DESIGN AND APPLICATION OF AN INTERACTIVE SIMULATION LANGUAGE

M. Alfonseca
IBM Scientific Center
P. Castellana,4.Madrid-1 (SPAIN)

The name SIAL/74 stands for two different concepts:

1. A digital continuous simulation language.
2. The system interactively implementing such language.

We shall subsequently explain both parts with some detail.

1. THE LANGUAGE

SIAL/74 is a block oriented analog-logical simulation language. We shall describe it with the help of a convenient example.

1.1. Description of the example.

The following system of differential equations is to be solved:

$$y'' = \frac{K^2}{y^3} - \frac{1}{2y^2}$$

$$\theta' = -\frac{K}{y^2}$$

The system defines the movement of a moving object which leaves the point of polar coordinates $(y(0),\theta(0))$ with an initial radial velocity $y'(0)$ and which is attracted by an inmobile body situated at the origin of the coordinates.

The constant K is proportional to the initial tangent velocity of the moving object. Note that when $K = 0$ (the initial tangent velocity is null), the system above is reduced to the following one:

$$y'' = -\frac{1}{2y^2}$$

$$\theta = \theta(0)$$

which defines the movement of a body falling directly towards another body.

The given system can be solved using standard analog computer procedures, by means of the following steps:

1. The differential equations are solved for the highest derivative term in every variable.(In our example, the equations are already given in that form).

2. The highest derivatives are supposed to be known, and integrated as many times as needed.

3. The second members of the equations obtained in 1. are constructed, and the loops are closed.

Although analog computer blocks use to introduce sign changes, these will not be considered in our language, as this will make programming easier. On the other hand, the symbols we shall use for our elementary blocks will be the commonly accepted ones in analog computer literature.

Applying steps 2. and 3. to the given system of differential equations, the following block diagram is obtained:(see figure 1)

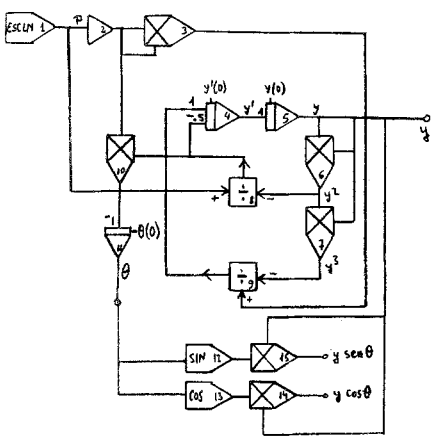


FIGURE 1. Block diagram for the gravity problem

1.2. SIAL/74 solution

The SIAL/74 program consists of the following parts:

1. A title which will head the print out of the results of the execution of the program.

2. The description of the block diagram, which consists of so many block descriptions as there are blocks in the diagram. A block des-

cription consists of a "code",defining the type of the block, and of two lists of numbers, one or both of which may not exist: a list of integers, giving the numbers of those blocks whose outputs are inputs for the block under consideration, and a list of parameters for the block. In the case of existence of only one list for a given type of block, the list will be given at the right of the block code. If both lists exist, the parameter list will appear to its left.

Each block will be assigned a number, its position within the block diagram description. Blocks may be numbered arbitrarily, that is to say, they may be defined in any order.

In our example, the description of the block diagram will consist of the following lines, in which the numbering appearing in figure 1 has been kept:

```
[1]   ESCLN ¯1
[2]   =0 SUM 1
[3]   MULT 2 2
[4]   ¯1.471 ¯.5 1 INTEGR 8 9
[5]   .316 1 INTEGR 4
[6]   MULT 5 5
[7]   MULT 5 6
[8]   DIV 1 6
[9]   DIV 3 7
[10]  MULT 2 8
[11]  1.5708 ¯1 INTEGR 10
[12]  SIN 11
[13]  COS 11
[14]  MULT 13  5
[15]  MULT 12  5
```

Block number 1 is defined as a step function with the step occurring at time = -1 (that is to say, before initial time for the simulation run; in this way, its output is assured to be the constant 1). In the definition of block number 2, another property of SIAL/74 is apparent: the definition of adjustable parameters. As soon as a value is encountered which is preceded by an equals sign, the parameter to which this value corresponds will be considered to be an adjustable parameter, with initial value the number after the equals sign. This feature is useful for the solving of automatic adjustment of parameters problems. Blocks number 2 defines thus an adder whose output is equal to the product of the adjustable parameter (initial value 0) times the output of block number 1.

