The Software and Information Systems Department

Achi Racov Engineering School

Kinneret College on the Sea of Galilee

The Third Kinneret Conference on Software Engineering Education

February 17th, 2015

PRESENTATION ABSTRACTS

Guest Keynotes

Graduate attributes: what competencies our students need to develop, beyond the technical ones?

Philippe Kruchten, Ph.D., P.Eng.
University of British Columbia, Vancouver, Canada
pbk@ece.ubc.ca

For young engineers to be able to perform in today's world, they need to develop a palette of competencies going way beyond mathematics, physics, and the mastery of various technologies. Over the last few years, a range of such competencies have been identified, ranging from communication, teamwork, ethics, professionalism, understanding of social and environmental issues, ethics, life-long learning, etc. Sometimes they are pejoratively dismissed as "soft skills", and we often assume that they'll just be learned as they go, "on the job". But nowadays they have become an integral part of accredited engineering programs in several countries. *Graduate attributes* form a set of individually-assessable outcomes that are indicative of a graduate's potential competency. In this talk, I will describe the set of graduate attributes defined by EngineersCanada for accreditation purpose, and show how they can be objectively assessed throughout the curriculum. This emphasis on non-technical attributes is also aligned with the process of licensing professional engineers, based on competency assessment frameworks. In the case of *software* engineering —an emerging engineering discipline— this whole process runs into additional challenges: should we license software engineers as we do for, say, electrical or civil engineering? This even leads to more fundamental questions, such as: is software development truly a form of engineering?

Philippe Kruchten *is professor of software engineering in the department of electrical and computer engineering of the University of British Columbia in Vancouver, Canada. He holds an NSERC chair in design engineering. Prior to joining academia in 2004, he was a software architect with Rational Software (now part of IBM) for 17 years, where he led the development of the Rational Unified Process®. His experience is mostly in large software-intensive system in telecommunication, defense and aerospace.*

Computer science and software engineering: Draft report of a special committee of the CHE

Prof. Dror Feitelson, the Hebrew University in Jerusalem
and the committee chair

feit@cs.huji.ac.il

In 2014 the Israeli Council for Higher Education (CHE) formed a special committee to define the fields of computer engineering, electrical and electronic engineering, software engineering, and computer science, and to set guidelines for study programs in these fields. To perform this mission the committee collected data about existing study programs in Israel and about similar efforts in the world. The lecture will describe the recommendations and main findings with emphasis on the distinction between computer science and software engineering. In particular, we will focus on the academic aspects of software engineering, on the engineering aspects of software engineering and the connection to vocational training, and on human and social aspects of software engineering.

Note: At this time the report has not yet been approved and adopted by the CHE.

Dror Feitelson *is professor of computer science at the Hebrew University of Jerusalem, where he had also studies for all three degrees. For many years he has worked on performance evaluation, and especially on parallel system scheduling and the modeling of workloads on such systems; his book on workload modeling is to be published this year. Recently he has also started to work on software engineering, and specifically on factors affecting the difficulty to understand software. The common factor in all his work is the empirical aspect, and the emphasis on using real-life data.*

Morning Session 1 (AM1)

Advanced Approaches and Methods

Chair: Dr. Arnon Sturm, Ben Gurion University

Teaching Data Mining through Games

Dan Ophir, Ph,D., Department of Computer Science and Mathematics,
Ariel University Center of Samaria, P. O. Box. 3, 44837, Israel
dano@ariel.ac.il

Data Mining – a relatively new field in ***Computer Science*** that owes its growing importance to the exponentially increasing information from the Internet media, by increasing access to this information.

The ***Data Mining's*** principle is looking for some common denominator among a huge collection of elements or finding an extraordinary element among elements in its environmental elements, this principle is taught through specific games.

In order to emphasize the meaning of ***Data Mining,*** a new term is introduced: ***Inverse Data Mining***, which refers to mixing meaningful data in a sophisticated way – coding it.

The concept of ***Data Mining Visualization*** is demonstrated by generating a set of three images and requiring one to indicate the extraordinary image and to formalize the property that makes the image extraordinary. Another introduced term is the ***Common Denominator Degree,*** which emphasizes the complexity of a logical comparison between the three pictures in finding a common denominator of the properties of the picture's elements. The normalized and computerized IQ tests may be a byproduct of the above concept. Several examples of using ***Data Mining*** methods are from the commercial and biological realms - searching for significant patterns of DNA sequences in the genome.

