New generation Graphical programming
for the all of the specialists, not just for programmers

Velbitskiy I.V.
GLUSHKOV Fund
Ukraine
glushkov.org
ivelbit@gmail.com

Abstract— New generation visual programming technology with R-charts (VTR) for the first time offers not to write but *to draw* programs in their all life cycle according to ISO/IEC 8631H:1989(E). It is easier, faster and more efficient to draw than it is made now by traditional methods in the form of texts. It simplifies and improves the existing programming process by some times and allows for the all of the specialists to program but not just for programmers. At present moment the graphical development environment (no-commercial version), which has 14 visual controls and contains the graphical editor, conversion of R-charts into R*-charts (R-charts without implementation details) and back, translator of RR*-charts into C++, etc. was implemented.

Keywords- Graphical programming, R-chart, R*-chart, visual programming with RR*-charts, graphic shell, graphical development environment, programming for all specialists, arc-loaded graph, parallel programming, the proof of correctness of programs, a meta-language for specifying the syntax and semantics, network graph, automatic generation of the work plan.

I. ESSENCE OF NEW GENERATION TECHNOLOGY

The new technology instead of traditional operators like **if**, **for**, **while**, **goto**, etc (**all!** about 10) are invited to use one, only the R-chart - horizontal arc that has a direction to the right or to the left, Figure 1. At the top of the arc are ALWAYS written Condition passing along an arc, and the bottom Actions executed hereby, the arrow of arc points to the next top for analysis or execution. The top has no name and sets the state of program or process of its development. Any

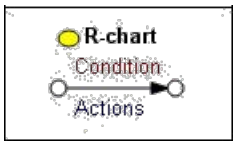


Fig. 1. R-chart (one arc to the right or to the left) to record any program in any language.

number of arcs to the left and/or to the right may come out from the top. Fig. 2 shows R-charts of selection and loop statements and the corresponding record in C++ language at the right. On the left below in Fig. 2 there is R*-chart,

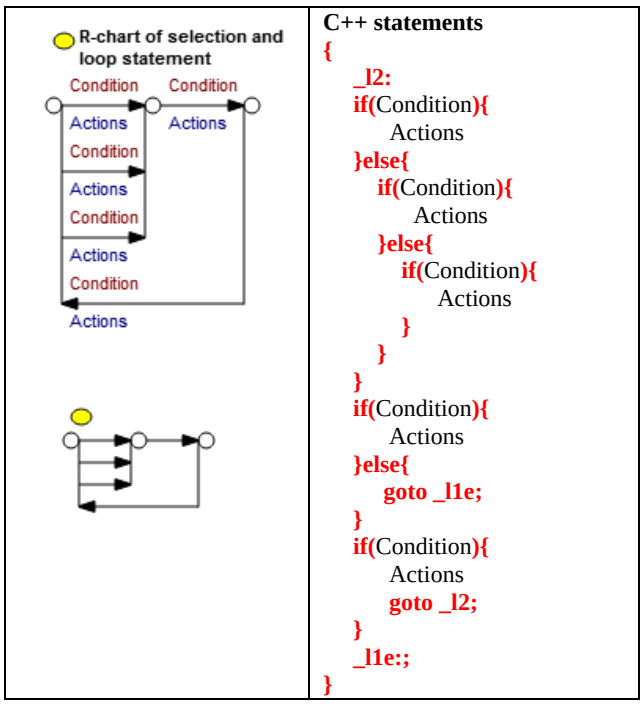


Fig 2. Record of selection and loop statements in R-charts and in C++. Extra symbols for R-chart in C++ are marked by red color.

which is R-chart without implementation details, without records on arcs. This is the new form to write a program, which is absent in traditional programming. It allows visually and very compactly depicting the essence itself, scheme, algorithm skeleton. It is very useful in development of VTR graphical programs.

R-chart in Fig. 2 is incomparably more visual, more compact by two times (R*-chart is more compact by **12** times) than a record of programs in C++ and is input only by **4** (number of horizontal arcs) key presses of mouse or keyboard (by **37** times quicker than input in C++). A half of odd symbols in C++ are excluded from record of R-chart (they are in red color on Fig. 2). A VTR user does not see the right part of Fig. 2, which is shown only as the explanation of R-chart in traditional notations for reader. Language entry on the arcs can be anything: Russian, English, Chinese, Mathematical, Programming (computer sci): Fortran, Cobol, C++, JAVA, Delphi, etc. (**any!**). No restrictions are put on record of Conditions and Actions; it can be in one or some lines [1-7].

