Programming without tears.

On plane, or on bullock cart?

Károly Farkas

Budapest Tech Polytechnical Institution John von Neumann Faculty of Informatics
Bécsi út 96/b, 1034 Budapest, Hungary
farkas.karoly@nik.bmf.hu

Abstract. I think that life-long playing is more important than life-long learning. The teaching of informatics may be playful and enjoyable on every level of education. Perhaps this can be the most enjoyable subject. Use the computer in an adequate way, make the computer compute and do not compute yourself! There are several methodological elements which serve these goals: to push an advantage of motivation, creativity, syntonicity, the logo-pedagogy by Papert, the school of thinking by Pólya, and the tenure of computer as a plane not so as a bullock cart.

Key words: syntonicity, logo-pedagogy, Pólya, Papert, playful informatics

1 Introduction

Is it true that scientific knowledge can only be achieved with hard work? Can we accept that at university the only correct discipline results in the failure of a huge number of students? Is it possible to learn programming in an amusing way?

Informatics subjects may cause satisfaction on all levels of education, however, they may also be scholastic, boring and may cause suffering to the student. In this work some methodological elements will be listed that enhance learning informatics and they make the learning process more enjoyable at the same time.

2 Motivation, creativity

‘Great inventions solve great problems, but even the smallest problem requires some little invention. The problem you are thinking about may be simple, but if it arouses your interest, activates your inventiveness, and finally if you manage to solve it on your own, you experience the thrill and triumph of discovery. … What is realised of it greatly depends on the mathematics teacher. If the mathematics teacher spends most of his/her times making the pupils solve routine problems, he/she kills their interest, hinders their mental development, and turns the favourable conditions on their wrong side. He has to realise that solving a mathematics problem can be as enjoyable as doing a crossword puzzle, or a strenuous mental exercise can be as good as a game of tennis.' [4]

György Pólya's book was primarily written for mathematics teachers or students dealing with mathematics, but his words are worth bearing in mind for every teacher and are highly recommended to teachers of information science. And indeed, if we keep saying, for example, when teaching Logo: repeat it four times, fifty forward, ninety to the right, then joy of creation is lost. If Michelangelo carves chair legs, he is no more than a craftsman. We have to set a goal that can be achieved. We have to rely on our knowledge in carrying out the task. We would like to make the turtle draw the picture of a beautiful castle. At first sight it seems to be a very complicated task. But if we analyse the task and divide it into parts, it does not seem to be unrealisable anymore. We set about the task with pupils in the fourth form. We named the picture 'The Enchanted Castle'.

So the teacher must choose only the suitable curricula, and show the right way on which the student can rich to the goal.

3 Syntonicity

A useful part of the informatics-pedagogic is the logo-pedagogic. One of the most important pedagogical benefits of Logo is that it is an effective tool for improving thinking. Let us see what is the syntonicity. The examples are made with one of the best new Logo versions, namely Elica. Elica is made by Pavel Boytchev. [2] It is a free and can be downloaded from the net.

How many methods do you know to draw a circle? Let us see some made by computer.

First of all here it is algorithm of Paper [3]:

```
repeat 360[fd 1 rt 1]
```

A similar algorithm, but easier to observe by people, is `repeat 36[fd 1 rt 10]`: you step a little, turn a little, and your trace becomes a circle. To understand the algorithm it is important to understand the movement itself. Papert says:

'The Turtle circle incident illustrates syntonic learning. This term is borrowed from clinical psychology and can be contrasted to the dissociated learning already discussed. Sometimes the term is used with qualifiers that refer to kinds of syntonicity. For example, the Turtle circle is body syntonic in the circle is firmly related to children's sense and knowledge about their own bodies.' 'Turtle geometry is learnable because it is syntonic. And it is an aid to learning other things because it encourages the conscious, deliberate use of problem-solving and mathematic strategies. Mathematician George Polya has argued that general methods for solving problems should be taught.' 'Turtle geometry lends itself to this exercise. The key to finding out how to make a Turtle draw a circle is to refer to a problem whose solution is known very well indeed – the problem of walking in a circle.' [3]

Logo is one of the simplest and most teachable languages. And there is one more important thing – similarly to Mathematics, Logo is suitable for increasing the ability of rapid thinking. That’s why we think that math teaching could become more effective and more enjoyable if we use Logo, especially in earlier ages. Mathphobia is quite common in math classes. However, using Logo helps us to eliminate it, because children love computers.

One of the main features of Logo is its turtle geometry. Whatever turtle we play with it helps us to use the advantage of syntonicity.