Block number 3 multiplies by itself the output of block number 2.

(Squares it). Block number 4 defines an integrator. Its input is defined as the difference of the output of block number 9 and half the output of block number 8 (¯.5×Z[8]+1×Z[9]) and an initial output of ¯1.471 is given. The remaining block descriptions are self explaining.

56 different block types ("codes") may be used in the construction of SIAL/74 models. The number of blocks in a model is only limited by the work space size of the actual system where SIAL/74 is implemented.

3.After the description of the block diagram, a set of global data must be given. In our example, the following ones where defined:

```
Δ←1E¯3
II←.02
TM←.3
INTEG←RECT
I←5 11 4
PLOT←15 14
```

where Δ is the elemental time interval for the simulation run, II is the results print-out interval, TM is the final (end run) time, the value of I is a list with the numbers of the blocks whose outputs are to form part of the results of the execution, INTEG defines the integration method desired, and PLOT contains a list of block numbers. At the end of the execution a plot will be drawn of the outputs of all the blocks in the list but the last one, against the output of that one. If the plot is desired against time, the last number in the list must be 0.

2. THE SYSTEM

SIAL/74 has been implemented as an APLSV work space, containing a compiler which accepts SIAL/74 programs and translates them into equivalent APL programs, capable of execution on the same interpreter executing the compiler.

Two modes of compilation are possible: interactive and self-controlled. In the interactive mode, the source program sentences must be given individually and successively. This mode is specially useful at the time a program is entered for the first time. If an error is detected during the compilation, it is pointed out, and the same instruction corrected must be input again. In this way, at the end of the process, we have the corrected source program and the complete correspon-

ding object program will be available.

Once a program has been entered for the first time, in case we wish to correct it, it will be better to change directly the source program and to compile it once it has been corrected, instead of having to rewrite it. The compilation will be done in this case in the self-controlled mode; any error encountered now will be signaled, but it will not be possible to correct it during compilation.

We shall make this concept clearer with an example.

```
      COMPILAR
INTERACTIVE COMPILATION?
YES
SIAL→APL COMPILATION
TITLE:BESSEL EQUATION
[1]   RAMPA .0001
[2]   0 ¯1 ¯1 INTEGR 3 4
[3]   1 1 INTEGR 2
[4]   DIV 2 1
[5]   ¯1 SUM 2
[6]   //
//
[1]   TM←10
[2]   II←.5
[3]   Δ←.05
[4]   I←3 5
[5]   PLOT←3 5 0
[6]   INTGR←RECT
[7]
```

At this point, compilation is completed and we have the source program in the variable *FUENTE*.

```
      LISTAR FUENTE
[0]   ESSEL EQUATION
[1]   RAMPA .0001
[2]   0 ¯1 ¯1 INTEGR 3 4
[3]   1 1 INTEGR 2
[4]   DIV 2 1
[5]   ¯1 SUM 2
[6]   //
[7]   TM←10
[8]   II←.5
[9]   Δ←.05
[10]  I←3 5
[11]  PLOT←3 5 0
[12]  INTGR←RECT
```

To make changes in the input program, we may use the function CAMBIAR in the following way:

