SKY'2016 – Intensive School Day on Software Knowledge

Thursday, 10/November/2016

Overview - The main theme of the SKY'2016 Intensive School on Software Engineering is "Software Infrastructure for Software Knowledge". The four lectures are dedicated to the following topics:

1. **AMG** = Algebraic Multi-Grid – It assumes that any software system is a multi-level hierarchy. AMG is a technique to obtain the coarser higher levels, given a fine lower-level.
2. **FCA** = Formal Concept Analysis – Is a mathematical theory of lattices of formal concepts inspired by natural concepts. It offers us a generic approach to mine regularities in software.
3. **Requirements in Systems Engineering** – The talk deals with precise techniques to improve the quality of system requirements, among others to reduce ambiguity.
4. **Conceptual Integrity** – A fundamental property of well-designed software systems, its characterization is discussed as well as how to achieve it.

Detailed Program

9:00 – 9:45

Radel Ben-Av - Azrieli College of Engineering, Jerusalem, Israel

Tutorial about AMG and Possible Applications to Software Knowledge and Software Engineering

ABSTRACT

Algebraic MultiGrid (AMG) is a method that is used for solving a large set of linear equations usually represented by a very sparse Matrix. The method is a generalization of the Geometrical Multi Grid. The main idea behind the method is that for many systems the many equations are a representation of a "fine scale" whereas a much smaller number of degrees of freedom can represent the large scale. The AMG algorithm is based on finding in an hierarchical manner coarse degrees of freedom that represent more fine degrees of freedom. Structures that appear naturally in Software and Knowledge (SK) and Software Engineering (SE) are also represented as very sparse matrices – they usually arise as an Adjacency matrix of graphs with a large number of nodes and a relatively small number of neighbors per node. It is rather interesting to investigate what kinds of "coarse degrees of freedom" arise when the AMG methods are applied to graphs of SK and SE domain.
In this tutorial, I will give a brief background of the AMG methods. I will then describe in detail the main building blocks of the algorithm (the *coloring scheme*, the *Interpolation Operator* and the Coarsening Operator). I will then describe various cases where the concepts can be applied as well as possible obstacles for the implementation in the SK and SE domain.

SHORT BIO

Radel Ben-Av earned his Ph.D. in Physics and Applied Mathematics from the Weizmann Institute of Science, Rehovot, Israel. Then he moved to Princeton University, New Jersey, USA, for his PostDoc. Afterwards he worked in Israel High-Tech industries for about 15 years in various positions in Software Engineering and Physics. He is teaching at the Azrieli College of Engineering since 2005 in the Software Engineering Dept. His research area includes the use of Algebraic MultiGrid for classification and clustering purposes and Quantum Information algorithms.

9:45 – 10:30

Daniel Speicher - University of Bonn, Germany

Mining Source Code Regularities with FCA and Association Rules

ABSTRACT

Software encodes knowledge about the application domain as well as about the solution domain, i.e. about the used technologies and how they are combined. The correct implementation at this level is very often characterized by regularities, which are seldom expressed explicitly. Therefore it is useful to identify these regularities automatically. To name two obvious examples: "95% of the classes that implement equals() implement as well hashCode()" or "80% of the methods that open a transaction close it as well". The deviations from these rules are at least worth a review if not implementation errors. Formal Concept Analysis offers a formal framework to mine Association Rules automatically and Concept Lattices are helpful in gaining an intuitive understanding.

SHORT BIO

Daniel Speicher is a Mathematician associated with the Research Group "Artificial Intelligence Foundations" of Emeritus Prof. A. B. Cremers at the B-IT at the University of Bonn. His main research interest lies in Software Science, more specifically in the area of code quality and how the consideration of developer ideas and intentions is essential for code quality evaluation. He contributed to the project "Analyzing and Striking the Sensitivities of Embryonal Tumors (ASSET)" of the European Union and the project "Context Sensitive Intelligence (CSI)" of the Deutsche Telekom Laboratories. As member of the research groups "Research on Object-Oriented Technologies and Systems (ROOTS)" and "Software Architecture and Middleware (SAM)" he contributed to a variety of software engineering lectures and seminars. Together with his colleagues he designed and conducted a very successful series of student labs in "Agile Software Development" in the context of the "International Program of Excellence (IPEC)" at the B-IT in Bonn and Nanjing.

