The *Communications* website, http://cacm.acm.org, features more than a dozen bloggers in the BLOG@CACM community. In each issue of *Communications*, we'll publish selected posts or excerpts.

twitter
Follow us on Twitter at http://twitter.com/blogCACM

DOI:10.1145/3484986 http://cacm.acm.org/blogs/blog-cacm

Assessing Internet Software Engineering, Encouraging Competitions

Andrei Sukhov considers the potential for reducing international tensions through competitive events, while Vivek S. Buzruk looks at the evolution of teaching Internet software engineering.

Andrei Sukhov Competitions, Not Confrontation

https://bit.ly/3fFRNxV
May 21, 2021

In recent years, there has been an increasing tendency for confrontation between states and ideologies. The institution of sanctions, countersanctions, and the emergence of new hotspots occur with alarming regularity, reminiscent of the worst years of the Cold War. The level of tension is constantly increasing, and the meeting of preconditions for a decrease in the degree of confrontation is not generally within reaching distance. All this has had a negative impact on international scientific projects, including in the field of computer science.

The number of international competitions supported by government research funds has rapidly decreased. New competitions involving collaborations between scientific teams of opposing 'sides' are no longer announced. These factors have led to a gulf between researchers from different countries, which does not contribute to the search for new knowledge or to the acquisition of communication and teamwork skills.

New collaborative research could be the bridge to allow the maintenance and development of communication between countries.

The question then arises, which international institutions are in a position to act as organizers of new competitions involving teams from different countries? Scientific foundations supported and funded by governments are often not well placed to take on this role, while international research communities united by professional interests could be seen as ideally suited to it. Organizations such as ACM and IEEE include in their ranks researchers from a great variety of countries; these researchers successfully interact within the framework of these organizations. The influence of politics on the activities of such communities is still minimal; this allows ACM, for instance, to organize new competitions involving joint research by scientists in countries otherwise at odds.

From where can the necessary funding be sourced? ACM could act as founder of a fund specifically for this purpose. It would have to carry out the necessary preparatory work under its own auspices, but subsequently submit for wider discussion regulations regarding the activities of the fund. To improve transparency, it would be necessary to form a board encompassing all stakeholders on an equal basis. Such a board ideally would include ACM members representative of countries in conflict with each other, on a parity basis—as well as independent members. That is, on the one hand, representatives from the U.S. and other Anglo-Saxon countries; the European Union; and Japan should be included. And, on the other hand, representatives from China and Russia should also be at the table. In addition, largely non-aligned countries such as India, Brazil, Argentina, South Africa, Indonesia, and those of the Middle East should also be represented.

The board members also would have to represent universities, research institutions, and private companies. There should be restrictions regarding the representation of governmental organizations—the fewer involved, the better.

It also would be desirable to reduce the direct contribution of governments to zero (or near zero) in the financial sphere as well; it might be necessary to refuse contributions to the fund by governmental organizations. Contributions from private companies would be preferred, but undue influence by any company would have to be avoided. To achieve this, it would be prudent to limit financial contributions made by any one organization to 10% of the total.

It is imperative that such a foundation be formed and start its activities as soon as possible, to prevent the further isolation of researchers living in different countries. I propose to conduct discussions to encourage comment and feedback, and when/if the idea is approved, to proceed with its practical implementation.

Vivek S. Buzruk
20 Years of 'Software Engineering for Innovative Internet Applications'

https://bit.ly/3ABdB7k

July 6, 2021

Twenty years ago at the Tenth International World Wide Web Conference, Hal Abelson and Philip Greenspun presented a paper on "learnings from teaching a Subject offered at MIT."[1] Its subject was "Software Engineering of Innovative Internet Applications."

I came across the paper in 2006/2007 while browsing the content of a similar subject/course. As a software professional, I was impressed by the focus and intentions of the paper, which emphasized "Engineering Software," while teaching "Development of Innovative Internet applications" using current/emerging technologies.

The paper starts: "Why is software engineering part of the undergraduate computer science curriculum"? Industry expects the software developer to be also a great software engineer. How much does a student really practice software engineering during their undergraduate studies? Even if a student applies/uses software engineering principles during, say, a student project, are they evaluated based on their skills?

The paper suggests that in addition to the core computer science curriculum, we also have to teach students:

- object-oriented design, in which each object is a Web service (distributed computing);
- concurrency and transactions;
- how to build a stateful user experience on top of stateless protocols; and
- about the relational database management system.

In 2021, these views need the attention of both academics and Industry. In the last 20 years,

- Internet Applications have evolved in size, complexity, and availability.
- Underlying front-end/back-end Web technologies and Internet Application execution platforms have undergone multiple changes.
- The evolution of technologies and platforms allowed software professionals to develop and support user-centric, multi-channel, secured, easily adaptable/integrable/accessible, intelligent, scalable, multi-tenant, Geography/Region-agnostic and real 24/7 high-performance Internet applications.

Yet, have they impacted the Vision and Mission of "Software Engineering for/of Innovative Internet Applications"? The title of the paper transcended changes in such technologies and platforms. I am sure content must have been modified every year. More importantly, in addition to technologies and platforms, students who attended this subject/course must have learned real-life situations/challenges, and the subject must have instilled the importance of Modularity, Distribution, Scalability, etc., through course content.

Step Forward: Retrospective Action

When I recently got involved with an institute for its course curriculum of IT undergrads, I remembered Abelson and Greenspun's paper. I recalled their smooth integration of various software engineering methods while teaching a student about building end-to-end (full-stack) Internet applications.

Today, during evaluation of IT undergraduate course curricula, I ask myself, considering that students learn "OO programming in XYZ," "Web Technologies," … "Distributed & Cloud Computing," etc., as independent courses/subjects:

- How can we ensure students will be able to apply a systematic, disciplined, quantifiable approach to the development and maintenance of software using such paradigms and technologies?
- How they will be able to develop production-ready, maintainable software?

The question is more about time-window and habit. Industry will be overjoyed if it finds a Fresher who appreciates software engineering and, to some extent, the assimilated principles of software engineering.

Educational institutes try to match industry expectations and student aspirations within their own constraints. They cannot take responsibility for making every student a great software professional.

To achieve these goals within given constraints, few educational institutes think of logical threads running through their courses. One typical logical thread is continuously improving the student's programming proficiency.

The missing dimension is the engineering approach of applying these isolated learnings to build applications; that is, students do not appear to learn "Software Engineering for XYZ Applications." They are not evaluated on their software engineering approach during problem solving.

In fact, this small step of the software engineering dimension while educating students on various programming languages and new technology areas will make a big difference for students and industry.

In brief, Abelson and Greenspun's paper needs attention from industry and Academics, who need to think about:

- What should be the Vision and Agenda in 2021?
- What, in addition to the core computer science curriculum, should academics be teaching IT undergraduates for "engineering innovative Information Systems of the next 20 years"?

Reference

1. Teaching Software Engineering—lessons from MIT: by Hal Abelson and Philip Greenspun. Presented at the Tenth International World Wide Web Conference (Hong Kong), May 1–5, 2001; https://bit.ly/3hce6gn

Andrei Sukhov is a professor and head of the Network Security Research and Study Group of HSE University, Moscow, Russia. **Vivek S. Buzruk** is a DevOps consultant at SarvaTech Consultants Inc. India, and a 'Board of Studies' member for the IT Dept. of Pimpri Chinchwad College of Engineering (PCCoE) in Pune, India, mainly focusing on DevOps insights and custom software development.