II. REALISTIC EXAMPLE

Such graph in Mathematics is called an arc-loaded graph, unlike top-loaded graph (all known Flow charts, UML-diagrams, DRAGON-system, etc. and more than 100 items). A graph in VTR has the name; a program is set by ANY NUMBER of such graphs, interrelated by names. Fig. 3 shows R*-chart for realistic graphical OOP-program, which carries out reading the special type of files in language **Delphi** and is defined by eight R-charts [8]. The first R-chart in Fig. 3 sets the architecture of the whole OPP-project. The first arc in this R-chart sets the name of class: **Type=class.** Five arcs in the second column define FIELDS of class: **TextFile**, **array of string**, **Boolean**, etc.

While setting (describing) data into FIELD of class above on the arc as Condition, we always write the true Boolean constant – keyword (**TextFile**, **array of string**, **Boolean**, etc.), which defines Actions – structure of data under arc.

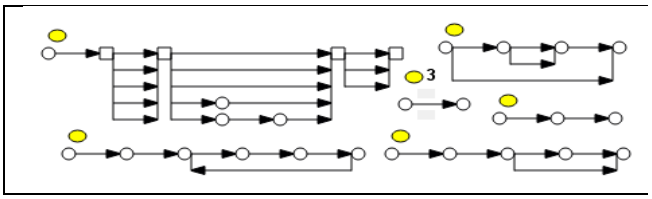
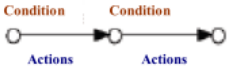


Fig. 3. R*-chart of program without implementation details is **by 45.1 times more compact**

third column describes METHODS, the fourth column – PROPERTIES and METHODS for properties. In total seven methods, which are further defined by other R*-charts on Fig. 3, are set. Three of these methods are set by the simplest one arc.

The program on Fig. 3 has one peculiarity, which is absent in traditional programming – it is the possibility very easily to set the special use and execution of arcs. To this effect it is enough to change the configuration of tops by double click of mouse. Change for square on Fig. 3 – and as a result the single option arcs, equal to the following successive record of arcs: 

, etc., are written in one column. It allows improving the visualization and compactness of record for description of data. Similarly, so easily we may set the parallel execution of arcs, expectation of external interruption, prohibition or permit for interruption, etc. The third column demonstrates the convenient possibility for combination of description with definition, it is absent in traditional programming.

The record of program in Fig. 3 by traditional way in language **Delphi** using **if, for, goto,** etc. statements takes **2.5 pages** of text [8]. R-chart of this program is by **5.3 times** more compact, and R*-chart is by **45.1 times** more compact. **860 (41.3%)** symbols, unnecessary for R-chat, are deleted from program in **Delphi**.

This means as many times increases the visibility, clarity, humanity programs in R-charts and as many times easier and improves the quality of its manufacturing process. Programming are available to all, not only programmers.

RR*-charts on Fig. 3 for the first time (it is absent in traditional programming) are not only the graphical shell of program but the network graph of its development. According to it, the first R-chart on Fig. 3 automatically generates the plan and procedure for definition of all seven OOP-methods. R-chart (arc-loaded graph) – this is a universal way to set information in Mathematics. Historically such graphs (charts) have been used in the *Network planning and management* for a long time, providing with the most visual presentation about progress of work, their availability and completeness, see Fig. 4. The names of works: 01, 11, 12, etc. are written on the arc above such graphs, and below in brackets (see the first arc of R-chart, Fig. 4) – time for their execution. The use of such charts in programming opens the large perspectives for organization of planning industry on development of software projects in the single system of notations, Program objects and Management system of their development.

III. THREE MAIN ADVANTAGES OF NEW TECHNOLOGY

1. Simplicity and incomparable visualization. The important peculiarity of VTR that provided it with simplicity and incomparable visualization is the **exclusion** of statements **if**, **for**, **goto,** etc (**all!** about 10) from programming, which remained unchanged in programming since the end of the 40-ies. These machine-oriented statements and their accompanying symbols-parasites (**then, else, do ";",** etc.), **labels,** brackets of **begin- end**, **{-},** etc. confuse the simple, clear and comprehensible programming processes, are the sources for its difficulties and problems, when as a result "we cannot see a forest behind the trees".