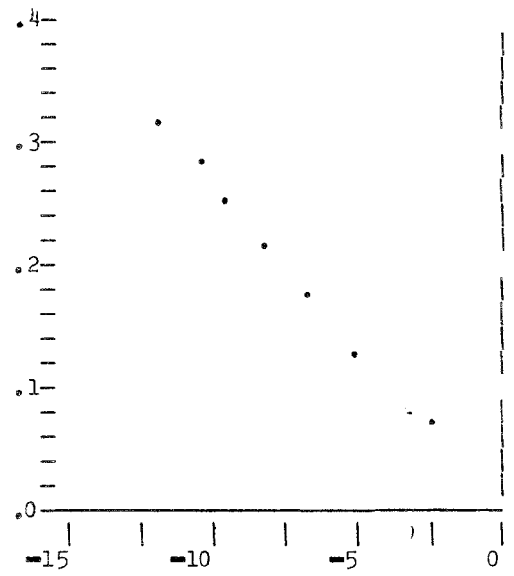
```
      FUENTE←CAMBIAR FUENTE
INSTRUCTION NUMBER:0
ESSEL EQUATION
/
?
BE
INSTRUCTION NUMBER:1
RAMPA .0001
          /
?
0000001
INSTRUCTION NUMBER:7
TM←10
   //
?
5
INSTRUCTION NUMBER:¯1
      LISTAR FUENTE
[0]   BESSEL EQUATION
[1]   RAMPA .0000000001
[2]   0 ¯1 ¯1 INTEGR 3 4
[3]   1 1 INTEGR 2
[4]   DIV 2 1
[5]   ¯1 SUM 2
[6]   //
[7]   TM←5
[8]   II←.5
[9]   Δ←.05
[10]  I←3 5
[11]  PLOT←3 5 0
[12]  INTGR←RECT
      COMPILAR
INTERACTIVE COMPILATION?
NO
PROGRAM NAME?
□:
      FUENTE
[0]   BESSEL EQUATION
[1]   RAMPA .0000000001
[2]   0 ¯1 ¯1 INTEGR 3 4
[3]   1 1 INTEGR 2
[4]   DIV 2 1
[5]   ¯1 SUM 2
[6]   //
[7]   TM←5
[8]   II←.5
[9]   Δ←.05
[10]  I←3 5
[11]  PLOT←3 5 0
[12]  INTGR←RECT
SIAL→APL COMPILATION
```

At this point the source program has been corrected and recompiled. The pair of object programs are contained in the variables PO and POI.

3. SAMPLE SESSION

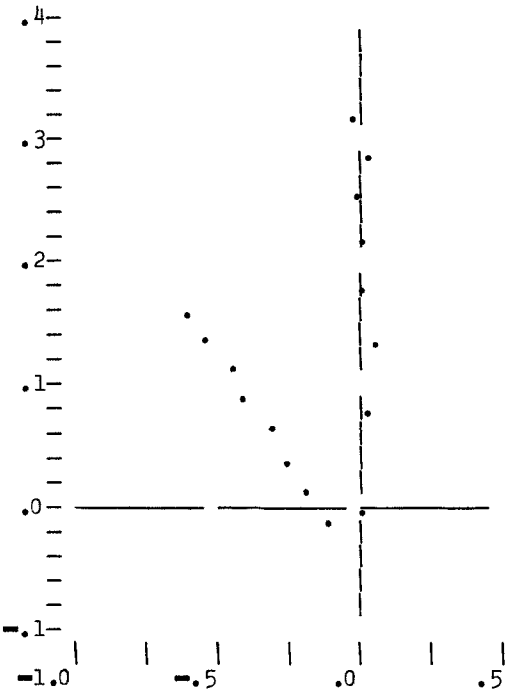
In this sample session, we shall solve the gravitational problem explained before, corresponding to Figure 1.

```
      COMPILAR
DESEA COMPILACION INTERACTIVA?
NO
QUE PROGRAMA DESEA COMPILAR?
□:
      GRAVITACION
[0]   GRAVITACION
[1]   ESCLN ¯1
[2]   =0 SUM 1
[3]   MULT 2 2
[4]   ¯1.471 ¯.5 1 INTEGR 8 9
[5]   .316 1 INTEGR 4
[6]   MULT 5 5
[7]   MULT 5 6    *ESTE ES EL CUBO DE Y
[8]   DIV 1 6
[9]   DIV 3 7
[10]  MULT 2 8
[11]  1.5708 ¯1 INTEGR 10
[12]  SIN 11
[13]  COS 11
[14]  MULT 13 5
[15]  MULT 12 5
[16]  //
[17]  Δ←1E¯3
[18]  II←.02
[19]  TM←.3
[20]  INTEG←RECT
[21]  I←5 11 4
[22]  PLOT←15 14
COMPILACION SIAL→APL
      EJEC POI
      TM←.13
      'GRAV'DEF PO
GRAV
      GRAV
GRAVITACION
TIEMPO          5          11           4
   .0000  3.160E¯01  1.571E00  ¯1.471E00
   .0200  2.856E¯01  1.571E00  ¯1.581E00
   .0400  2.527E¯01  1.571E00  ¯1.719E00
   .0600  2.167E¯01  1.571E00  ¯1.899E00
   .0800  1.764E¯01  1.571E00  ¯2.157E00
   .1000  1.297E¯01  1.571E00  ¯2.583E00
   .1200  7.026E¯02  1.571E00  ¯3.610E00
```



