Journal of Software Engineering and Applications, **2013, 6, 34-40**
http://dx.doi.org/10.4236/jsea.2013.69A004 Published Online September 2013 (http://www.scirp.org/journal/jsea)

New Theoretical Aspects of Software Engineering for Development Applications and E-Learning

Ekaterina Lavrischeva[1*], Alexei Ostrovski[2]

[1]Department of Software Engineering, Institute of Software Systems, Kiev, Ukraine; [2]Institute of Cybernetics, Kiev, Ukraine.
Email: *lavryscheva@gmail.com, ostrovski.alex@gmail.com

Received July 29th, 2013; revised August 28th, 2013; accepted September 4th, 2013

ABSTRACT

This paper presents new theoretical aspects of software engineering which oriented on product lines for building applied systems and software product families from readymade reusable components in conditions of program factories. These aspects are the new disciplines such as the theory of component programming; models variability and interoperability of system; theory for building systems and product families from components. Principles and methods of implementing these theories were realized in the instrumental and technological complex by lines of component development: assembling program factories using lines, e-learning to new theories and technologies in textbook of "Software Engineering" by the universities students.

Keywords: Software Engineering; Theory; Disciplines; Technologies; Interoperability; Applied Systems; Software Industry; Fabrics; E-Learning

1. Introduction

The "Software Engineering" learning course is ratified by the cabinet council of Ukraine in December 2006. The program of the course was created on the basis of Curricula-2001 and 2004, SWEBOK-2001 and developments at the Software Engineering Department, Institute of Software Systems of Ukraine. On subject of SE there were executed more than ten NAN fundamental projects and several applied projects. The author, Prof. E. Lavrischeva, gives lectures on subject of programming technology (programming languages, compilers, programming automation systems, applied systems, OOP, UML, reusability, component programming, etc.) at Taras Shevchenko Kiev National University.

The main direction of researches in SE department currently is the industry of applied systems and SPFs from reusable components. The training course on SE for students of KNU and Kiev Branch of Moscow Institute of Physics and Technology (MIPT) contains new subjects which are connected with system software in operational environments (IBM, Apple, CORBA, Java) and methods of industrial manufacturing of software products (technological and product lines, methods of conveyor assembly on these lines, interaction, quality, etc.). As a result, a new theoretical foundation of SE disciplines for producing applied systems from readymade components was developed [1-5].

In this article, a new point of view on separate theoretical and technological aspects of disciplines of software product industry is elaborated, as well as approaches to teach these aspects within the training course on SE using program factories and web sites.

2. New Disciplines in Product Industry

The main principles of SE are productivity, industry and quality, according to the corresponding body of knowledge-SWEBOK, which was developed in 1999-2004 by the international committee, formed by ACM and IEEE.

SE is connected with the following disciplines: Computer Science, electronic engineering, mathematics, telecommunication and nets, cognitive sciences and so on. SE establishes relations with these sciences and integrated the principles and ground of fundamental sciences such as [1,4,6]:

- The theory of algorithms, which encompasses normal algorithms, computing function, Turing machines, algorithmic algebras, model of algorithms and programs,

[*]Corresponding author.

etc.;

- The mathematical logic—rules of logical calculi and propositions promoting formal definitions of theory correct conclusion and logical-algebraic specification of programs;
- A set theory—quantifiers of general existence and operations over sets that are used for formal representation of axioms and theorems for various collection of software objects;
- The management theory—principles, methods and general laws of planning and control to obtaining and process information in cybernetic, organization and administrative systems;
- A Proof theory—mathematical inferences of theorems about programs with the help of axioms and statements, consistency theory, the theories of algorithmically problems and verification by Dijkstra, Hoare, and Wirth and so on;
- The theory of classification mathematical and informational object and programs, etc.

Ten areas of SWEBOK knowledge have the two directions:

Development: requirement engineering, designing architecture, developing programs, testing and maintaining software.

Management: managing project, configuration, quality, methods and means of SE.