10:30 – 10:45 Coffee Break

10:45 – 11:30

Anabel Fraga - Carlos III of Madrid University, Spain

Systems Engineering in the industrial environment: Requirements Engineering Processes

ABSTRACT

With more than eight thousand members from academia and industry, the International Council on Systems Engineering (INCOSE) is a key reference point in systems engineering best practices. A review of the INCOSE guidelines for writing high-quality requirements and an evaluation of the impact in the quality of requirements specifications regarding the previous version of the guidelines is needed in the industrial environment. Many of these rules involve detailed aspects of the text which are cumbersome to evaluate manually. Therefore, implementing the rules within an automated tool to assess quality can greatly aid to improve the requirements engineering process within the Systems Engineering Process.

SHORT BIO

Anabel Fraga is a Computer Engineering professional. Before getting involved in academic work, she has worked in the industry as UNIX/Windows Administrator, Application Administrator for Telecom companies, Project Management and Consultancy. She obtained in 2004 her E-commerce and Networking M.Sc. in the Carlos III University of Madrid and in 2010 her PhD degree in Computer Science in the same university at the Knowledge Reuse Research Group. Her central areas of research are: Software Architecture, Information Engineering, Knowledge Management, Requirement Engineering, Systems Engineering, ITIL/ISO20000 and Reuse. She is also interested in Ethics, Innovative methods of learning for supporting new software architects and the improvement of the CS Curriculum. She is Visiting Professor of Software/Systems engineering, Information/Knowledge Engineering and Programming in Carlos III University of Madrid. She is a member of ACM CSTA, INCOSE, AEIS and IASA, and she is one of the leaders of the IASA Chapter of Madrid.

11:30 – 12:10

Iaakov Exman - The Jerusalem College of Engineering - JCE – Azrieli, Israel

Conceptual Integrity of Software Systems - An overview

ABSTRACT

Conceptual Integrity has been declared as the most important feature for software system successful design and development. The notion of software "*Conceptual Integrity*" was first introduced by Fred Brooks in his well-known book "The Mythical Man-Month" and reiterated in his later book "The Design of Design". Although the notion was proposed and its importance was stressed, it has been only vaguely delineated, lacking a clear definition. One of Brooks' ideas on how to achieve Conceptual Integrity was that of a single brilliant architect in charge of the software system design, as a way to enforce Integrity. It has been compared to the work of such an architect designing a cathedral, say that in Firenze inspired by the great mind of Brunelleschi.

Conceptual Integrity has been characterized by the principles of "orthogonality, propriety and generality". Daniel Jackson and co-workers have proposed formulations for these three principles and analyzed software systems in these terms. For instance, they studied the conceptual design of Git, the basis of the widely used GitHub service which enables concurrent distributed development of various versions of software systems.

Recently, we have proposed the Modularity Matrix as a source of the above mentioned principles of Conceptual Integrity. Orthogonality and propriety follow very nicely from the desirable characteristics of the Modularity Matrix of a given software system.

This lecture reviews the meaning of what is "*Conceptual*" within software Conceptual Integrity, formulates in detail the mentioned principles, overviews how researchers and software developers have applied Conceptual Integrity to real software design problems, and summarizes the characteristics of the Modularity Matrix relevant to software Conceptual Integrity.

SHORT BIO

Iaakov Exman is a faculty member of The Jerusalem College of Engineering (JCE – Azrieli), Department of Software Engineering. He received his Ph.D. from The Hebrew University of Jerusalem, Israel (HUJI) and performed post-doctoral research at Stanford University, California, USA. He had extensive industrial experience both with very large R&D teams within IAI (Israel Aerospace Industries) and small agile groups in start-ups, and has been a research associate at HUJI, before joining JCE in 2001. His research interests are in the area of software theory, including algebraic software theory, automated software generation from ontological conceptual structures and software aspects of quantum computation. Dr. Exman has organized together with Spanish colleagues from UC3M (University Carlos III of Madrid) the SKY International Workshop on Software Knowledge, since 2010, which has been held annually in various European locations. He has been actively associated with the GTSE Workshop on General Theory of Software Engineering from its first 2012 edition at the KTH in Stockholm, and since then annually co-located with the ICSE International Conference on Software Engineering. He is a member of the Editorial Board of the International Journal of Software Engineering and Knowledge Engineering for the "Theory of Software Engineering" area.

12:10 – 12:15 Concluding Remarks