SCALE FACTOR FOR ABSCISSA 1E-7

```
      A INITIAL TANGENT VELOCITY:.158
      EJEC POI
      P[1]←.158
      GRAV
GRAVITACION
TIEMPO          5          11           4
   .0000  3.160E-01  1.571E00  -1.471E00
   .0200  2.857E-01  1.536E00  -1.563E00
   .0400  2.535E-01  1.493E00  -1.675E00
   .0600  2.187E-01  1.436E00  -1.815E00
   .0800  1.807E-01  1.357E00  -2.003E00
   .1000  1.383E-01  1.233E00  -2.271E00
   .1200  8.914E-02  9.858E-01  -2.700E00
   .1400  3.099E-02  -7.953E-02  -2.552E00
   .1600  8.068E-02  -2.972E00   4.368E00
   .1800  1.635E-01  -3.218E00   3.955E00
   .2000  2.406E-01  -3.299E00   3.763E00
   .2200  3.148E-01  -3.341E00   3.654E00
   .2400  3.871E-01  -3.368E00   3.583E00
   .2600  4.583E-01  -3.386E00   3.533E00
   .2800  5.286E-01  -3.399E00   3.496E00
   .3000  5.982E-01  -3.409E00   3.467E00
```



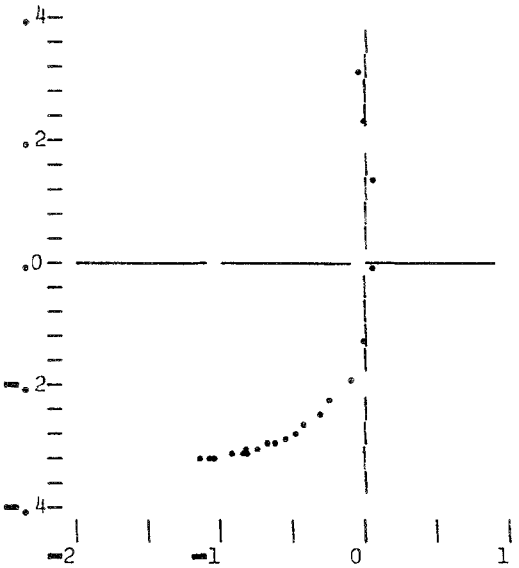

```
      ⍝ (A HYPERBOLIC MOVEMENT AROUND THE ORIGIN
      ⍝  CAN BE SEEN)


      ⍝ IN THE NEXT CASE, THE INITIAL TANGENT
      ⍝ VELOCITY WILL BE EQUAL TO .316

      EJEC POI
      P[1]←.316
      TM←1
      II←.05
```

