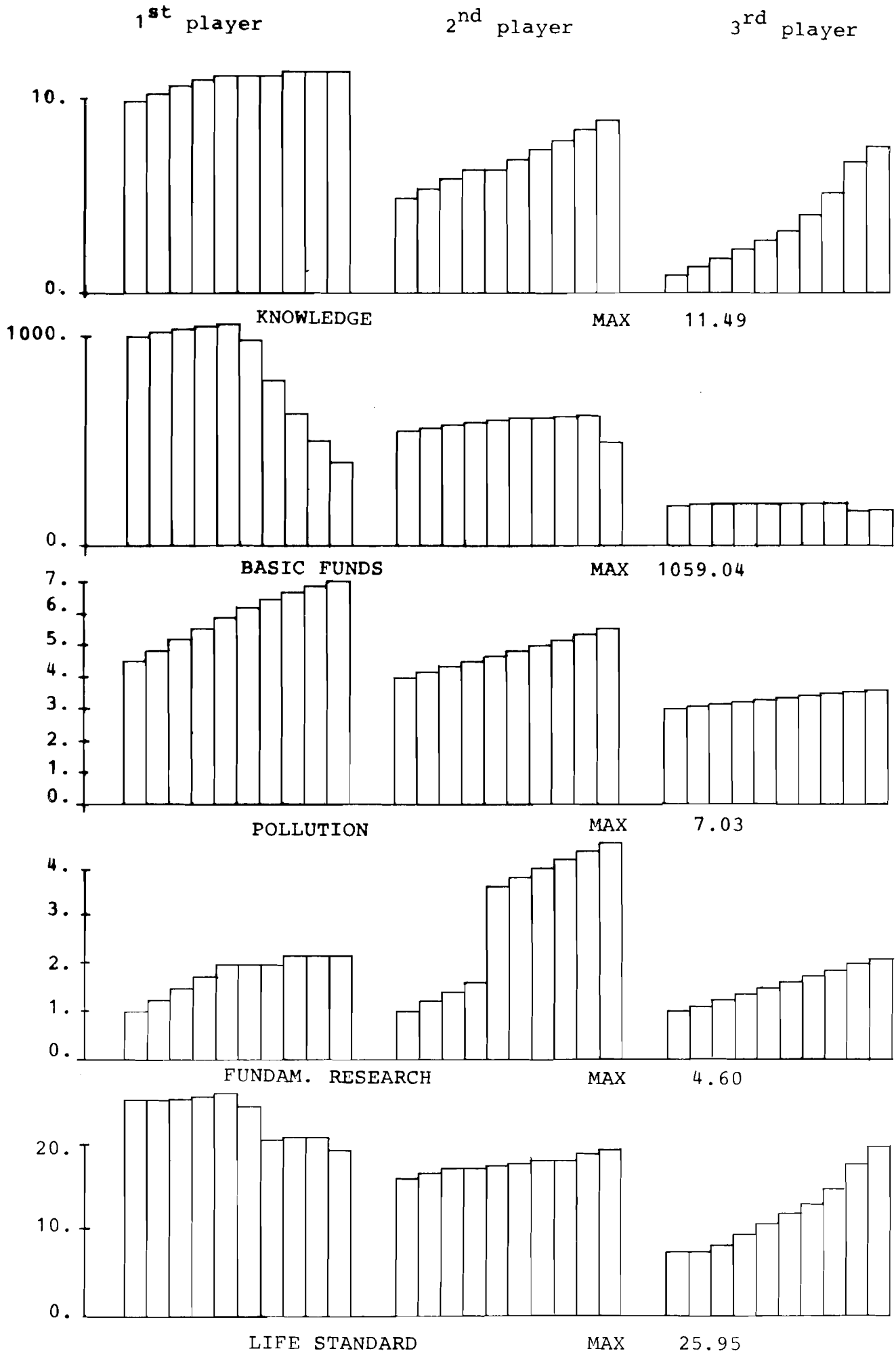
1st player
2nd player
3rd player
10.
0.
KNOWLEDGE
MAX 11.49
1000.
0.
BASIC FUNDS
MAX 1059.04
7.
6.
5.
4.
3.
2.
1.
0.
POLLUTION
MAX 7.03
4.
3.
2.
1.
0.
FUNDAM. RESEARCH
MAX 4.60
20.
10.
0.
LIFE STANDARD
MAX 25.95

Figure 5.