Inverse Data Mining is demonstrated by showing the use of steganography methodology.

Dr. Dan Ophir *holds a Ph.D. Degree from the Math and CS department, Weitzman Institute of Science, and a B.Sc. from the Applied Math department in the Technion. He is a senior lecturer and researcher in Ariel University and in Afeka College. Dan was a consultant and a partner developer in HiTec companies and in defense companies. He is the author of scientific papers and he participated in many international scientific conferences in the topics of algorithmics and mathematics.*

Using Reference-based Framework for Improving the Comprehension of Software Engineering Principles

Arnon Sturm, Oded Kramer

Information Systems Engineering, Ben-Gurion University of the Negev

sturm@bgu.ac.il, odedkr@bgu.ac.il

To better implement software engineering principles, students need to be provided with the theory and examples. However, it seems that moving for the next step of applying these principles, students still have difficulties. To bridge this gap, we propose to adopt a reference-based framework that guides students with such principles, as well as validating their applications. In this work, we evaluate such an approach which relies on a domain engineering approach, called Application-based DOmain Modeling (ADOM), in the context of the programming task with Java, and thus termed ADOM-Java, for improving productivity in terms of code quality and development time. To achieve that objective we have qualitatively evaluate the approach using questionnaires followed by a text analysis procedure. We also set a controlled experiment in which 50 undergraduate students performed a Java-based programming task using either ADOM-Java or Java alone (i.e., without guidance). The qualitative evaluation reveals that the approach is easy to use and provides valuable guidance. The results of the experiment indicate that the approach is applicable and that the students that used ADOM-Java achieved better code quality, as well as better functionality, and within less time than the students who used Java. The results of the experiments imply that by providing a code base equipped with guidelines for programmers can increase programming productivity in terms of quality and development time. These guidelines may also enforce coding standards, architectural design, and other software engineering principles.

Dr. Arnon Sturm is a faculty member at Ben-Gurion University. He received his Ph.D. in Information Management Engineering from the Technion, Israel Institute of Technology. His research interests include modeling, domain engineering, agent-oriented software engineering, and system development processes. Prior to his studies, Arnon has gained extensive experience in developing software systems in industry.

Oded Kramer is currently a software developer. He received his both B.Sc. and M.Sc. in Information Systems Engineering from Ben-Gurion University of the Negev. His professional interests include design principals and patterns, software productivity, and Java related technologies.

Lessons From Teaching a Software Engineering for Cloud Computing Course - From Class to MOOC to SPOC

Reuven Yagel
Software Engineering Department, Azrieli - College of Engineering, Jerusalem, 9103501, Israel
robi@jce.ac.il

This is a report about a course which integrates classical teaching in class with materials of an online popular course. The course is based on an advanced software engineering course and a book which was transformed to the Edx MOOC platform: "Engineering Software as a Service: An Agile Approach Using Cloud Computing" by Armando Fox and David Patterson from the University of California at Berkeley and Samuel Joseph from Hawaii Pacific University.

In the last two years we joined a program in which the course content was packed for our settings as a Small Private Online Course – SPOC, a term by the authors.

The course itself combines a lot of software engineering content, e.g., development processes, software as a service and cloud computing, tools, practices, version control and deployment, testing and executable specifications, legacy code and maintenance, team work, project management and more. The students complete in one semester a two-part online course and also a project for external non-profit customer.

I'll shorty review the course content and also report on the actual integration in class, the online tools, the project, the way this course addresses industry requirements and latest ACM/IEEE software engineering curricula changes. I'll also talk about some student experience, interaction with other lecturers worldwide, challenges and conclusions to date.

Course materials, tutorials and report papers can be found at http://www.saasbook.info/

Reuven Yagel *is a senior lecturer of software engineering at Azrieli College of Engineering in Jerusalem. He also has some industry experience.*

Morning Session 2 (AM2)

Team and Project Work

Chair: Prof. Shlomo Mark, Shamoon College of Engineering

An Innovative Mobile Device Programming Based Model for Teaching Software Engineering

Guy Leshem and Michal Chalamish, Ashkelon Academic College

leshemg@cs.bgu.ac.il, chalm@acad.ash-college.ac.il

Software Engineering is taught in Ashkelon Academic College as a specialization area, which includes 2 courses and a seminar. The first course focuses on object-oriented analysis and design, based on UML and the"Visual Paradigm". The second course focuses on the development process *from idea to product* based on mobile device programming, that is, the development of mobile applications for cellular devices and tablets. The seminar combines lectures by industry people and by the students, who present new articles published in international journals. This combination seems to be an optimal model for training students in both theory and practice. It also provides them with an industry-attractive capability for cellular and tablet application development.