ONE, more powerful and foolproof at a glance, human (not machine) essence (Fig. 1), which reflects the thinking process of human itself (documents his/her thinking process) and connects more powerful visual apparatus of human associative thinking to programming, is introduced

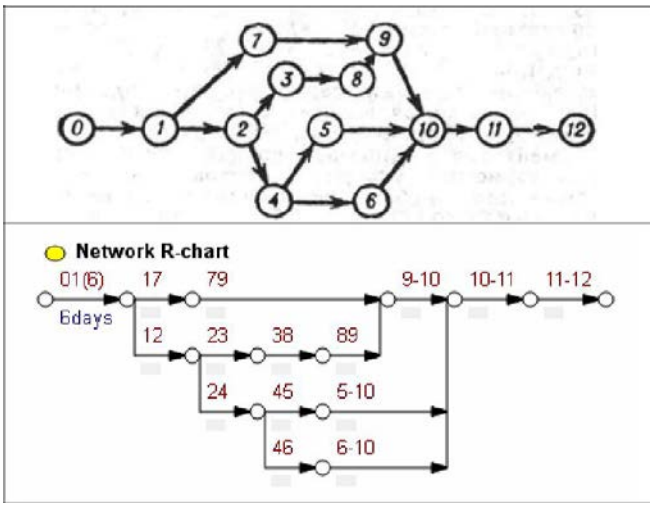


Fig. 4. Record of network graph in R-charts

instead of them. It removes many problems of modern programmers, opens the way for their union and accumulation of professional experience. R-chart of program for the first time coincides with the network graph of its development and automatically generates the action plan for collective of programmers.

2. Compactness is by 50 and more times better in comparison with traditional record of programs in programming languages. This is the most compact form to record programs among all known ones. It allows visually setting any hierarchy (nesting) of delimitations in natural language for each stage of design. Hereby it is important to mention that nesting due to type of nesting of statements in traditional languages is **excluded** and it significantly simplifies understanding a program in R-charts. The main advantage of new graphical R*-chart, which is ABSENT in traditional programming – one can see the chart of the whole project at once, it helps to understand (to tell, to discuss, to protect, to develop, to add, to approve, to forward, etc.) the project and it turned out to be very important in professional programming. The larger a software project is, the higher the compactness of RR*-charts is.

3. Power. One mentioned arc of R-chart is enough to record any algorithm. Practically any language (including natural) may have the mentioned simplest graphical shell, uniform (!) for all languages. It means that all that may EASILY be recorded in their graphical shell, which is more efficient (and thus more compact, simpler translated into computer language and has better characteristics of result computer code) than their traditional record in the form of texts. The contrary is much more difficult or in general is incomprehensible yet as R-charts are more powerful and set many tasks for the first time, there is NO (equivalent) in traditional programming. The fact that one can write the logic of natural texts for the first by a new way using R-charts solves an everlasting unsolved problem CUSTOMER-PROGRAMMER and for the first time provides with program correctness proof.

A tool (R-chart) in the new graphical paradigm is flexibly adapted, developed for the task, being solved, and its executors but the task is not transformed to existing fixed machine language statements (tool) as it happens now in traditional programming. As a result, now a programmer (human) instead of solving his task has to adapt its solution and his thoughts to these machine statements, which do not concern either a human or his task and are absolutely far and are not designed for solution of tasks by human – they are very complex and primitive (low-powered, knotty and overall).

Many things in VTR technology starts from the words FOR THE FIRST TIME and UNLIKE OF traditional programming technology. For example, the reflection of essence for R*-program (algorithm) without implementation details, which is by 50 and more times compact than traditional record of programs, self-descriptiveness of programs, absence of nesting of statements, most compact record and quick input of programs from all known, program correctness proof, three-dimensional and multidimensional programming,

simple and visual setting the parallel execution of programs, meta language for formal setting syntax and semantics of languages, (automatic) generation of action plan for collective of programmers, etc. It is only the top of iceberg of advantages.

IV. COMPARISON WITH OTHER GRAPHICAL PROGRAMMING METHODS

At present time top-loaded Graphs and Visual programming are most popular. The wide use of arc-loaded Graphs in programming is unknown [9-11].