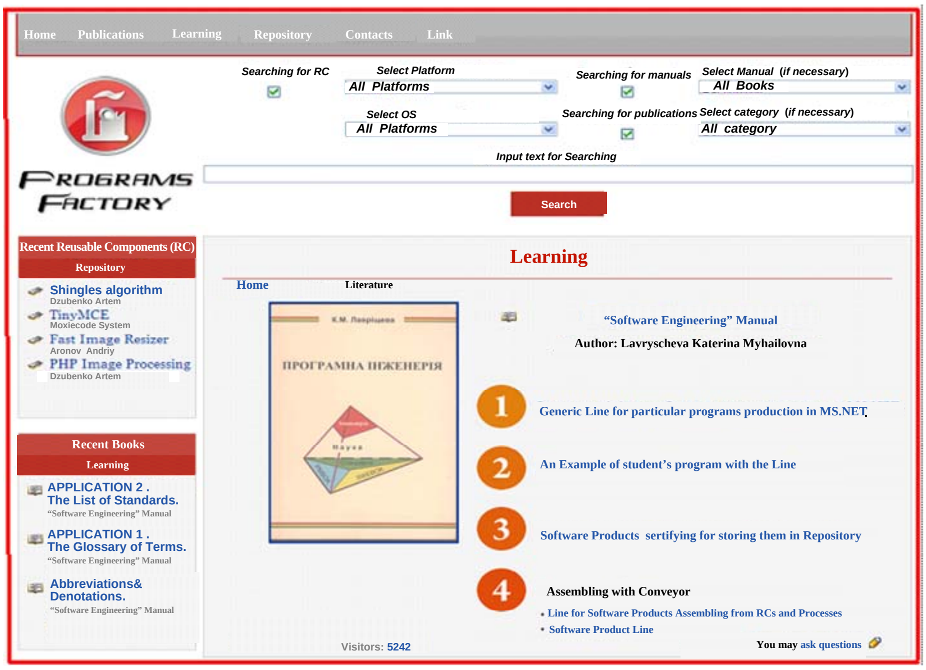
These areas knowledge correspond to the lifecycle processes of ISO/IEC 12207-1996, 2007 standard.

Definition 1. Software engineering is a system of methods, techniques and disciplines of planning, development, maintenance and exploitation of software that can be mass-produced. This definition covers all the aspects of the creation of software beginning with the formulation of requirements, development of a product and its maintenance and ending with its removal from exploitation.

Body of SWEBOK is oriented towards software objects and provides means for software development of multipurpose applied systems and informational systems. It does not, however, introduce target objects (AS, SPF, domains, etc.) and technologies for their development.

According to the approach towards teaching computer sciences in the United States, software engineering plays a prominent role in the informational disciplines (**Figure 1**). The area of software engineering comprises the entire hardware level, as well as technological and organizational levels of informational sciences. As a discipline, software engineering covers systematic software development of software system families, domains, and other scalable software projects. The main goal of software engineering is developing systematic models, reusable ready-made components, and reliable methods for producing high-quality software products. This goal encompasses theories, concepts, and paradigms, including both development structures for reliable software systems and management aspects of the designing process to meet customers' needs.

We proposed a new classification of SE disciplines (**Figure 1**), which is necessary in industrial production of applied systems and SPFs [1,2,6]:



Searching for RC ☑

Select Platform: All Platforms

Select OS: All Platforms

Searching for manuals ☑

Searching for publications ☑

Select Manual (if necessary): All Books

Select category (if necessary): All category

Input text for Searching

Recent Reusable Components (RC)

Repository

- Shingles algorithm
  Dzubenko Artem
- TinyMCE
  Moxiecode System
- Fast Image Resizer
  Aronov Andriy
- PHP Image Processing
  Dzubenko Artem

Recent Books

Learning

- APPLICATION 2 . The List of Standards.
  "Software Engineering" Manual
- APPLICATION 1 . The Glossary of Terms.
  "Software Engineering" Manual
- Abbreviations& Denotations.
  "Software Engineering" Manual

Learning

Home Literature

"Software Engineering" Manual

Author: Lavryscheva Katerina Myhailovna

1 Generic Line for particular programs production in MS.NET

2 An Example of student's program with the Line

3 Software Products sertifying for storing them in Repository

4 Assembling with Conveyor

- Line for Software Products Assembling from RCs and Processes
- Software Product Line

Visitors: 5242

You may ask questions

Figure 1. Classification of SE disciplines.