In order to imitate the real world of software engineering and development, the students' projects focus on the concept of *from idea to product* on relatively small projects. In recent years, the students developed applications such as "Sending fax from mobile phone and tablet", "Quick editing of journalistic reporting from the field" and more.

In 2014, two students developed the "Social Worker" application for a tablet with an Android operating system for Dr. Yaron Yagil of the Department of Social Work, Academic College of Ashkelon. The "Social Worker" is a system for the bio-psychosocial documentation of patients, which is crucial in any organization employing a social worker. The system is designed to support the individual social worker as well as the entire organization by collecting data, identifying trends and long-term planning. The application was presented to Mr. Yekutiel Zabe a Senior Director for Research, Planning and Training in the Ministry of Social Affairs and a pilot is planned to begin shortly, hosted by the Ministry of Welfare.

In 2014 the seminar hosted Dr. Barak Chizi of Deutsche Telekom who talked about working under SCRUM. The students presented a variety of papers published in IEEE Software 2010 and 2011, for example, "Object oriented parallelization of Java desktop programs".

Dr. Guy Leshem is a lecturer in the Department of Computer Science, Ashkelon Academic College, and head of practical software engineering at Ashkelon Regional College. His research interests are in applied statistics, machine learning, secured communication protocols for distributed network of sensors, and cyber security. He has been teaching the Software Engineering course at Ashkelon Academic College for the past 5 years.

Dr. Michal Chalamish is a lecturer in the Department of Computer Science at Ashkelon Academic College. Her research interests focus mainly on Artificial Intelligence and Law, Multi Agent Systems and Machine Learning. She has been teaching the Software Engineering course in the School of Engineering at Bar Ilan University and the Seminar in Software Engineering at Ashkelon Academic College for the past 4 years.

Transdisciplinary Development Teams: The Case of Website Development Workshop

Meira Levy[1], Yaron Shlomi[1], Yuval Etzioni[2]
[1]Department of Industrial Engineering and Management,
[2]Department of Textile Design,
Shenkar College of Engineering and Design, 12 Anna Frank St. Ramat-Gan 52526,Israel
{lmeira, yshlomi, yetzioni}@shenkar.ac.il

Software engineers create designed artifacts, therefore they are required to have various talents such as ability to represent and conceptualize "wicked problems"; problem solving; effective use of information; creativity; and adept and apply former experiences. Software design encompasses both implementation of traditional knowledge such as the designer's former goals, ideas and solutions, as well as the development of innovative solutions for unfamiliar problems. In particular, software innovation is important for companies operating in the global economy. Past research pointed at the inefficacy of software development teams, in particular in dealing with business changes and shifting demands. Agile software development aims at solving such problems towards delivery of innovative, high-quality solutions in short time, to an increasingly demanding and dynamic market. However, the agile manifesto does not address people innovation which requires changing "mental models" and developing new perspectives for novel possibilities and solutions. Our research aims at facilitating people innovation by creating transdisciplinary development teams of engineers and designers.

Designers are often characterized by their creativity, whereas engineers are often characterized by their analytical thinking style. Both capabilities are vital in the new economic era, particularly for innovation. Thus, the collaboration between designers and engineers is crucial and should be fostered by academia and industry. Our preliminary research examined perceptions and interactions among 14 students in engineering, art and design programs during a website development workshop. The research indicates that students enjoyed the experience and found it educational and socially engaging. The data collected through questionnaires and observations were qualitatively analyzed, revealing that students learned each other's jargon, conducted discussions that addressed both technological and design concerns and considered various opinions leading to joint decision making. This study sheds light on the potential of transdisciplinary teams in development processes in general and in website development processes in particular.