GRAV
GRAVITACION

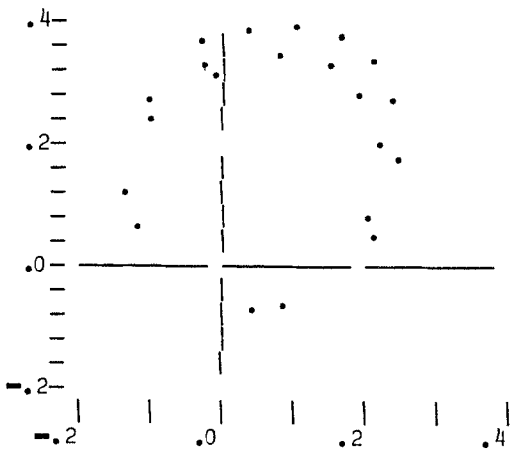
TIEMPO	5	11	4
.0000	3.160E-01	1.571E00	-1.471E00
.0500	2.402E-01	1.364E00	-1.560E00
.1000	1.616E-01	9.604E-01	-1.541E00
.1500	1.013E-01	-6.345E-02	-4.820E-01
.2000	1.331E-01	-1.440E00	1.443E00
.2500	2.132E-01	-2.007E00	1.651E00
.3000	2.942E-01	-2.260E00	1.575E00
.3500	3.707E-01	-2.406E00	1.485E00
.4000	4.430E-01	-2.502E00	1.408E00
.4500	5.118E-01	-2.572E00	1.344E00
.5000	5.776E-01	-2.625E00	1.291E00
.5500	6.410E-01	-2.668E00	1.245E00
.6000	7.023E-01	-2.703E00	1.206E00
.6500	7.618E-01	-2.733E00	1.172E00
.7000	8.197E-01	-2.758E00	1.142E00
.7500	8.761E-01	-2.780E00	1.116E00
.8000	9.313E-01	-2.800E00	1.092E00
.8500	9.854E-01	-2.817E00	1.070E00
.9000	1.038E00	-2.832E00	1.051E00
.9500	1.090E00	-2.846E00	1.033E00
1.0000	1.142E00	-2.859E00	1.016E00



A HYPERBOLIC MOVEMENT WITH LARGER EXCENTRICITY CAN BE SEEN

```
      ⍝ IN OUR LAST EXAMPLE, THE MOVING POINT
      ⍝ HAS A POSITIVE RADIAL VELOCITY OF .5,
      ⍝ AND AN INITIAL TANGENT VELOCITY OF .25

      EJEC POI
      P[1]←.25
      CI[1]←.5
      II←.1
      TM←2
      GRAV
GRAVITACION
TIEMPO          5           11           4
   .0000  3.160E¯01   1.571E00   5.000E¯01
   .1000  3.519E¯01   1.349E00   2.216E¯01
   .2000  3.614E¯01   1.154E00   ¯3.157E¯02
   .3000  3.457E¯01   9.569E¯01  ¯2.879E¯01
   .4000  3.030E¯01   7.220E¯01  ¯5.772E¯01
   .5000  2.279E¯01   3.694E¯01  ¯9.480E¯01
   .6000  1.116E¯01   ¯5.689E¯01 ¯1.292E00
   .7000  1.349E¯01   ¯3.636E00   1.390E00
   .8000  2.534E¯01   ¯4.345E00   9.652E¯01
   .9000  3.326E¯01   ¯4.637E00   6.331E¯01
  1.0000  3.829E¯01   ¯4.831E00   3.803E¯01
  1.1000  4.100E¯01   ¯4.989E00   1.637E¯01
  1.2000  4.163E¯01   ¯5.135E00   ¯3.927E¯02
  1.3000  4.023E¯01   ¯5.283E00   ¯2.452E¯01
  1.4000  3.668E¯01   ¯5.450E00   ¯4.718E¯01
  1.5000  3.066E¯01   ¯5.669E00   ¯7.470E¯01
  1.6000  2.143E¯01   ¯6.039E00   ¯1.125E00
  1.7000  8.520E¯02   ¯7.383E00   ¯1.069E00
  1.8000  1.697E¯01   ¯1.018E01   1.406E00
  1.9000  2.883E¯01   ¯1.068E01   9.854E¯01
  2.0000  3.719E¯01   ¯1.091E01   7.009E¯01
```




```
      ⍝ IN THIS CASE, THE MOVING POINT IS CAPTURED
      ⍝ AND DESCRIBES AN ELLYPSE.
```

4. CONCLUSSIONS

An interactive simulation language has been developed. Trough applications such as the gravity problem described above, and several biological simulations such as a conditional reflex model and the system for the regulation of body-water volume, its flexibility has been demonstrated.

REFERENCES

M.Alfonseca "SIAL/74: Lenguaje de simulación digital contínua", PCI-06.74, Nov.1974, UAM-IBM Scientific Center Publications.