Here is an algorithm to draw a square:

```
fd 55 rt 90   fd 55 rt 90   fd 55 rt 90   fd 55 rt 90
```

Let us notice that the fragment fd 55 rt 90 repeats several times. An important aspect of thinking is the recognition of such repeat patterns.

We can now provide a shorter algorithm for drawing a square:

```
repeat 4[fd 55 rt 90]
```

To make it more understandable we can imagine these steps as if done by ourselves – first we move forward, then we turn left, and so on.

The algorithm of drawing a triangle with the use of syntonic is obvious:

```
repeat 3[fd 55 rt 120]
```

We can see that the correct angle for turning to the right is 120, because in this way the actual angles in the triangle would be 60 degrees each. After this experiment it is not hard to see that when drawing a pentagon, a hexagon, or a n-gon, we should turn 360/5, 360/6, or 360/n degrees. There is no need for calculating 360/n, we can tell to the computer: ‘you must calculate 360 divide 5, and use the result’.

When we increase the number of the sides we start to approximate a circle. The next commands draw a 360-polygon which is considered a circle in Logo communities:

```
repeat 360[fd 1 rt 1]
```

Another classical example of the turtle geometry is to draw a house. The next example shows how the problem can be divided into two simple problems:

```
SQUARE
transfer
TRIANGLE
return
```

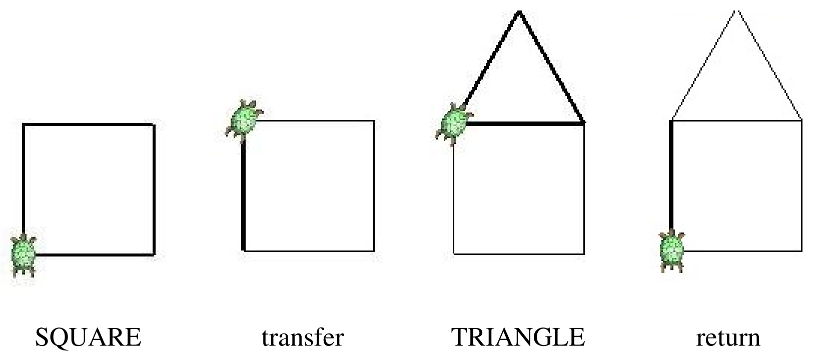


Fig. 1. ‘Divide the problem into parts!’

We use the commands transfer and return to position correctly the upper half of the house. For example, the transfer commands might be fd 55 rt 30. The return commands are necessary to complete the procedure and get the turtle back to where it was initially.

According to the book ’Mindstorms’: ‘*Turtle geometry provides excellent opportunities to practice the art of spilling difficulties. For example, HOUSE was made by first making SQUARE and TRIANGLE. In sort, I believe that Turtle geometry lends itself so well to Polya’s principles that the best way to explain Polya students is let them learn Turtle geometry. Thus Turtle geometry serves as a carrier for the general ideas of a heuristic strategy.’* [3]

How can we draw a circle in another way?

A possibly more syntonicity method is to draw the circle this way:

```
repeat 360[penup fd 55 pendown fd 1 penup fd -56 rt 1]
```

This short program draws a circle by using traditional Logo primitives. The method features two turtles. ’Adam’ is tied to ’Eve’. ’Eve’ is in front of Adam and the distance is constant between them at every time. Adam is spinning, so that Eve is drawing a circle.

We can draw a circle using the equation $x^2 + y^2 = r^2$ and the setpos primitive. This solution is also well-known.

Certainly we can use the circle logo primitive too. In the Elica circle requires four parameters, the three coordinates, and the radius:

```
make "c1 circle point 0 0 0 30
```

Before we show another algorithm for drawing a circle in 3D, let us see how we can draw a cube in Elica.

Again following Polya’s ideas let us first look for something similar! Apparently the cube has some similarity with the square. We already have a procedure for making a square:

```
to square repeat 4[fd 55 rt 90] end
```

Do we need to draw six squares for each side of the cube? No, it is not necessary. Placing four squares as shown below is enough. Now we use another form of square:

```
to square2 repeat 4[fd 55 up 90] end
```

And then the cube algorithm becomes:

```
to cube repeat 4[square2 fd 55 rt 90] end
```

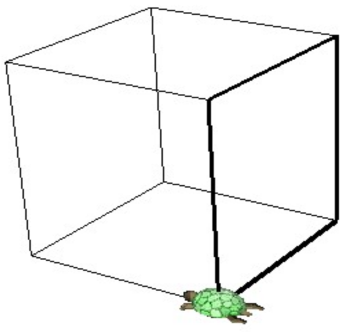


Fig. 2. Algorithm for cube

Let us now go back to the circle. To draw a vertical circle we may use this command:

```
repeat 360[fd 1 up 1]
```

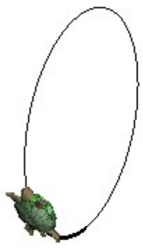


Fig. 3. Circle with a tumbling turtle

This interpretation is, of course, the well-known algorithm by Papert, but realized in other dimension.

