Milestones for Computing Future [*]

Teodor Rus, Emeritus Professor
The University of Iowa,
Iowa City, IA 72242, USA

November 3, 2015

Vinton Cerf in Communications of the ACM 07/2015, 7 calls for setting anniversary milestones for Computing Future similar to the Hilbert program. At the end of 19 century mathematicians started doubting whether or not some of the problems they were working on for long time do or do not have solutions. At the 1900 International Conference on Mathematics held in Paris, Hilbert proposed 23 unsolved problems as challenge for mathematicians of the 20 century. Contrasting this situation with today computer science (computing?) computer scientists are challenged to find milestones for future computing. However, since computer science has no unanimous accepted definition as a science, one cannot easily identify unsolved computing problems as such challenges. On the other hand, the concept of computing describes an activity that evolved from the very early stages of human society as a mechanism for "problem solving". Of course, originally the problems were very simple, such as counting the number of sheep one had. But with time computing evolved as a branch of mathematics called the theory of algorithms, computing tools evolved from abacus to today computers, and problem solving methodology evolved to computer programming. Hence, perhaps the very first challenge for computing anniversary in 2017 would be to settle for a unanimous accepted definition of computing (or computer) science. Generalizing the concept of computing as it evolved to 21 century, the obvious definition would be:

Computer science is the science of computer-based problem solving of problems arising from any domain of human endevour.

Problems with this definition is the *contradiction between the universal methodology used by current computer-based problem solving process (programming) and the specific methodology used for problem solving in any individual domain of human endevour.*

This contradiction is currently resolved by developing tools (software tools) that allow computer user to formulate the problem using a problem domain specific methodology while computer computes the solution using computer methodology. These tools rely on mappings of domain specific problem expressions into computer language expressions (programs). Since domain specific problem expressions are natural language

[*]An answer to Vinton Cerf's call for Computing Milestones, Communications of the ACM 07/2015, 7

expressions and programs are computer based expressions this methodology does not resolve the fundamental contradiction of today computer based problem solving process: *the programming*.

Programming (or coding) asks computer users to be computer educated in order to use the computer to solve their problems. The deepening of this contradiction is best illustrated by the growing complexity of the software tools which threatens to ”kill computer” (Markoff, 2012) as we know it. Due to knowledge spiral, the number and complexity of new computer applications increases exponentially with the successes of current computer applications and thus leads to exponential increase in complexity of software tools required by current computers-based problem solving process. This increases professional expertise requirements for computer usage putting strong burdens on computer education (Horn, 2001). Computer research seek “revolutionary approaches” to overcome this situation but does not challenge the programming as the computer-based problems solving methodology. By the contrary, by “computer democratization” programming is advocated to be moved lower on the scale of student education. This does not simplify software complexity and does not make easier computer usage for non-computer educated people. In other words, computer democratization is a vacuous expression as long as computer usage as problem solving tool is based on programming. The monsters created by computer democratization exploded recently under the AI panic of creating “killer machines more dangerous than nukes”. Unfortunately this panic is not only promoted by journalists but it is embraced by computer scientists that have left a mark on computer technology and by Nobel price awarded scientists. And it all started with the Turing idea of creating intelligent machines. But Turing created his machine to help him solve problems. So, by creation Turing machine needed to be better than Turing in the problem solving task he created it to help with. Answering the question whether human can create intelligent machines Turing said “if only scientists could discover a mechanical explanation of how analogical thinking works in the human brain, they could program a computer to do the same”. In 1970-s Hubert Dreyfus identified six-levels of skills during problem solving process: beginner, advanced beginner, competent, proficient, expert, and master, where each represents a “higher level of embodiment” than previous. The embodiment means that skills are encapsulated within human body, not just in the brain, through immersive practice, until they become automatic actions performed with no aware of what they are doing (Denning, 2015). Hence, it seems practically impossible for people to describe their action rules they are not aware of.

However, while Turing used his machine as a tool helping his brain solve some given problems, today computer-based problem solving methodology asks the human brain to work as such a machine (computational thinking), which is obviously a gross aberration. Therefore another obvious milestone would be to *create a computer-based problem solving methodology where the computer is used as a human brain assistant instead of using the human brain as a computer assistant*. The successes of computer applications such as iPhone and Internet, which do not require their users to be programmers, tell us that yes, such a technology is feasible. Then, why don’t we try to find it instead of creating panic with threatens posed by AI? The answer is perhaps in the thinking-inertia created by the successes of human thinking process.