	1st Player	2nd Player	3rd Player
Z(1)	0.1	0.11	0.08
Z(2)	0.0005	0.0003	0.0007
Z(3)	2.5	2.5	2.5
Z(4)	2.5	2.5	2.5
Z(5)	0.001	0.001	0.001
Z(6)	0.05	0.05	0.05
Z(7)	0.2	0.2	0.2
Z(8)	44.0	32.0	25.0
Z(9)	0.09	0.059	0.024
Z(10)	2.0	2.0	2.0
Z(11)	0.05	0.05	0.05
Z(12)	5.0	5.0	5.0
Z(13)	0.0007	0.0005	0.0004
Z(14)	0.	0.	0.
Z(15)	0.03	0.02	0.01
Z(16)	0.22	0.366	1.0
Z(17)	0.1	0.12	0.15
A(1)	50.	38.	25.
A(2)	1200.	660.	240.
A(3)	50.	52.	55.
A(4)	200.	240.	350.
A(5)	10.	5.	1.
A(6)	1000.	550.	200.
A(7)	4.5	4.	3.
A(8)	1.0	1.0	1.0

Figure 6. Structural coefficients and initial values of phase variables in the model.

Additional constraints were imposed on the range of controls to account for the limitation existing in real systems (e.g. for the rate of growth of population or basic funds). In this experiment one step corresponds to ten years. Figures 4 and 5 clearly show that players 1 and 2 were not able to cope with the situation owing to the constraints on the resources available and they met serious crisis conditions, while the third player who started with not such good conditions gained a high rate of development by balancing the sale of resources with a reasonable economic development program and environmental protection policy.

III. Conclusions

This paper presents the second stage of research aimed at creating an interactive system for development planning under the complex conflicting environment. The results illustrate a computer implementation of the interactive system and instruct programmers, players, and experiment supervisors how to operate the system. The Appendix contains the whole set of source programs used in the system. The system as it stands now will be used in a wide program of experiments, some of which have been mentioned heretofore. Moreover, it will be further developed in two directions:

- more accurate presentation of system mechanisms;
- disaggregating the system by introducing several hierarchical levels of development planning and specifying respective mechanisms of interaction.

APPENDIX

Programs

1. The Main Program

```
DIMENSION A(9,1Ø,3),B(4,6)
DIMENSION X(6,3),SU(6)
DIMENSION Y(13,1Ø,3),Y1(1Ø,3,3),Y2(1Ø,3,3),Y3(1Ø,3,3)
DIMENSION Y4(1Ø,3,3),Y5(1Ø,3,3),Y6(1Ø,3,3)
DIMENSION Y7(1Ø,3,3),Y8(1Ø,3,3)
INTEGER T(1ØØ,8)
DIMENSION Z(17,3)
 COMMON/BB/T
C
 CALL SETFIL(3,"AY")
READ(3)K,K1,K2,K3,K11,K22,K33,A,Y,Y1,Y2,Y3,Y4,Y5,Y6,Y7,Y8,B
REWIND 3
C
 CALL SETFIL(2,"DATA")
DO 4Ø L1=1,1ØØ
READ(2,19)(T(L1,L2),L2=1,8)
4Ø CONTINUE
DO 5Ø L1=1,17
5Ø READ(2,18)(Z(L1,L2),L2=1,3)
REWIND 2
C
 CALL TEXT(9Ø)
WRITE(6,33)K
 CALL TEXT(91)
KK=K11-1
WRITE(6,33)KK
 CALL TEXT(92)
KK=K22-1
WRITE(6,33)KK
 CALL TEXT(93)
KK=K33-1
WRITE(6,33)KK
C
15Ø CALL TEXT(1)
READ(5,12)I
IF(I.GT.3.OR.I.LT.1)GO tO 15Ø
 CALL TEXT(2)
READ(5,13)KØ
IF(I.EQ.1)KK=K11
IF(I.EQ.2)KK=K22
IF(I.EQ.3)KK=K33
IF(K.NE.KØ.OR.K.NE.KK)GO TO 2ØØ
 CALL TEXT(3)
READ(5,14)KEY
```