- Scientific discipline consists of the classic sciences (theory of algorithms, set theory, logic theory, proofs, and so on), lifecycle models, fundamentals data types, theory of integration and interface, theory of programming and the corresponding language tools for creating abstract models and architectures of the specified objects, etc.
- Engineering discipline is a set of technical means and methods for software development by using standard lifecycle models; software analysis methods; requirement, application and domain engineering with the help of product lines; software support, modification and adaptation to other platforms and environments.
- Management discipline contains the generic management theory, adapted to team-based software development, including job schedules and their supervising, risk management, software versioning and support.
- Economy discipline is a collection of the expert, quailtative and quantitative evaluation techniques of the interim artifacts and the final result of product lines, and the economic methods of calculating duration, size, efforts, and cost of software development.
- Product discipline consists of product lines, utilizing software resources (reusable components, services, aspects, agents, and so on), taken from libraries and software repositories; it also contains assembling, configuring and assessing quality of software.

The disciplines of scientifically and engineering recommended for students training from the second courses, and disciplines management, economies and production will be learning from magister and aspirant courses. These specialists knowledge will be used in developing product lines for building AS and SPF from RC and estimating quality of software products [1,2,5-7].

2.1. Scientific Discipline

Theory of programming is a set of methods, languages, means of description (specification) and design of target objects and methods of their proof, verification, and resting. Methods of programming in SE include [8]:

- Theoretical (algebraic, algorithmic, logical etc.) and applied (object, component, agents aspects, etc.) tools intended for designing different types of target objects;
- Methods of program verification with the help of formal (assertion, inherence, and proof) procedures of OOP, UML, VDM and so on;
- Methods of estimation of the results design from analysis environment to programming and estimation characteristic quality (reliability, accuracy, functionality, etc.) of finished products.

Besides these theories, we proposed the new scientific results of the long-term research and development:

- Concept of formal definition for reusable components, applied systems and software product families as target objects for production methodology;
- Component-based methods for the formal design of applied systems, and adapting this theory as evolutionary generative programming [6,7];
- A new model of variability and methods for management process for applied systems within SPFs [8];
- Interoperability between programs, systems, and environments [9-11].

It is necessary not only to improve theoretical level of students' qualification on software engineering, but also to teach them the new concepts, ideas and facilities, which had substantial influence on development of industrial production of applied systems.

Scientific discipline is a basic course of teaching in Universities, related to informatics and computer science. This course, per se, is theoretical, and must be supported by some courses on systematic programming (object-oriented, component-based, service-based, etc.) and additional courses from the Curricula-2004 program.

2.2. Engineering Discipline

Engineering discipline is defined as a set of engineering tools and standards oriented towards making target AS objects using the scientific discipline [2,6]. The basic means of this discipline are as the follows:

- Standards for software lifecycle (ISO/IEC 12207,) assessment (ISO/IEC 15504), etc.;
- Engineering reusable components, applications and domains;
- Generic system tools and instrumental environments for software product development.
- Engineering discipline contains the following basic tasks:
- Developing product lines from repository components and services;
- Developing AS from multi-language components using application and domain engineering [5];
- Methods and means for supplying correctness (V&V); assessing system parameters (quality, cost, capacity) and configuring AS from reusable components [2].

The core notion for engineering discipline is that of product lines. *Product line* and product family are defined in the ISO/IEC FDIS 24765:2009(E)—Systems and SE Vocabulary as "a group of products or services having the common manageable set of properties which meets the requirements of a certain market segment" [12].

Models for representing processes of software development, according to Software Engineering Institute (SEI), are the engineering and the process model.

The engineering model is created at SEI and corresponds to the three-step production: developing reusable components; merging them into software systems; managing components.

The development process encompasses defining SPF domain, designing the manufacturing process for a set of components, allowing for its context of use, restrictions and production strategy. The merge process includes designing implementations of each software system on the base of the manufactured resources and components. The management process is oriented towards processes' coordination.

The process model sorts out a set of processes that run on the two levels: domain engineering level, which is also referred to as the development "for reuse", and software systems' engineering level, referred to as the development "with reuse". Assembly lines use ready-made components, which improve the time of development and are able to support the whole manufacturing cycle according to the specific requirements and needs.

Product lines for manufacturing experimental target object elements are developed using the instrumental and technological complex (ITC) [11] developed by authors. The ITC is represented with a web site (http://sestudy.edu-ua.net), oriented on developing reliable and interoperable components, systems, and SPFs for modern environments.

2.3. Engineering Discipline

The site in question was developed as a collection of tools for SE and at the same time was displayed during lectures on SE at Kiev National University [13]. This drove authors to orient the complex towards teaching students and graduate students the basics of SE, includ ing various tools and means of its support, along the following aspects:

- Developing programs, reusable components, and services in the repository;
- Generating components and domains with subject-oriented languages like DSL;
- Assembling software products and their families from software resources and components.