The algorithm for drawing a tetrahedron is edifying too. We just act in a similar manner! In the same way as we draw a cube from four squares, we draw a tetrahedron from three triangles.

```
to triangle repeat 3[fd 55 rt 120] end
to tetrahedron repeat 3[rightroll -70.5 triangle right-
roll 70.5 fd 55 rt 120]end
```

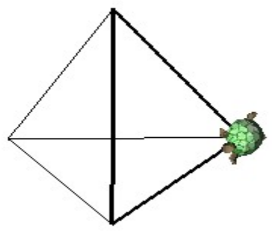


Fig. 4. Tetrahedron I.

The angle of rotation of 70,5 degrees is easy to find by experimenting. For example if we try 60 degrees the result will be ‘a closed rose with three petals’. At 90 degrees the flower will be wide opened. So the right value is between 60 and 90. Trying 75 degrees also gives us an opened rose so we continue to search the angle somewhere between 60 and 75.

When we use a computer we can do more frequent approximation, because the machine works quickly. Of course we can find the exact mathematical value for the angle and modify the algorithm like this:

```
 to tetrahedron
   make "angle 2*arcsin 1/(sqrt 3)
   repeat 3[rightroll -:angle triangle rightroll :angle
   fd 55 rt 120]
 end
```

The logo-pedagogy has a valuable element - the microworld. Elica has some microworlds too. For example there is a car turtle. By using it we can draw a circle in a very simple way:

```
rt 360
```

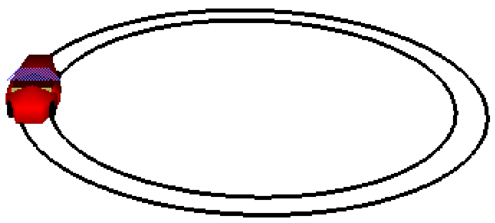


Fig. 5. Circle with the car turtle

This microworld is very interesting for smaller children. The spherical turtle may be interesting for the older students. This is a turtle model on the sphere's surface. To use it we must first say that our turtle is a spherical one:

```
make "defaultturtle sphericalturtle
```

Now we can draw a spherical triangle which has three 90-degree angles:

```
repeat 3[fd 90 rt 90]
```

After executing the command `demo 300` we can see a triangle in which the sum of the angles is 270.

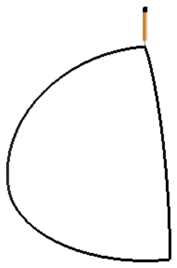


Fig. 6. Spherical-triangle.

The spherical turtle geometry gives us another interesting situation. Imagine we go forward some distance then we turn 90 degrees and go forward the same distance as before, and finally we are on the same initial position. This is possible only if we use spherical turtles:

```
repeat 2[fd 180 rt 90]
```

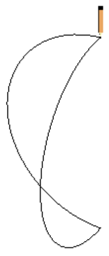


Fig. 7. 'Spherical twoangle'

Following this line of thoughts we can draw an 'oneangle'. And this happens to be a new algorithm for drawing a circle:

```
make "defaultturtle sphericalturtle
fd 360
```

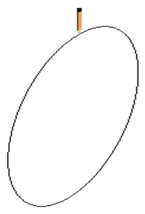


Fig. 8. Circle with Spherical turtle

After this we can draw orb as well:

```
repeat 18[fd 360 rt 10]
```

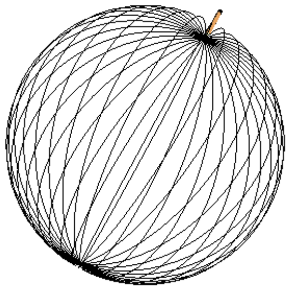


Fig. 9. Orb

4 Computer-syntonicity

Computer languages are also useful in a non-computer context in informatics games to enable students to write programs that describe the solutions to a particular range of problems. For example Logo has been used with success to describe LEGO construction algorithms. Me described [2] how a Logo-like language called Brick-Logo was developed to use with Lego-Logo. It is a pseudo-language using Logo commands, to describe three-dimensional turtle graphics. In a computer implementation of Brick-Logo, a turtle would move around leaving LEGO brick as it goes, but thus building any LEGO structure that was programmed. The example in Figure 10 illustrates how a chimney structure could be described in Brick-Logo.

