Interview of Alex Stepanov by Yuyong Zhao for the Chinese Popular Computer Weekly (www.cpcw.com)

Published 28 February 2005

How did you become involved with computers?

In 1972 a friend of mine recommended me for a job as a computer programmer in one of the Moscow research institutes. They hired me and I have been programming ever since. I was fortunate to join a laboratory that was developing a fault-tolerant minicomputer from scratch. I learned a great deal about architecture, operating systems and other system software while working there. I participated in the design of the OS, wrote a debugger, assembler, and linker-loader. I was fortunate to work closely with Alexander Gurevich, the chief designer of the mini-computer and a wonderful programmer.

What do you teach your children and students?

While I am not a professor now, I do spend a lot of time teaching engineers at Adobe. The course that I teach is based on my belief that programming is a mathematical discipline and I try to show that algorithms are defined on abstract mathematical structures. The books that I recommend to my students are *The Art of Computer Programming* by Donald Knuth, which is the great encyclopedia of programming techniques. I was told that a very good translation of it was published by the Defense Industry Publishing Company in Beijing and I urge all of your readers to buy a copy of all three volumes. It is something that they should keep studying for the rest of their lives. The other book that I urge my students to read is *The Textbook of Algebra* by George Chrystal. It is a massive two volume work covering most of elementary algebra. Sadly enough, nowadays even people with graduate degrees in Mathematics do not know most of the material in Chrystal. I do not know if it is translated into Chinese; if it is not, I would hope that somebody does translate it. Next year I plan to finish a book based on my lectures. Of course, I will be very happy if it is translated into Chinese.

As far as my children are concerned, several of them are programmers. One of my daughters works on compilers and performance tools; another is an engineering manager at large anti-virus company. I, however, did not teach them much of anything. My children seldom listen to me.

What is good way to learn how to program? Is it necessary to learn more than one programming language?

Yes, it is very important to learn several different programming languages. I used Algol-60, Common Lisp, Scheme, Ada, C, C++, Java, and several assembly languages. However, do not concentrate on the language. It is just a tool – and often a very imperfect tool – of expressing algorithms and data structures. As Niklaus Wirth correctly observed: *Programs = Algorithms + Data Structures*.

What are the differences between computer languages and human languages?

Nobody ever managed to write poetry in a computer language. We have a long way to go before programming languages evolve to a point of allowing us to write truly beautiful things in them.

What do you think of OO? Is it a good style of programming? Is there a necessary and useful tool for learning OO?

I try not to think of OO. I am not impressed with their approach to programming. Quoting from my interview to an Italian journal: "I find OOP technically unsound. It attempts to decompose the world in terms of interfaces that vary on a single type. To deal with the real problems you need multisorted algebras – families of interfaces that span multiple types. I find OOP philosophically unsound. It claims that everything is an object. Even if it is true it is not very interesting – saying that everything is an object is saying nothing at all. I find OOP methodologically wrong. It starts with classes. It is as if mathematicians would start with axioms. You do not start with axioms – you start with proofs. Only when you have found a bunch of related proofs, can you come up with axioms. You end with axioms. The same thing is true in programming: you have to start with interesting algorithms. Only when you understand them well, can you come up with an interface that will let them work." I repeat: programming is about algorithms and data structures, not about inheritance and polymorphism.

What is the relationship between Mathematics and Computer Science?

Computer Science is a mathematical discipline. Quoting from Dijkstra: "As soon as programming emerges as a battle against unmastered complexity, it is quite natural that one turns to that mental discipline whose main purpose has been for centuries to apply effective structuring to otherwise unmastered complexity. That mental discipline is more or less familiar to all of us, it is called Mathematics. If we take the existence of the impressive body of Mathematics as the experimental evidence of the opinion that for the human mind the mathematical method is indeed the most effective way to come to grips with complexity, we have no choice any longer: we should reshape our field of programming in such a way that, the mathematician's methods become equally applicable to our programming problems, for there are no other means."

Why did you invent STL? How did it come about? What is the influence of STL on your life and research?

I invented STL after many years of trying to find ways to decompose programs into flexible, parameterized modules. I am still unhappy with it: there are many imperfections caused by my inability to say what I want to say in C++. It should be viewed as a draft of a future library. I may not live long enough to see such a library, but I believe that eventually there will be a really standard way to define algorithms and data structures. Maybe one of your readers will be the designer.

It is hard to say what the influence of STL on my life or research has been. STL did not change my life materially, since I did not bring me any monetary rewards. I wanted to show my friends how to program correctly and, as far as I was concerned, STL was done just for them. The fact that I was able to finish it makes me happy. Sometimes I hear that it helped somebody and it also makes me happy. There are people who are sure they can do a much better job designing a library and they often feel that saying something nasty about STL proves that they can. Public attacks do hurt me.

You have studied in the USSR but now work for an American company. What do you think about the differences between the two countries and their universities? What is a good university education, especially in Computer Science?

I left USSR in 1977 and for years I felt quite at home in the USA. Now, when I am getting old, I often return to the books of my youth: Tolstoy, Dostoevsky, Gogol, Pushkin. And I spend lots of time reading classical books by great mathematicians of the past: Euclid, Gauss, Dirichlet, Euler, Felix Klein. I do find that American Universities, and the American society in general, does not value culture and science as much as it was valued in Russia. It is not valued in Russia now, and I hear that a new generation of Chinese students thinks more about acquiring money than about acquiring knowledge. This is sad.

Could you make any predictions about IT in the future? What programming languages will we be using?

For the next few years you will be using C++ and Java. I do hope that eventually some new, beautiful language will come along. But I do not hope that I will live long enough to see it.

What do you think about China and Chinese programmers? What do you want to say to them?

China is a great country. It produced some great mathematicians in the past: *The Nine Chapters* by Ch'in Chiu-Shao is one of the masterpieces of Ancient Mathematics; and Master Sun' Manual by Sun Tse (Sun Zi) contains a remarkable discovery of what we in the West call Chinese Remainder Algorithm. And in the XXth century China produced some really great mathematicians such as Jing Run Chen with his results on Goldbach conjecture. I hope that Chinese programmers will become as great as Chinese mathematicians. My advice to them is simple: learn Mathematics, learn Computer Science, learn English.

How many times have you visited China? What were your first impressions of China?

This is going to be my first trip to China. I am looking forward to visiting your great and beautiful country. And I hope to get to eat some of the wonderful food and drink some oolong tea.