Taking the above into account, we have chosen a strategy of teaching various aspects of industry-compliant SE. In order to gradually and consistently implement this strategy within the ITC, we utilized the Internet-based methods and modern programming systems that support different aspects of software development, namely:

- Protégé system to model object domains;
- Eclipse as a tool to embed different programming and system components into the ITC by using its plug-ins;
- Microsoft Visual Studio as a multifunctional tool to organize team development of the new systems, including developing software via Internet using PL, OOP, UML, and cloud computing frameworks (Azure, Amazon, etc.).
- CORBA system that has a universal broker providing interoperability between programs, written in different PL by using stub/skeleton mechanism.
- New subject-oriented DSL tools for designing domains, systems, SPFs and for implementing DSL-based descriptions (Eclipse-DSL, Microsoft DSL Tools, and others).

The start page of the web site features a list of implemented sections and subsections concerning SE. The sections in question are: Main Page, Technologies, Interoperability, Tools, Presentations, and Learning.

Each section contains subsections with keywords that specify the names of product lines (10 altogether). All sections and subsections include standardized pages, such as an overall theoretical description, an example that illustrates the concerned topic (developed with one of the workbench programming environments, in most cases), a thorough description of the example, and so on. Employees of the SE department and students of KNU and MIPT in course of writing their thematic and graduate papers implemented the web site and several product lines of software development on the basic of RC and components. Particularly, they have developed an experimental program factory [13], software means of variability and interoperability support between programs and systems, a domain description in DSL, Protégé, and so on.

2.4. Economic Discipline

The economy of software engineering is an independent SE discipline, connected with economic aspects of AS industry. It is based on economic calculations of different aspects of activity within a project, with allowance made for knowledge of all economic factors and current expenditures in the project [1,2].

The discipline contains theory and practice of solving problems on the examination of a project, cost estimation, assessment quality specified in the requirements on the system. Economic discipline provides estimation of requirements, design solutions, architectures, and development risks connected with available material and human resources, quality indices of the AS, and also financial calculations with each executor at all stages of contracts.

The discipline in question is mostly developed from the viewpoint of methods of economic calculations in SE. It includes size prediction methodologies for AS (function points analysis, feature points, mark-II function points, 3D function points, etc.), estimation of labor expenditures for system development with the help of the family of COCOMO models, and several other mathematical models for estimation of labor expenditures (Angel, Slim, Seer-SEM, etc.), as well as models connecting economic indices of AS with quality characteristics [5,9,14,15].

In forming this discipline, fundamental economic methods should be used that are related to the principles of distribution and examination of jobs in complex systems, methods of cost estimation for separate parts of a system depending on the size of its component parts and the cost of the system as a whole, existing standards providing the estimation and certification of finished products, etc.

A systematized and scientifically grounded course of the economic discipline of SE will eliminate the gap that exists in SE and is caused by the absence of the corresponding manuals and textbooks for teaching specialists who will be engaged in solving economic problems in the industrial cycle of AS production.

2.5. Management Discipline

The basis of this discipline consists of the classical management theory, production management and the standard PMBOK (Project Management Body of Knowledge) (IEEE Std.1490-2003) [1,2]. In this standard, control processes of a project and fundamental knowledge domains grouped around problems such as initiation, planning, monitoring, management, and completion are specified. The main domain of PMBOK is integration that includes the conception of management of organizational activity of the team of executors of a project. It is based on methods of making decision on resources, general design problems, correctness check services of a project, and provision of the project cost specified by the customer.

These developed basic management and planning theories, standard principles of PMBOK, the series of ISO 9001 standards regulating quality management, and the corresponding methodical project support must become the basis of the management discipline in SE. In higher schools, a curriculum on this discipline with the use of management theory will provide the preparation of highly skilled project managers and other specialists in the field of organizational management for software development.

2.6. Product Discipline

There are multiple approaches to SPF industry:

- Assembly conveyor by academician V. M. Glushkov from readymade modules (1975);
- Conveyor by K. Czarnecki and U. Eisenecker [1];
- Software factories of assembling applications by J. Greenfild, K. Short *et al.*;
- Continuous integration by Martin Fowler [3];
- EPAM assembly line (Belarus) for building various types of software, improving software quality and reducing risk;
- Compositional programming by E. Lavrischeva for developing software products from reusable components, services, artifacts, and so on [5,9,13-18].