In the non-computer example, the student design and build simple structures from small LEGO bricks. At the same time they discuss and make notes on the algorithm describing their construction. This game is particularly valuable as it developing multiple abilities at the same time: algorithmic thinking, spatial vision, manual manipulation, etc. With the help of Brick-Logo, for example, we can promote the discovery of symmetries and realize a transfer effect of problem-solving abilities.

Program to describe a chimney in Brick-Logo:

```
to level1 repeat 4[put_forward brick_right] end

to lift1 brick_rise brick_right end

to level2 repeat 4[put_forward  brick_left] end

to lift2 brick_rise brick_left end

to chimney :h
 repeat h/2 [level1 lift1 level2 lift2]
end
```

Where :h the number of rows in the chimney.

In this Logo version put means take down a brick, brick_right means turn the brick right 90 degrees around the corner, brick_rise means lift one brick-think.

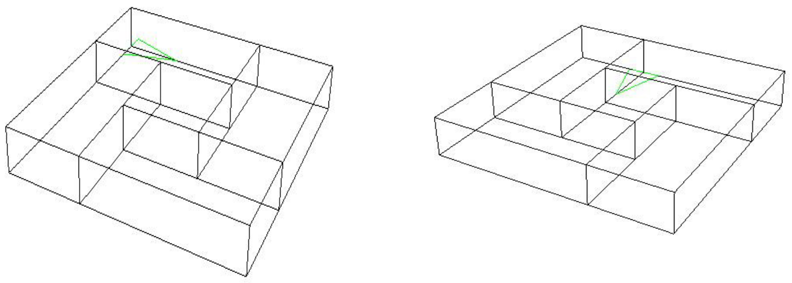


Fig. 10.

Figure 10 shows the cross-section for each alternative layer of bricks form the chimney.

5 What can you use the computer for?

The curriculum and the teaching methodology are in close connection. Pólya wrote in his famous book that the ability of thinking is more important than knowledge, hence how is more important in teaching than what.

Let us see one chapter of the subject operational research, namely linear programming, one of whose key elements is the basis transformation. It is evident that this may be helpful to develop mathematical thinking. But is this part of mathematics a good choice for the curriculum? Can the average student apprehend this theme? *'It is important to remember the distinction between* ***mathematics*** *- a vast domain of inquiry whose beauty is rarely suspected by most nonmathematicians - and something else which I shall call* ***math or school math****.'* wrote Papert when he disserted about the math curricula of the primary school. Is linear programming adequate for the students in colleges? It is known that most people in the US solve the addition 1/2 + 2/3 to get 3/5 ! Of course I do not say this is the goal, even I bemoan that we are getting near the US in this respect. In one of the colleges, at the comprehensive exam of informatics – which comes later than math - I usually ask the students about the dice. The students usually know the probability of the outcome six (1/6). But many times the answer is wrong when the outcome is five; they guess that the probability is 1/5.

In Hungary, in many colleges, an operation research program optimization must be solved on paper by hand. It is hard work. A professor declared, 'To solve this problem using a computer is not operational research. But this is a typical exam - you can make the algorithm easy with many iterations, which in real life will be solved by computer.' Why would you teach the plane pilot how to ride a bullock cart?

Citation again from Papert: '*Before electronic calculators existed it was a practical social necessity that many people be „programmed" to perform such operation as long division quickly and accurately. But now that we can purchase calculators cheaply we should reconsider the need to expend several hundred hours of every child's life on earning such arithmetic functions. … But utility was only one of the historical reasons for school math. … As I see it, a major factor that determined what mathematics went into school math was what could be done in the setting of school classrooms with the primitive technology of pencil and paper.*' [3]

6 Example taken from the subject of operation research – shipping exercise

A shipping exercise is traditionally solved after the unit cost matrix has been derived and the conditions have been defined as follows:

- Reduction of the unit cost matrix.
- Provision of a distribution of goodapproximation using one of the methods (e.g. Vogel- Korda).
- Improvement of the programme if it is not optimal, using the potentials.