Dimension Statements

Read old (or initial) values from disk file "AY"

Read the file "DATA"

Print the current step number and number of steps done by each unit

Read the unit number, step number and keyword

```
IF(I.EQ.1)KEY1=K1
IF(I.EQ.2)KEY1=K2
IF(I.EQ.3)KEY1=K3
IF(KEY1.NE.KEY)GO TO 2Ø1
C
IF(K.GT.1)GO TO 78
DO 6Ø L=1,17
6Ø WRITE(6,61)L,Z(L,I)
C
C
78 IF(I.EQ.1)M1=2
IF(I.EQ.1)M2=3
IF(I.EQ.2)M1=1
IF(I.EQ.2)M2=3
IF(I.EQ.3)M1=1
IF(I.EQ.3)M2=2
GO TO 3
C
4 CALL TEXT(4)
READ(5,6)M
IF(M.GT.28)GO TO 4
IF(M.GT.Ø)GO TO 39
IF(A(1,K+1,I).LE.Ø.)CALL TEXT(61)
IF(A(2,K+1,I).LE.Ø.Ø5*A(2,1,I))CALL TEXT(62)
IF(A(3,K+1,I).LE.Ø.Ø5*A(3,1,I))CALL TEXT(63)
IF(A(4,K+1,I).LE.Ø.Ø5*A(4,1,I))CALL TEXT(64)
IF(A(6,K+1,I).LE.Ø.Ø5*A(6,1,I))CALL TEXT(65)
IF(A(7,K+1,I).GE.5.)CALL TEXT(66)
 CALL TEXT(67)
GO TO 4
39 IF(M.LT.22.OR.M.EQ.28)GO TO 2
N1=M-21
IF(A(B,K,I).GE.B(4,N1))GO TO 2
 CALL TEXT(18)
GO TO 4
2 GO TO(3ØØ,3Ø1,3Ø2,3Ø3,3Ø4,3Ø5,3Ø6,3Ø7,3Ø8,3Ø9,31Ø,
&311,312,313,314,315,316,317,318,319,32Ø,321,322,323,324,
&325,326,327),M
C
3ØØ CALL TEXT(2Ø+M1)
READ(5,31)Y1(K,I,M1)
GO TO 3
3Ø1 CALL TEXT(2Ø+M2)
READ(5,31)Y1(K,I,M2)
GO TO 3
3Ø2 CALL TEXT(23+M1)
READ(5,31)Y2(K,I,M1)
GO TO 3
3Ø3 CALL TEXT(23+M2)
READ(5,31)Y2(K,I,M2)
 GO TO 3
3Ø4 CALL TEXT(26+M1)
ZZ=1./Z(3,M1)
WRITE(6,7)ZZ
READ(5,31)Y3(K,I,M1)
```

Print Z-values (before the first step only)