The notion of assembly conveyor by Glushkov has been developed for many years; nowadays the experimental program factory based on product lines for reusable components has been created at Kiev National University [14-18]. Interfaces of these components contain the standardized description in certain programming languages (PL), as well as the communication interface to enable interoperability with other objects.

3. Textbook of Software Engineering

Teaching students the aspects of software industry at Ukrainian universities is currently at its initial stage. To solve some education problems, a new approach to e-learning students some aspects of SE, oriented on assembling heterogeneous components using product lines and e-learning SE disciplines, has been introduced [2,3,14-18].

The textbook is meant for teaching high school student's development and management disciplines of SE. The main idea of the textbook is development of domains, applied systems and software families with reusable components and product lines [2].

The textbook is available in three languages (Russian version is hosted on http://www.intuit.ru, Ukrainian and English versions are available on http://sestudy.edu-ua.net). The book is meant for students, magistrates and aspirants, which are specialized on department of computer science and informatics. Each area of knowledge covered by the textbook contains an overall description, questions and literature for further study. During studying these areas, students may access examples on the ITC web site—www.programsfactory.univ.kiev.ua (**Figure 2**) and other tools [13-18].

Besides that they may elaborate new concepts and artifacts and present them in a work or put on the aforementioned web site.

4. Conclusions

Software engineering disciplines regarding theoretical and industrial aspects of AS development are elaborated:

- Essentials of SE disciplines and concepts for assembling software product families from reusable components;
- Concepts of product lines for producing AS using conveyor principles;
- New models of interoperability and assembling components in systems;
- Configuring components within systems for work in ITC environment;
- Ontology representation of knowledge on new SE disciplines on the web site;
- Lines (1-4) for the components development and assembling them into complex programs systems;
- Ontologies of Life Cycle standard ISO/IEC 12207-2007 and General Data Types of standard ISO/IEC

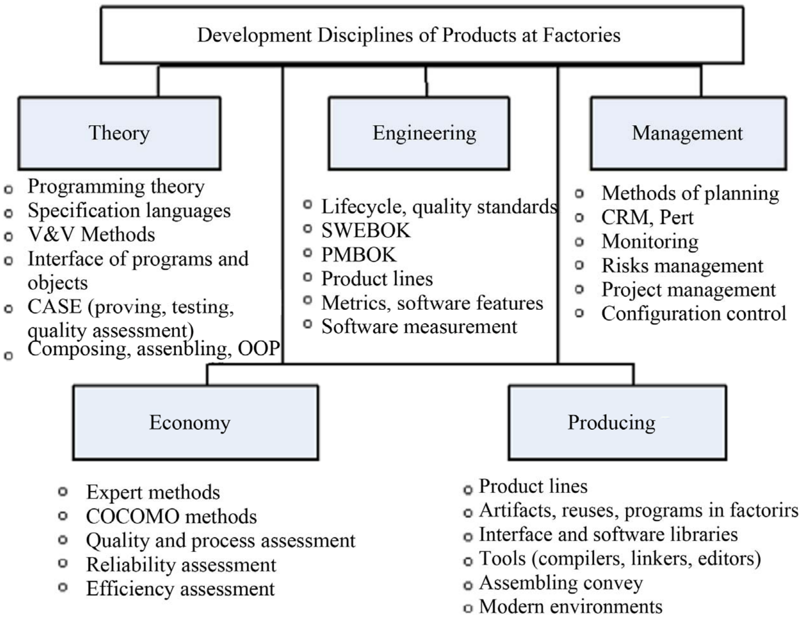


Figure 2. Classification of SE disciplines.

11404-2007 for generation the new special processes and data types of PL;

- The e-learning textbook on software engineering (**Figure 2**—ПРОГРАМНА ІНЖЕНЕРІЯ).

REFERENCES

[1] E. Lavrischeva, "Classification of Software Engineering Disciplines," *Cybernetics and Systems Analysis*, Vol. 44, No. 6, 2008, pp. 791-796. doi:10.1007/s10559-008-9053-5

[2] E. Lavrischeva, "Software Engineering (in Ukrainian)," Akademperiodika, Kiev, 2008, 319 p.