Meira Levy *is a senior lecturer at the department of Industrial Engineering and Management, Shenkar College of Engineering and Design, having completed doctoral studies at the Technion, and*

a postdoctoral Fellowship at the department of Industrial Engineering and Management, and at Deutsche Telekom Laboratories, Ben-Gurion University of the Negev. Her research interests include knowledge engineering and management, both from human and technological perspectives; social networks and knowledge sharing; and transdisciplinary teams of engineers and designers in development processes.

Yaron Shlomi *is a lecturer at Shenkar's department of Management and Industrial Engineering. He completed his doctoral and postdoctoral training in cognitive psychology in the University of Maryland. His research interests include production and interpretation of subjective probability forecasts, and fostering transdisciplinary collaboration.*

Yuval Etzioni *is a lecturer at the textile design department at Shenkar College of Engineering & Design. Etzioni is a self-employed textile designer and artist participating in commercial design and art exhibitions such as Heimtextile annual trade fair in Frankfurt and Contemporary Textile exhibition in Israel. She graduated the Textile Design department at Shenkar and Art Education Studies-with Senior Teacher certificate at Oranim Academic College of Education, and currently is writing her final thesis in the M.Ed. Studies for Visual Literacy at Seminar Hakibuzim.*

Information System student's engagement in final project: complexity, innovativeness and programming environments

Lavy, Ilana
Management Information Systems, Max Stern Yezreel Valley College, Israel
ilanal@yvc.ac.il

Rashkovits, Rami
Management Information Systems, Max Stern Yezreel Valley College, Israel
ramir@yvc.ac.il

Most Information Systems (IS) and Software Engineering (SE) programs include a capstone course (final project) in their curriculum at the last year of studies, in which the students design and implement a complete software system. The project is defined as a team effort that allows an opportunity to practice personal and interdependence skills to ensure team members empowerment and success. The project has a structured framework that follows the life-cycle of a software product, aiming to facilitate the student's progress along the project timeline. The students have to submit an initiation document followed by a system specification document, and then they have to implement a complete software system according to the handed specifications. The initiation document includes a preliminary description of the system, its users, its goals and feasibility tests. The design document includes a list of customers' requirements and a conceptual design of the planned system. The implementation phase includes the development of software in some development environments (e.g., tools, language, libraries, database, etc.) along with user and maintenance guides.

In this study we explore students' decisions regarding the following aspects:

> (1) What were the expectations of the students from the final project and how they affected their choice of the project's innovativeness and complexity?
>
> (2) What were the underlying reasons for choosing the programming environments used to implement the software?

Data was gathered from projects' documentation; a questionnaire handed to the entire study participants, and from in-depth interviews conducted with representative group of the study participants. Analysis of the data revealed that as expected, high achievers tend to develop more complex and innovative projects than low achievers, and they tend to use new and innovative technologies that require significant time investments in extension of knowledge gained during studies. On the other hand, low achievers tend to develop simplistic systems, and use merely programming environments they were already familiar with.

Surprisingly, some of the average and low achievers and none of the high achievers tended to use programming environments that were new to them. They did so mainly because they wanted to prove self-learning capabilities in order to impress potential employers, and achieve knowledge in additional programming environments.

Prof. Ilana Lavy *is an Associate Professor with tenure at the Academic College of Yezreel Valley and head of the Management Information Systems department. Her research interests are in the field of pre service and mathematics teachers' professional development as well as the acquisition and understanding of mathematical and computer science concepts.*

Dr. Rami Rashkovits *is a Lecturer at the Academic College of Yezreel Valley in the department of Management Information Systems. His research interests are in the fields of content delivery in wide area networks, and the perception of object oriented concepts.*

Afternoon Session 1 (PM1)

Human and Social Aspects

Chair: Dr. Ariel Frank, Bar Ilan University

Hard facts about soft skills

Lavy, Ilana
Management Information Systems, Max Stern Yezreel Valley College, Israel
ilanal@yvc.ac.il
Yadin, Aharon
Management Information Systems, Max Stern Yezreel Valley College, Israel
aharony@yvc.ac.il

This paper provides practical, real world evidence for the importance of soft skills. The term, soft skills, is a general definition associated with a variety of personality traits addressing social, personal and job related capabilities.