Setup of sequence numbers of other two players

Go through statements before first input

Read variable number

Help-command execution

Test whether natural resources are available

Switch according to input variables number

Print text on the terminal, calculate prices if any, read value of variable, go to gaming model execution

```
GO TO 3
3Ø5 CALL TEXT(26+M2)
ZZ=1./Z(3,M2)
WRITE(6,7)ZZ
READ(5,31)Y3(K,I,M2)
GO TO 3
3Ø6 CALL TEXT(29+M1)
ZZ=1./Z(4,I)
WRITE(6,7)ZZ
READ(5,31)Y4(K,I,M1)
GO TO 3
3Ø7 CALL TEXT(29+M2)
ZZ=1./Z(4,I)
WRITE(6,7)ZZ
READ(5,31)Y4(K,I,M2)
GO TO 3
C
 3Ø8 CALL TEXT(32+M1)
ZZ=1./Z(6,M1)
WRITE(6,7)ZZ
READ(5,31)Y5(K,I,M1)
GO TO 3
3Ø9 CALL TEXT(32+M2)
ZZ=1./Z(6,M2)
WRITE(6,7)ZZ
READ(5,31)Y5(K,I,M2)
GO TO 3
31Ø CALL TEXT(35+M1)
ZZ=1./Z(6,I)
WRITE(6,7)ZZ
READ(5,31)Y6(K,I,M1)
GO TO 3
311 CALL TEXT(35+M2)
ZZ=1./Z(6,I)
WRITE(6,7)ZZ
READ(5,31)Y6(K,I,M2)
GO TO 3
312 CALL TEXT(11)
READ(5,31)Y(1,K,I)
GO TO 3
313 CALL TEXT(12)
READ(5,31)Y(2,K,I)
GO TO 3
314 CALL TEXT(13)
READ(5,31)Y(3,K,I)
GO TO 3
315 CALL TEXT(14)
READ(5,31)Y(4,K,I)
GO TO 3
316 CALL TEXT(15)
READ(5,31)Y(5,K,I)
GO TO 3
317 CALL TEXT(41+M1)
READ(5,31)Y8(K,I,M1)
GO TO 3
```

The value of effective funds has an upper limit

```
318 CALL TEXT(41+M2)
READ(5,31)Y8(K,I,M2)
GO TO 3
C
319 AMIN=A(6,K,I)
IF(AMIN.GT.A(4,K,I)/Z(16,I))AMIN
&=A(4,K,I)/Z(16,I)
IF(AMIN.GT.Y(5,K,I)/Z(17,I))AMIN
&=Y(5,K,I)/Z(17,I)
 11 CALL TEXT(16)
 WRITE(6,2Ø)AMIN
READ(5,21)Y(6,K,I)
IF(Y(6,K,I).GT.AMIN)GO TO 11
GO TO 3
C
32Ø CALL TEXT(17)
READ(5,31)Y(7,K,I)
GO TO 3
321 CALL TEXT(7Ø)
READ(5,31)Y(8,K,I)
GO TO 3
322 CALL TEXT(71)
READ(5,31)Y(9,K,I)
GO TO 3
323 CALL TEXT(72)
READ(5,31)Y(1Ø,K,I)
GO TO 3
324 CALL TEXT(73)
READ(5,31)Y(11,K,I)
GO TO 3
325 CALL TEXT(74)
READ(5,31)Y(12,K,I)
GO TO 3
326 CALL TEXT(75)
READ(5,31)Y(13,K,I)
GO TO 3
C
327 DO 45 L5=1,6
45 IF(A(8,K,I).LT.B(4,L5))GO TO 46
46 L5=L5-1
IF(L5.GT.Ø)GO TO 8
 CALL TEXT(76)
GO TO 4
8 CALL TEXT(8Ø)
WRITE(6,44)(L6,L6=1,L5)
WRITE(6,47)(B(1,L6),L6=1,L5)
WRITE(6,48)(B(2,L6),L6=1,L5)
WRITE(6,49)(B(3,L6),L6=1,L5)
GO TO 4
C
3 A(2,K+1,I)=A(2,K,I)+A(1,K,I)+SUM(Y1,K,I)+SUM(Y4,K,I)
&+SUM(Y6,K,I)-(Y(1,K,I)+Y(2,K,I)*A(4,K,I)+Y(3,K,I)
&+SUM(Y3,K,I)+SUM(Y5,K,I)+Y(4,K,I)+SUM(Y2,K,I))
A(2,K+1,I)=A(2,K+1,I)-Y(8,K,I)-Y(9,K,I)-Y(1Ø,K,I)-Y(11,K,I)
&-Y(12,K,I)-Y(13,K,I)-Y(7,K,I)
```