Now, let us try to solve the fundamental contradiction of computer-based problem solving differently. For that let us assume that every problem domain is provided with a specific abstract machine, characteristic to the domain, which can interpret computationally the domain concepts. That means every concept of the problem domain is computationally characterized by its natural language term and the computational meaning of that term. For example, a cell in biology will be characterized by the word cell and a data-model of the biological cell expressed in terms of cell components, which in turn are characterized by the natural language terms denoting them and their computation models expressing their meanings. Then a domain statement is seen as a computational model resulted from the composition of the computational models of the terms composing that statement. Hence, a domain expert would use a computer to solve her problems by expressing her solution algorithm in terms of the domain concepts. That is, the domain expert communicates with her machine using the natural language of the domain. By normal education, domain experts integrate within their brains concepts and their meanings. Consequently natural language of the domain is the fragment of natural language spoken by domain experts. Cell in biology is not confused with the cell in telephony or in automata theory, unless the user is neither a biology expert nor an cell phone user. But in that case the term cell has the meaning of the domain the user belongs to. If she has no domain, then the term make no sense. The assumption we are making here is that during the learning process provided by normal school education, the domain concepts are stored both in the human brain (as done by today process of education) and although on a data-carrier specific to the domain, by a process we call Computational Emancipation of Application Domain (CEAD) (Rus, 2015). CEAD-ing a domain is performed by usual educators collaborating with computer scientists. Domain educators provide the tuple $(term, meaning)$ and computer experts provide data-carriers and the tools to record domain concepts and the computer expressions of the computational meaning of the terms thus recorded. The tools (XML, RDF, URI, SPARLQ, etc.) used today to advance Semantic Web can be effectively used for this purpose. That is, during computer-based problem solving process computer user could use the natural language of the domain while programming is performed by computer domain educated programmers. Therefore in any domain of expertise, during problem solving process, the computer can be used as a brain assistant by natural communication, as any other humanly developed tool. The complexity of software tool development and use is factored out from the problem solving process as an activity performed by the computer science domain experts. The process of education is no longer complicated by melanging domain concepts with computer concepts.

Concrete mechanism provided by today computer science for this endevour would be the cloud (Rus and Bui, 2010). Therefore further we assume that a cloud provider provides its users with CEAD-ed problem domains and abstract machines that interpret problem domain statements by performing the computations associated with them using appropriate computer architectures. This approach of computer-based problem solving methodology shows that there should be no fear of loosing jobs by extensive developments in AI. These developments concern the CEAD-ing problem domains by an activity which effectively democratizes the computer thus showing the entire iceberg, not just its tip, where everybody (from childhood to death) will have the usual natural job of learning. The difference is that learning process now is complimented by recording the knowledge not only in the brain, which is perishable, but also on appropriate data carrier supports by the CEAD-ing of the domain, which is persistent. This activity can be carried out by software tools that create domain dedicated virtual machines that perform the brain work better, because that is why they are created for in the first place. And if during this process one can CEAD the process of human brain transformation of electrical signals into language concepts then, yes, AI will create robots that could behave like human Brian. So, yet another computing milestone would be *the development of computer based problem solving methodology where computer is used as a brain tool (thinking with computer's help or thinking computationally) and the software tools supporting it*. This would allow the computer to become a thinking tool integrated within the cognition process (Rus, 2013).

References

Denning, P. (2015). The Profession of IT. *Communications of the ACM*, 58(9):34–36.

Horn, P. (2001). Autonomic computing: IBM's perspective on the state of the information technology. http://www.research.ibm.com/autonomic/manifesto.

Markoff, J. (2012). Killing the computer to save it. *ACM TechNews*, October(31).

Rus, T. (2013). Computer integration within problem solving process. In *Proceedings of RoEduNet 11-th International Conference*.

Rus, T. (2015). *Computer-Based Problem Solving Process*. World Scientific.

Rus, T. and Bui, C. (2010). Software development for non-expert computer users. In *Proceedings of the International Conference on Cloud Computing and Virtualization*, pages 200–207, Management University, Singapore.