Recent developments in the modern workplace have increased the perceived importance of soft skills for transforming the IT graduates into effective and successful employees and managers. Several existing and emerging trends, i.e. globalization, team based product development, the move to service based infrastructures and streamlining social networks into business practices imply that soft skills are important not just for entry level employees but for experienced professionals as it was revealed in a poll conducted in 2008 by the American Society for Human Resource Management (SHRM). An additional report by the Indiana Business Research Center (IBRC) concluded that although technical credentials are important, the social soft skills are critical for developing a strong and effective workplace. Many researchers agree that soft skills not only predict success in life, but also help attaining that success.

The current paper describes an ongoing study for assessing the IT (Information Technology) professionals' soft skills required by the Israeli market. The study that started in 2012 analyzes 2000 IT classified advertisements each year. The required soft skills attributes were clustered into four categories (human interactions, task interactions, organizational interactions and common skills) and their relative importance was calculated based on their number of appearances. During the past three years the human interactions appeared in 36-41% of the ads, task interactions in 22-30%, organizational interactions in 16-19% and common skills in 16-20%.

The paper concludes with some practical ways for incorporating soft skills into some of the academic courses.

Prof. Ilana Lavy *is an Associate Professor with tenure at the Academic College of Yezreel Valley and head of the Management Information Systems department. Her research interests are in the field of pre service and mathematics teachers' professional development as well as the acquisition and understanding of mathematical and computer science concepts.*

Dr. Aharon Yadin *is a Senior Lecturer at the Academic College Yezreel Valley in the Management Information Systems Department. Prior to entering the academic world, Aharon worked in the High Tech industry. He has over 40 years of IT experience.*

Responsible Software Development

Ariel J. Frank, Department of Computer Science, Bar-Ilan University

ariel@cs.biu.ac.il

Responsible Software Development – what does it mean? In a nutshell, people who write and develop software should have a clear and integrative sense of responsibility for the social, economic, legal and ethical implications of the software package they develop, understanding of the need for V&V (Verification & Validation) of both the developed and related software, and of the importance of the reliability of software running in the field. Notwithstanding the attempts to use relevant Software Engineering models, methodologies and tools, such as methods for secure/safe programming, tools for formal provable correctness, and pedagogies of ethical practice, the general state of Responsible Software Development is not encouraging.

However, attempts to bring about Responsible Software Development, in its various contexts, are on the rise. For example, the use of programming environments that enable immediate visualization of the results of executing a program that is currently under development (http://worrydream.com). For V&V there is rising awareness of the need to develop user-friendly interactive tools, unlike those currently available, to enable proofs of correctness or at least to negate the existence of counter examples. For ethical software development there are attempts to develop pedagogies of ethical practice (http://pages.stolaf.edu/csci-263-2014/) that take into account varied aspects such as security, safety, privacy, intellectual property, and social justice.

To empower Responsible Software Development there is of course a need to systematically educate and update all stakeholders. The increasing efforts to regulate and license software engineers can have a significant contribution to that. An additional direction can be to disallow sale of software packages without a warranty deed and a standard stamp of approval, while exerting in parallel the required legal and economic pressures on software houses and suppliers.

Dr. Ariel Frank *is a member of the Department of Computer Science, Faculty of Exact Sciences, Bar-Ilan University, Israel, since 1984. He has served as deputy chairperson of his department for 18 years (non-consecutive). He received his undergraduate, master and doctoral degrees in Computer Science from Bar-Ilan University (Israel), Weizmann Institute (Israel), and SUNY at Stony Brook (USA), respectively. Ariel has supervised several Doctorates, over fifty Master theses, and hundreds of undergraduate and graduate projects.*

Instructing web applications course using an evolving project and online assessment tools

Naomi Unkelos-Shpigel

Naomi.u.s.se@gmail.com

When I lectured a web applications course , whose goal is to expose the second year IS students to web environment development , I chose to combine several interesting elements which I find interesting for other lecturers who teach similar courses.

The course included the requirement to develop in html and then some web development language.

I chose a story which accompanied all the course assignments, and I divided the students into groups of four. The story included building a frame for submission of academic papers in groups, and has them reviewed by the instructor.

First, the students built personal html pages, which included basic information about them, and another page which included information about the group.

Next, after studying the elements of user interface design, the students learned ASP.Net environment, and now they could build more advanced pages, which include elements of programming in C#.Net, and LINQ.

At this stage, students were asked to perform a short think aloud experiment, to check user response to their site. They were asked to improve the site after receiving feedback from their peers and me.