The solution of this exercise requires the introduction and understanding of several notions (potential, loop, etc.).

A manual solution is time-consuming even in the case of textbook exercises (i.e. few dispatch locations and only a few destinations, single-digit figures). So much so, that one is bound to make mistakes in the course of solving the problem (just as it would happen when writing an unstructured, linear computer software). Finding the error is an almost hopeless effort - it is easier to start the whole exercise over again. Rest assured, this will not be your students' favourite pastime.

The same exercise using Solver means the following:

Reduction of the matrix is not necessary.

The first distribution can be entirely layman-like, not even the parameter equations need to be met:

Once the calculation method for the cost matrix (spreadsheet multiplication) and the calculation of the total cost (summa) have been stated, all we have to do is fill in the Solver auxiliary chart. Rather than calculating ourselves, we have to make the system calculate in our stead.

If there is an optimum, the computer will calculate it (obviously incomparably faster than manual calculation would be) at the required accuracy.

The solution can be reached easily and efficiently using real data, as well. What's more, though, the linear model in economics is a rough and ready abstraction of reality. Twice as many products do not involve twice the amount of costs and do not result in twice the yield. In terms of the development of thinking in economics terms, the most important benefit of using Solver is, in my opinion, that we have a choice of several mathematical models, instead of linearity we can assume relations that are in closer proximity to reality. 'Microsoft Excel Solver uses the method of General Reduced Gradient non-linear optimisation (GRG2), developed by Leon Lasdon (University of Texas at Austin) and Allan Waren (Cleveland State University).'

7 On similarity

Any economist, engineer, or professional has to have a clear understanding that linearity but often even geometrical similarity is an unnecessary and often disqualifying condition for similarity. I analysed in detail the relationship between geometrical and functional similarity in my PhD thesis myself.

Ervin Szücs says in his argumentation:

'A flea can jump at a hight of approximately 1 metre. How high could the flea jump if it was as tall as a human being? The mobility of living beings is subject, among others, to the relation between their muscle strength and their body mass. Muscle strength is proportionate to the cross section of the muscle, i.e. the square of the λ linear dimensions, while the body mass is proportionate to the volume, i.e. the third power of the linear dimensions. The hight h of the original jump is proportionate to the ratio between the strength and the mass:

$$h \approx \lambda^2 / \lambda^3 = 1 / \lambda$$

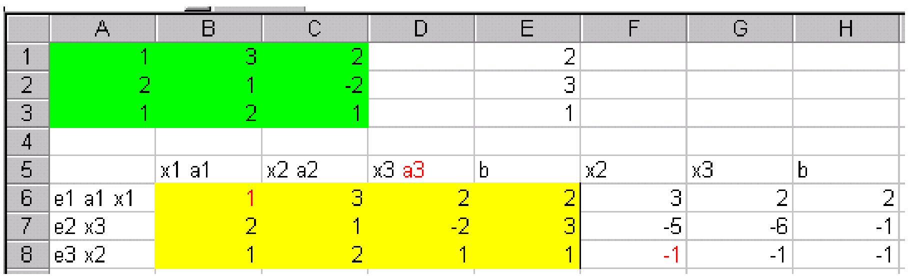
If every dimension of the animal is increased c-fold: $\lambda' = c\ \lambda$, , then we arrive at a being which is geometrically similar, the jump height of which is $h' = h / c$, i.e. declined c-fold! The flea as tall as a human being is approximately thousand times bigger than the original animal. The order of magnitude of the jump height of such an animal would be *mm* instead of the original *m*.' [5]

8 Excel, as a decision support system

Let us return to the solution of the shipping exercise using a computer. Once the shipping programme is ready, then it can be easily recalculated within a matter of seconds (!) after changing some of the basic data. The new, changed outcome, based on the new parameters is immediately visible, thus enabling competition among the various scenarios! When used in this way, Excel can indeed be viewed as a management information system, as well. This is exactly what the work of the economist is all about: weighing the benefits and drawbacks of the different solutions based on the performance of dozens of calculations and finally decision making on the basis of the various results, rather than machine-like, accurate manual performance of long algorithms. Computers can be used for counting, as well! How else could mathematics, advance mathematics be taught and applied in the age of computers, other than using computers? How could mathematical models be used without the help of computer science?

If my arguments have failed to convince someone, who will continue to use, let's say the simplex method, he should, at least consider writing it in Excel and having the computer perform the basic mathematical operations of the steps.

The first three lines of the following table (Fig.11.) show the coefficients and constants of a linear system of equation.