The variable 28 is not any true control variable, but the command to see the table describing available natural resources

```
A(3,K+1,I)=A(3,K,I)+Z(3,I)*SUM(Y3,K,I)-Z(4,I)
&*SUM(Y4,K,I)-Y(6,K,I)*Z(17,I)
A(4,K+1,I)=A(4,K,I)*(1.-POS(Z(9,I)-Y(2,K,I))
&*Z(1Ø,I)+POS(Y(2,K,I)-Z(9,I))-POS(A(7,K,I)
&-Z(12,I))*Z(11,I))
IF(A(4,K+1,I).LT.Ø.)A(4,K+1,I)=Ø.
A(5,K+1,I)=A(5,K,I)+Z(5,I)*Y(4,K,I)*A(4,K,I)
&+Z(6,I)*SUM(Y5,K,I)
A(6,K+1,I)=A(6,K,I)*(1.-Z(7,I))+Y(1,K,I)
&*Z(8,I)
A(1,K+1,I)=Y(6,K,I)*(Z(1,I)+Z(2,I)*A(5,K,I))
A(7,K+1,I)=A(7,K,I)+Z(13,I)*Y(6,K,I)+Z(14,I)
&*Y[2,K,I)*A(4,K,I)+SUM(Y7,K,I)-SUM(Y8,K,I)-Z(15,I)*Y(3,K,I)
A(8,K+1,I)=A(8,K,I)+Y(7,K,I)
IF(A(4,K+1,I).GE.1.)GO TO 71
A(9,K+1,I)=Ø.
GO TO 72
71 A(9,K+1,I)=1ØØ.*A(1,K+1,I)/A(4,K+1,I)
C
72 DO 1 J=1,9
WRITE(6,22)J,A(J,K+1,I),A(J,K,I)
1 CONTINUE
IF(M.EQ.Ø)GO TO 4
WRITE(6,23)
READ(5,24)14
IF(14.EQ.4HOK.OR.14.EQ.4H)GO TO 4
IF(14.EQ.4HEND)GO TO 99
GO TO 2
C
99 DO 9Ø L=1,9
 CALL TEXT(5Ø+L)
WRITE(6,42)A(L,K+1,I)
9Ø CONTINUE
C
IF(I.EQ.1)K11=K11+1
IF(I.EQ.2)K22=K22+1
IF(I.EQ.3)K33=K33+1
IF(K11.NE.K22.OR.K22.NE.K33)GO TO 93
DO 77 I1=1,3
DO 77 J=1,3
77 Y7(k,J,I1)=Y8(K,I1,J)
DO 92 I1=1,3
92 A(7,K+1,I1)=A(7,K+1,I1)+SUM(Y7,K,I1)
DO 133 I1=1,3
DO 133 I2=1,6
X(12,I1)=Y(7+I2,K,I1)/B(2,I2)
IF(A(5,K,I1).LT.B(3,I2))x(I2,I1)=Ø.
133 SU(I2)=SU(I2)+X(I2,I1)
DO 134 I2=1,6
IF(SU(I2).LE.B(1,I2))GO TO 135
IF(SU(I2).EQ.Ø.)GO TO 134
DO 138 I1=1,3
138 X(I2,I1)=B(1,I2)*X(12,I1)/SU(I2)
B(1,I2)=Ø.
GO TO 134
```