Finally, each group presented their site in front of the other students in class. In addition, each student was asked to assess each group`s site, rewarded extra credit. Most students completed this assessment, providing interesting and enriching feedback.

I will present the course work process, challenges and feedback from students, as well as the interesting insights from the feedback I received at the last lecture.

Naomi Unkelos-Shpigel *is a PhD student at the IS department, University of Haifa, advised by Dr. Irit Hadar. She lectures at Kinneret College, YVC (Yizreel Valley College), and the Technion continuing studies department. She has worked for several years in real-time programming and in Information systems design and implementation. Naomi graduated with honor BSc in Information Systems at the faculty of Computer science, Technion. She graduated her Master's degree at the IS department, University of Haifa.*

Afternoon Session 2 (PM2)

SW Engineering - From Theory to Practice

Chair: Ida Vernikovsky, HP Software

Software Engineering in a Changing and Growing Business

Ida Vernikovsky, Director of SaaS RnD

HP Software

ida.vernikovsky@hp.com

Is it possible to perform a radical business transformation from heavy legacy to an agile modernized business with no interruption to a 24\7, $79M business, which is growing by 50% YOY? YES IT IS

How we took an immature small organization unit (SaaS RnD), Built with old technologies stuck (struts1, jsp), with long release cycle (~year and a half), old architecture and methodologies and transformed it to a successful modernize organization (Agile, Kanban, CI CD, technology stuck and architecture) in order to support a huge business growth in two years.

From hundreds of daily users to about 15,000, from 5 to 50 employees, from 500 to 1000 (enterprise) customers.

All on the fly and almost without any business interruption.

How to perform this scale of transition, keep alive and even success in it?

Our success is in hands of our employees, YOUR STUDENTS.

Ida Vernikovsky *started her career at Mercury, where she absorbed the start up culture. Today she is Executive Manager in HP Software, with vast experience in leading RnD organizations in Enterprise companies. In HP she leads different RnD groups: Shared Platform Organization, Solutions & Integrations and Software as a Service RnD.*

Long-Term Career Development of Software Programmers (In Applicable R&D)

Dror Ben Ami

Zefat Academic College, Zefat, Israel

drorb2@Zefat.ac.il , Dbenami84@gmail.com

High Performance Computing (HPC) paradigms emerged significantly during the last years. Various types of problems require efficient techniques and methods to solve and support organizations with Knowledge. The request for "Efficiency" becomes much more relevant when we concentrate on Artificial Intelligence (AI) and Data Mining (DM) research fields.
It could take **months** of non-stop execution of micro/mini/supercomputers to carry these kinds of computations till completion. Biology, neural-networks, fuzzy-logic based apps, medical implemental research, networks and more – are just part of the fields, which are based on these algorithms.
Thus, the market demands are focused on "*head-hunting*" the BEST-programmers. This Human Resource requires specific-targeted preparation and training for *effective* and *efficient* development.

Practically, it takes few years of preparation and training the best Human-Resources to achieve this professionalism.
What is the profile of "BEST" programmer on one hand, and what are the AI/DM attributes, on the second hand? The research examines the CSFs and offers some ways, directions and highlights, to help companies to take part in the long-continuous process of the HR training and as long-term career development.
As part of the main research findings, two main professional cognitive factors would be specified and considered. These are the 'core' competencies which "build" AI/DM BEST-programmer:
(i) Mathematics wide knowledge (ii) High-level programming competencies
In addition, few arguments are examined, such as: Most of the leading algorithms on AI/DM are based on JAVA; BUT - a lot of the AI/DM algorithms are NOT written effectively. WHY?

Long-term career development can ensure to have experts and brilliant programmers and developers in these fields. Practical ways to achieve this goal would be proposed:
Establish professional relationships between Industry and Academy, through self & organizational trainings.