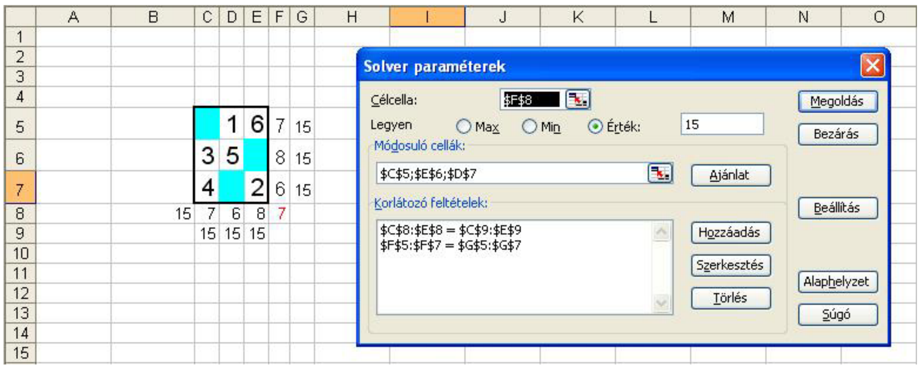
	A	B	C	D	E	F	G	H
1	1	3	2		2			
2	2	1	-2		3			
3	1	2	1		1			
4								
5		x1 a1	x2 a2	x3 a3	b	x2	x3	b
6	e1 a1 x1	1	3	2	2	3	2	2
7	e2 x3	2	1	-2	3	-5	-6	-1
8	e3 x2	1	2	1	1	-1	-1	-1

Fig. 11.

One possible solution is the following: as a first step the 1 in cell B6 is selected as a generating element. The formula in cell F6 is thus: C6/$B6, which can be dragged into two columns horizontally (arriving at the delta values). The formula to be entered into cell F7 is: C7-$B7*F$6, which can be dragged into two columns horizontally, and finally the section between f and h of row 7 can be dragged downwards vertically. Subsequently the value -1 of cell F7 is selected as a generating element….etc. Not only does the computer lend itself to the calculation, but the dragging accelerates the programming, as well. This solution can be compared to the rubber tyre mounted on the ox-driven cart. It is even better, though, to use containers on railways, or even air cargo for shipping these days (i.e. to use the Solver addition instead of the simplex method).

9 Solving the magic square using Solver

The task of finding the missing elements of the magic square can be solved without writing a programme, as well. Writing, let us say, all ones instead of the missing elements, as a first approximation, one of the equation criteria of the row, the column and the diagonal sums, is selected as a target function and the value of the other sums are made equal to the value containing the desired square feature, as a restricting condition.



	B	C	D	E	F	G
5			1	6	7	15
6		3	5		8	15
7		4		2	6	15
8	15	7	6	8	7	
9		15	15	15		

Solver paraméterek

Célcella: $F$8

Legyen ( ) Max ( ) Min [✓ Érték:] 15

Módosuló cellák: $C$5;$E$6;$D$7

Korlátozó feltételek:

```
$C$8:$E$8 = $C$9:$E$9
$F$5:$F$7 = $G$5:$G$7
```

Fig. 12. Magic square

10 Programme assembly instead of programme writing

Instead of writing programmes, it is more and more common that programme elements are used as building blocks to assemble programmes. An example for this is the use of Enterprise Guide of the SAS software packages, as well. Users can build and update charts, which can then be visually queried and connected without SQL expertise and the support of IT specialists. Users can view the code of the generated query in advance, can verify its syntax or can copy the code for repeated use in other applications.

The same can be observed in the case of the Sybase Power Designer. The source code automatically generated in the background is the result of the use of symbols for building and assembly. While SAS generates dot.net source codes, Sybase uses the programming language C#. Users do not even need to be familiar with these, though.

References

1. Boytchev, P.: http://www.elica.net (2007-04-15)
2. Farkas Károly: Informatics Games for Developing Children's Thinking Ability. In. Johnson D.C. Samways B. editors: IFIP Transactions. IA-34 Informatics and Changes in Learning. North-Holland, Amsterdam-London-New York-Tokyo, 1993.
3. Papert, S.: Mindstorms. Basic books.
4. Pólya Gy.: A gondolkodás iskolája (How to solve it. A new aspect of mathematical method), Bibliotheca, Budapest. 1957.
5. Szücs Ervin: Modellezés elmélete és gyakorlata. (Theory and Practice of Modelling) Szolnoki F iskola, Szolnok, 2000.