[3] E. Lavrischeva and V. Petruchin, "Methods and Means of Software Engineering," 2007. http://www.twirpx.com/ and http://www.intuit.ru/

[4] E. Lavrischeva, "Concept of Scientific Software Industry and Approach to Calculation of Scientific Problems (in Ukrainian)," *Problems in Programming*, Vol. 1, 2011, pp. 3-17.

[5] E. Lavrischeva, G. Koval, L. Babenko, O. Slabospitska and P. Ignatenko, "New Theoretical Foundations of Production Methods of Software Systems in Generative Programming Context," UK-2011, Isoftware Institute NANY, 2011. 277 p. http://www.nbuv.gov.ua/

[6] E. Lavrischeva, V. Grischenko, "Assembly Programming. Basics of Software Industry (in Russian)," 2nd Edition, Naukova Dumka, Kiev, 2009, 371 p. http://www.twirpx.com/

[7] E. Lavrischeva, "Formation and Development of the Modular-Component Software Engineering in Ukraine (in Russian)," Akademperiodika, Kiev, 2008.

[8] E. Lavrischeva, "Problem of Interoperability between Heterogeneous Objects, Components, and Systems. Approach to Solve It," *Proceedings of 7th International Programming Conference on* "*UkrProg* '2010", 2010, pp. 28-41. http://www.isofts.kiev.ua/

[9] E. Lavrischeva, "Instrumental and Technological Complex for Developing and Learning Aspects of Software System Development (in Ukrainian)," *Bulletin of NAS of Ukraine*, Vol. 3, 2012, pp. 17-27.

[10] A. Ostrovski, "Approach to Interconnection Support between Java and MS.NET Programming Environments (in Russian)," *Problems in Programming*, Vol. 2, 2011, pp. 37-44.

[11] I. Radetskyi, "One of Approaches to Maintenance interconnection Environments Visual Studio and Eclipse (in Ukrainian)," *Problems in Programming*, Vol. 2, 2011, pp. 45-52.

[12] K. Pohl, G. Böckle and F. J. Linden, "Software Product Line Engineering: Foundations, Principles and Techniques," Springer-Verlag, New York, 2005.

[13] K. M.Lavrischeva, "Theory and Practice of Software Factories," Springer, New York, 2011, pp. 961-972

[14] A. Aronov and A. Dzubenko, "Approach to Development of the Students' Program Factory (in Ukrainian)," *Problems in Programming*, Vol. 3, 2011, pp. 42-49. http://www.isofts.kiev.ua/

[15] P. Andon, G. Koval, T. Korotun, E. Lavrischeva and V. Suslov, "Foundation of Software Systems Quality Engineering (in Ukrainian)," 2nd Edition, Akademperiodika, Kiev, 2007. http://www.nbuv.gov.ua

[16] E. Lavrischeva, A. Ostrovski and I. Radetskyi, "Approach to E-Learning Fundamental Aspects of Software Engineering," *Proceedings of ICTERI*-2012: *ICT in Education, Research and Industrial Applications*, 2012, pp. 176-187. http://senldogo0039.springer–sbm.com/ocs/home/ICTERI 2012.-CEUR-WS-paper-17-p-176-187.pdf/

[17] A. Kolesnyk and O. Clabospitskaya, "Tested Approach for

Variability Management Enhancing in Software Product Line," *Conference ICTERI*-12. http://ceur-ws.org/Vol-848/ICTERI-2012-CEUR-WS-paper-31-p-155-162.pdf

[18] E. Lavrischeva, A. Dzubenko and A. Aronov, "Conception of Programs Factory for Representation and E-Learning Disciplines of Software Engineering," 9*th International Conference ICTERI*, *ICT in Education*, *Research and Industrial Applications*, I*ntegration, Harmonization and Knowledge Transfer*, Ukraine, June 17-21, 2013. http://ceur-ws.org/Vol-1000/

Appendix

About Authors

Katerina Lavrischeva, (E.M.) Dr. Sci. (Phys.-Math.), prof. (1989), first in USSR led the "Software Engineering" department (1980). Her scientific works in SE are known in Ukraine, CIS and journals of foreign countries. She elaborates the new concepts and methods technologies development of systems and programs from ready components by compositions. She has 8 monografies and 3 textbooks on tecnologies of programming and SE for software industry. She studies students of Kiev National University to this task from 1976 and to day of e-text-book.

Alexei Ostrovski, Graduate Student of Institute of Cybernetics National Academia Sciences Ukraine from 2012.