Keywords: High Performance Computing (HPC), CSF (Critical Success Factors), Efficiency, Data Mining (DM), Artificial Intelligence (AI)

Dror Ben-Ami *is in the last year of his PhD. Research, Informatics (Mathematics & computer-sciences) at Moldova State University. He has more than 20 years of experience as CTO and programmer of BI/AI/DM/DSS. Currently he is a lecturer at Zefat Academic College(Israel) and is involved in a few implemental research fields, such as Medicine and E-Learning adaptive systems.*
https://sites.google.com/site/drorbenamiwebsite

Teaching Formal Specification to Information Systems Practitioners: Challenges and Opportunities

Eitan Farchi, IBM Research, Haifa

Anna Zamansky, Information Systems Department, University of Haifa

farchi@il.ibm.com, annazam@is.haifa.ac.il

It is widely acknowledged that the field of software engineering (SE) can benefit greatly from the use of formal methods. Formal methods have been applied in a wide variety of settings, such as the design, specification, verification and testing of software systems with a notable number of industrial success stories. However, the fact remains that the primary proponents of systematic application of formal development have been academicians and those working in the areas of secure and safety-critical systems, and there is still much debate on their practicality to a wide range application in industry.

One of the reasons for this is the lack of widely-accepted, scientific underpinnings of the young discipline of formal methods in SE. Another concern is the ability of software engineers to overcome the mathematical barriers in formal specification, which is acknowledged as a limiting factor in the acceptance of formal methods by the industrial community. These problems present a real challenge for software engineering educators, and there has been much debate in the SE education community on *what* tecniques of formal methods to teach, and *how* to teach them.

In this talk we report our experience in teaching the course "Logic and Formal Specification" to graduate students at the Information Systems (IS) department at the University of Haifa. We believe this experience provides insights into dealing with barriers in the acceptance of formal methods due to lack of mathematical skills and background. In particular, our course design aims at teaching useful principles and logical thinking while minimizing the cognitive load of the students in the following two ways:

- **Careful topic selection**
  Since the undergraduate curriculum of the IS department does not include a course on formal logic**,** a brief coverage of selected basic topics in Formal Logic is absolutely required before proceeding to any topic in Formal Methods. Accordingly, the first part of the course covers the basics of propositional and first-order logic. However, at the expense of under-emphasizing such standard topics as compactness and completeness theorem, we put a special focus on mathematical induction and its applications. We believe understanding induction is essential for reasoning about specifications. Moreover, a basis in structured induction and its relation to fixed points serve as an essential background to other formal methods used in model checking and abstract interpretation for hardware verification and program analysis.

- **Simplifying complex notations**
  The second part of the course is dedicated to formal specification using the Z notation. From our experience, SE practitioners do not need to master the whole range of rich expressivity offered by the Z notation in order to apply it in practice: a limited scope of basic notations suffices. Moreover, complex notations are perceived as difficult and the students tend to develop a negative perception -- and in some cases even fear of applying formal specification methods in general. Accordingly, our idea in designing this part of the course is teaching the Z notation using industry-inspired, yet very "simple" examples of Z specifications**.** We provide a concrete algorithmic

method for measuring the "simplicity" of Z specifications based on their parsing trees. Using such "simple" examples, the students learn how to understand a formal specification and translate it to natural language, as well as how to work out a specification from a natural language description. Such simple notation does not inhibit, however, the students' formal reasoning about the specification using structured induction, which is is an important objective of the course.

Dr. Eitan Farchi *is the manager of the Software Performance and Quality research group at the IBM Haifa Research Laboratory, and a member of the IBM corporate Board of Software Quality. He has been appointed to the role of IBM Senior Technical Staff Member (STSM) for his many contributions in the fields of software quality and reliability. Dr. Farchi holds B.Sc., M.Sc. degree in mathematics from the Hebrew University and a PhD degree in mathematics from the Haifa University, where he also holds a teaching position. He is the author of more than 80 papers and numerous patents, and routinely provides consulting on software and service quality to IBM customers and business units across Europe and globally.*

Dr. Anna Zamansky *is a senior lecturer at the Information Systems Department at the University of Haifa. An alumna of the Technion Excellence Program, she holds B.A and M.Sc degrees from the Technion and a PhD from Tel Aviv University in Computer Science. She has spent two years as a Marie Curie Postdoctoral Fellow at the Computational Logic Group at Vienna University of Technology. Her research interests include Applied Logic for Information Systems and Computer Science. She has published over 40 papers in international peer-reviewed journals and conference proceedings. Anna has more than ten years of experience in designing and teaching courses in Logic and Formal Methods at Tel Aviv University, Academic College of Tel Aviv-Yaffo, Jerusalem College of Engineering and University of Haifa.*