The gaming model execution statements

Print results

Input OK, NO or END to continue, repeat input or finish the step

Print final results

Add one the number of steps

Update results when all three players finished the step (according to the natural resources purchase)

```
135 B(1,I2)=B(1,I2)-SU(I2)
134 CONTINUE
DO 136 I1=1,3
SUMA=Ø.
DO 137 I2=1,6
137 SUMA=SUMA+X(I2,I1)
136 A(3,K+1,I1)=A(3,K+1,I1)+SUMA
C
K=K+1
93 CALL SETFIL(3,"AY")
WRITE(3)K,K1,K2,K3,K11,K22,K33,A,Y,Y1,Y2,Y3,Y4,Y5,Y6,Y7,Y8,B
REWIND 3
 CALL TEXT(81)
 CALL TEXT(82)
 CALL TEXT(83)
STOP
C
2ØØ CALL TEXT(94)
STOP
2Ø1 CALL TEXT(95)
STOP
6 FORMAT(I2)
7 FORMAT(1ØX,6HPRICE=,F8.3)
12 FORMAT(I1)
13 FORMAT(I2)
14 FORMAT(I4)
18 FORMAT(3F1Ø.Ø)
19 FORMAT(8A4)
2Ø FORMAT(F 1Ø.3)
21 FORMAT(F 1Ø.3)
22 FORMAT(26X,2HA(,I2,2H)-,F13.2,3X,1H(,F13.2,1H))
23 FORMAT(1ØH IS IT OK?)
24 FORMAT(A4)
31 FORMAT(F1Ø.Ø)
33 FORMAT(I2)
42 FORMAT(F15.2)
44 FORMAT(16HLAYER NUMBER    ,619)
47 FORMAT(16HAMOUNT OF RES.  ,6F9.1)
48 FORMAT(16HPRICE FOR UNIT  ,6F9.4)
49 FORMAT(16HREQUESTED KNOWL.,6F9.1,/)
61 FORMAT(2HZ(,I2,2H)=,F1Ø,4)
END
```

Write the file "AY" back on the disk

Final messages

Invalid step number

Invalid keyword

Format Statements

2. The Function Sum

```
FUNCTION SUM(X,K,I)
DIMENSION X(1Ø,3,3)
SUM=Ø.Ø
DO 1 J=1,3
IF(J.EQ.I)GO TO 1
SUM=SUM+X(K,I,J)
1 CONTINUE
RETURN
END
```

3. The Function Pos.

```
FUNCTION POS(X)
IF(X)1,1,2
1 POS=Ø.
RETURN
2 POS=X
RETURN
END
```

4. The Subroutine Text

```
SUBROUTINE TEXT(N)
INTEGER T(1ØØ,8)
 COMMON/BB/T
WRITE(6,1)(T(N,J),J=1,8)
1 FORMAT(8A4)
RETURN
END
```

5. The Content of the Data File

```
YOUR UNIT NUMBER (1 INTEGER)
INPUT STEP NUMBER (2 INTEGERS)
YOUR KEYWORD PLEASE (4 DIGITS)
SELECT INPUT VARIABLE (2 INTEG.)
T5
T6
T7
T8
T9
T1Ø
DEFINE FUND INVESTMENTS
DEFINE CONSUMPTION
PROTECTION INVESTMENT
KNOWLEDGE INVESTMENT
PRODUCTION'S RESOURCES
DEFINE EFFEC.FUNDS LESS OR EQUAL
FUNDAMENTAL RESEARCH INVESTMENT
NO ACCESS TO THIS LAYER
T19
T2Ø
NEGOTIATE CAP IN FROM 1
NEGOTIATE CAP IN FROM 2
NEGOTIATE CAP IN FROM 3
NEGOTIATE CAP OUT TO 1
NEGOTIATE CAP OUT TO 2
NEGOTIATE CAP OUT TO 3
NEGOTIATE RES CAP OUT TO 1
NEGOTIATE RES CAP OUT TO 2
NEGOTIATE RES CAP OUT TO 3
NEGOTIATE RES CAP IN FROM 1
NEGOTIATE RES CAP IN FROM 2
NEGOTIATE RES CAP IN FROM 3
NEGOTIATE KNO CAP OUT TO 1
NEGOTIATE KNO CAP OUT TO 2
NEGOTIATE KNO CAP OUT TO 3
NEGOTIATE KNO CAP IN FROM 1
NEGOTIATE KNO CAP IN FROM 2
NEGOTIATE KNO CAP IN FROM 3
T39
T4Ø
T41
DETERM POLLUTION OUT TO 1
DETERM POLLUTION OUT TO 2
DETERM POLLUTION OUT TO 3
T45
T46
T47
T48
T49
T5Ø
PRODUCTION OUTPUT
CAPITAL STOCK
AMOUNT OF RESOURCES
POPULATION
ACCUMULATED KNOWLEDGE
TOTAL PRODUCTION FUNDS
```