In our opinion, the purpose (function) of the first ones (all known Flow-charts, UML-diagrams, DRAGON-systems, etc. more than 100 items) is somehow to smooth, neutralize the disadvantages (meanness) of textual (antihuman) computer programming statements. It is well known in classical mathematics (theory of programming) that such graphs are not efficient for use in programming and are applied only for modeling, designing, etc. at initial stages of search (constructing) of algorithm in order somehow to humanize even the first contact with computer. In comparison with one arc of R-chart they are significantly more bulky and are not acceptable in all life cycle of programs.

Again as it is known from classical mathematics, arc-loaded graphs (which are R-charts) are used to construct (to study) artificial intelligence systems (intelligence systems), which are closer to essence of programmers' job and, as our not large experience showed, IDEALLY suit for automation (and humanization) of this job.

At present moment the popular term is **Visual programming** as a way to create a program for computer through manipulation of graphical objects (icons, keys, labels, logo-blocks, etc.) instead of writing its text. Recently more attention is paid to visual programming due to development of mobile sensor applications (Pocket PC, pads), where the use of keyboard is not very comfortable. The term "Visual programming" itself is understood by all users wider than it is interpreted by the specialists, who practice it. The dissertation by Jeffrey Nickerson «Visual Programming» #9514409 contains the overall analysis of this direction. The main conclusion of this dissertation is: **"Visual programming languages are not practical, textual representations are more efficient and compact, the future is for hybrid systems".**

This conclusion means for us that OUR visual programming (with R-charts) is inconsistently **better** in comparison with what we have now and it is principal that they much exceed the visual and textual representation due to efficiency and compactness by **50 and more times.**

V. CONCLUSION

Thus, R-charts are the universal graphical shell, being the only one for all languages, including natural and programming languages, which formed as a result of long-term analysis of the existing organization and identify shortcomings of traditional programming. As a result, programming from primitive-artisan has become the mathematical, humane and received the possibility to be developed according to centuries-long principles of mathematics. Hereby it is important that the new graphical concept preserves the succession with what we have and does not reject all previously achieved and programmed, couching them into single, simple, visual and compact graphical shell, which by its simplicity allows for the all of the specialists to program but not just for programmers. Everything is made according to A. Einstein: "do as simple as possible, but not simpler this" because this is easier and more efficient all now.

REFERENCES

[1] V.M. Glushkov and I.V. Velbitskiy, "Programming technology and problems of its automation", *USIM*, Kyiv, no. 6, pp. 75-93, 1976.

[2] I.V. Velbitskiy, "Programming technology", *Technika*, Kyiv, p. 279, 1984.

[3] *Information technology, Program constructs and convention for their Representation*, International standard ISO/IEC 8631, Second edition 1989 Geneva 20, Switzerland, ISO/IEC Copyright Office, p.7, 1989.

[4] W.K. McHenry. "Technology: A soviet visual programming", Journal *of Visual Languages and Computing*, vol.1, no. 2, pp. 199-212, 1990.

[5] L.F. Drobushevich, “Common use of UML and R-chart notations in the training process for software system development methods”, *MEDIAS*-2010, Cyprus, pp.73-77, 2010.

[6] I.V. Velbitskiy, “Next generation visual programming technology with R-charts”, *Plenary report, MEDIAS-2012. Dedicated to 100 anniversary of Alan Turing (IEEE)*, Cyprus, pp. xiv-xxxiv, 2012.

[7] I.V. Velbitskiy, “ Graphical Programming and Program Correctness Proof”, *Plenary report, 9th IEEE Computer Science & Information Technologies Conference*, Yerevan, Armenia, 23-27 Sept.2013, IEEEXplore,CSIT-20.

[8] A.N. Valvachev etc. “Programming in Delphi. Textbook. Chapter 3 Object-oriented programming”, *INTERNET*, p. 19, 2005.

[9] Leonid Cherniak, “SOA – step beyond the horizon. Open systems: IT management”, *Open systems #09/2013*, p.12

[10] Graphical programming based on the weighted graph vertices. http://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B7%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5

[11] Jeffrey Nickerson Dissertations «**Visual Programming**»#9514409 http://c2.com/cgi/wiki?GraphicalProgrammingLanguage

Kiev 120714