Texts printed by the program on the terminal screen

```
POLLUTION
INVESTMENTS TO DISCOVER LAYERS
LIFE STANDARD
T6Ø
YOU PRODUCE NOTHING
YOUR CAPITAL IS EXHAUSTING
YOUR RESOURCES ARE EXHAUSTING
POPULATION IS DYING
BASIC FUNDS ARE DEPRECIATING
POLLUTION IS ENORMOUS
NO OTHER COMMENTS
T68
T69
NAT.RES.PURCHASE FROM LAYER 1
NAT.RES.PURCHASE FROM LAYER 2
NAT.RES.PURCHASE FROM LAYER 3
NAT.RES.PURCHASE FROM LAYER 4
NAT.RES.PURCHASE FROM LAYER 5
NAT.RES.PURCHASE FROM LAYER 6
NO ACCESS TO NATURAL RESOURCES
T77
T78
T79
AVAIL.LAYERS OF NAT.RESOURCES
THE STEP WAS FINISHED
I FEEL SOME RESULTS COULD BE
UPDATED.    ***GOODBYE-POP***
T84
T85
T86
T87
T88
CHANGE LAST INPUT
THE CURRENT STEP NUMBER
NUMBER OF STEPS DONE BY UNIT 1
NUMBER OF STEPS DONE BY UNIT 2
NUMBER OF STEPS DONE BY UNIT 3
INVALID STEP NUMBER
INVALID KEYWORD
T96
T97
T98
T99
T1Ø0
Ø.1Ø        Ø.11        Ø.Ø8
Ø.ØØ5       Ø.ØØ3       Ø.Ø7
2.5         2.5         2.5
2.5         2.5         2.5
Ø.ØØ1        Ø.ØØ1        Ø.ØØ1
Ø.Ø5        Ø.Ø5        Ø.Ø5
Ø.2         Ø.2         Ø.2
44.         32.          25.
Ø.Ø9        Ø.Ø59       Ø.Ø24
2.          2.          2.
Ø.Ø5        Ø.Ø5        Ø.Ø5
5.Ø         5.Ø         5.Ø
```

Z-values

Acknowledgment

We would like to thank Professor A. G. Aganbegjan who provided us with many useful comments on sessions and testing experiments with the model.

References

[1] Keringhan, B.W. "A Tutorial Introduction to the UNIX Text Editor." Murray Hill, New Jersey, Bell Laboratories, 1971.

[2] Meadows, D.H., Meadows, D.L., Randers, J., and Behrens, William III. "The Limits to Growth, A Report for the Club of Rome's Project on the Predicament of Mankind." New York, Potomac Associates, 1972.

[3] Mesarovic, M., and Pestel, E. Eds. "Multilevel Computer Model of World Development System." Extract from the Proceedings of the Symposium held at IIASA, Laxenburg, April 29-May 3, 1974. IIASA SP-74-1 to 6. Laxenburg, Austria, International Institute for Applied Systems Analysis, 1974.

[4] Pavlovsky, Yu. N. "Imitation Models of Historical Processes." International Seminar on Trends in Mathematical Modelling, Venice, December 13-18, 1971.

[5] Sokolov, V.B. and Zimin, I.N. "Gaming Model to Study the Problem of Sharing Natural Resources." IIASA RM-75-40. Laxenburg, Austria, International Institute for Applied Systems Analysis, 1975.

[6] "UNIX Programmer's Manual." Publication of Bell Telephone Laboratories, Incorporated. Fifth edition, June 1974.