Features of Development and Use in the Educational Process of NRNU MEPhI Tutoring Integrated Expert Systems in the Field of Study «Software Engineering»[1]

Galina V. Rybina [1][0000-0002-4077-3660], Andrey Y. Nikiforov [1][0000-0001-9990-2845], Elena S. Fontalina[1][0000-0003-3154-365X] and Ilya A. Sorokin[1][0000-0001-5959-5369]

[1] National Research Nuclear University MEPhI, Moscow, Russia
galina@ailab.mephi.ru

Abstract. The purpose of this work is to show some methodological aspects of software engineering, reflecting the departmental view that has developed over the years on issues of training specialists in the field of PT, and also highlight and briefly characterize modern innovations that are currently used for practical implementation of the educational process on the basis of the ontological approach and the use of tutoring integrated expert systems (IES) to support basic courses / disciplines. The formation of the modern scientific and educational appearance of the department in the field of PT is naturally associated with the historical, conceptual and technological integration of three specializations ("Computer Software", "Intelligent Systems and Technologies" and "Automated Information Systems") within the "Software Engineering" training area, which is reflected in the creation of a single ontological space of knowledge and skills on the basis of which, using the technology of teaching IES, competence-oriented models of graduates were constructed. The ontological approach integrates heterogeneous knowledge, data, documents (FSES standards) and others allows you to effectively organize the educational process in the field of "Software Engineering" using the most modern technologies in the development and use of intelligent systems of various classes.

Keywords: Tutoring Integrated Expert Systems, Competence-Oriented Model of a Specialist, Student Model, Software Engineering.

1 Introduction

The origins of the term "Software Engineering" (SE) in our country has long been preceded by the term "Programming technology" (PT), which denotes methods, means and tools that ensure the process of creating software systems [1].

[1] The work was performed with the Russian Foundation for Basic Research support (Project No. 18-01-00457).

Modern understanding of SE was summarized in 2004 in SWEBOK (Software Engineering Body of Knowledge) - as a system of methods, tools and disciplines for planning, developing, operating and maintaining software systems, including readiness for implementation. In 2014, SWEBOK v3.0 was published, which received international recognition as ISO / IEC Technical Report 197596: 2015.

These definitions are so close that, in the opinion of many experts [2, 3] and others, SE can be considered a natural development of the PT, which is very important both for the preservation and further development of the national experience of the PT as a whole, and in terms of training specialists in "Software Engineering".

The Department of Cybernetics of the National Research Nuclear University MEPhI has accumulated many years of experience in teaching PT courses / disciplines as part of the specialization of “Computer Software” (CS) of the basic specialty “Applied Mathematics”. The creation and active development of CS belonging to the periods 1971-1990 and 1991-2014 The current stage, which began in 2015, continues as Software Engineering degree programs (bachelor and master).

The purpose of this work is to show some methodological aspects of software engineering, reflecting the departmental view that has developed over the years on issues of training specialists in the field of PT, and also highlight and briefly characterize modern innovations that are currently used for practical implementation of the educational process on the basis of the ontological approach and the use of tutoring integrated expert systems (IES) to support basic courses / disciplines [3, 4].

2 History (or How the Foundation for the Training of Specialists in Computer Software Development was Laid)

Students studying at the Department of Cybernetics the basics of PT today have little idea that all the programs and curricula of specialization of the CS were compiled and implemented in close scientific and technical cooperation with such basic enterprises of the domestic industry as [5]:

- NICEVT (Research Center of Electronic Computing), which was the lead developer of software for third-generation computers (ES EVM), on the basis of which the full cycle of training in the field of system programming for one of the student groups of the department was carried out;
- INEUM (Institute of Electronic Managed Machines), together with which were developed compiler FORTRAN-VI for ASVT (Aggregate Computing Equipment), on the basis of which the training courses «Languages programming», «Basics of building translators» were created;
- Keldysh Institute of Applied Mathematics, together with which were created the SIMULA-67 compiler for BESM-6 and ES computers using the REFAL language (a license was obtained for the ES computers), developed an ADA language compiler, and laid the foundation for a number of courses on programming languages and system modeling;
- Minaviaprom of the USSR, which in 1976 organized at the department an ONIL-722 industrial laboratory for the development of on-board software development

tools on the basis of which the fundamentals of training specialists were laid and such courses as “Programming Technology”, “Software Verification and Certification”, “Modern Software Development Tools” were introduced into the educational process.

Thus, research and development in conjunction with the above-listed enterprises played an important role for the educational process of the department, ensuring the delivery of fundamental programming courses, the theory of building translators, and tools for supporting the development, operation and maintenance of software systems.

Another important historical prerequisite for the formation, evolution of development and modern view of the PT is the close interaction of the CS specialization with other specializations - “Systems of artificial intelligence” (the modern name is “Intelligent Systems and Technologies”) and “Information and Mathematical Software of the Automated Control System” (modern name "Automated Information Systems").

For example, the main accents of the scientific school of Professor L.T. Kuzin [5], the founder of the Cybernetics department and an outstanding scientist in the field of artificial intelligence (AI), has always been put on the research, development and development of models and programming languages for AI based on the dialects of the KRL, FRL languages, advanced OPSS and other popular concepts and knowledge modeling formalisms.

As the language of the implementations of the corresponding translators, the domestic language REFAL was used, which was the “calling card” of all the solutions of the department, which had a strong programmer school in the field of creating translators and specialized tools. Accordingly, courses on the basics of designing and programming intelligent systems of various architectural typologies, learning problem-oriented and domain-specific languages for AI (PROLOGUE, LISP), and tools were developed and tested for the educational process (INTER-EXPERT, ECO, Level 5 Object, G2, etc.).

Therefore, the formation of the modern scientific and educational appearance of the department in the field of PT is naturally associated with the historical, conceptual and technological integration of all three specializations within the “Software Engineering” training area, which is reflected in the creation of a single ontological space of knowledge and skills for training specialists in "Software Engineering".

3 The Place and Role of Languages and Programming Tools: Methodological Aspects

Experience has shown that the main criterion for future success of students in the field of PT is the inculcation of skills in computational, algorithmic and logical thinking within the framework of basic courses in programming, modern tools and design technologies for information and intelligent systems of various architectural typologies.

Among the languages traditionally studied today in introductory programming courses dominate Python, Java, as well as C and C++. There are many studies [6, 7] and others, where comparisons are made of the features of programming languages by such criteria as simplicity, integrity, capabilities of basic data types, syntax, degree of data abstraction, expressiveness, etc. In particular, in accordance with [6], according to

the aggregate of these criteria, Java turned out to be the best language, followed by Python, and then the rest. Starting from the first course, when learning C and C ++ languages, students learn the basics of algorithmization, building data structures, understanding abstract and user-defined data types and classes, and acquire skills in documenting programs. However, for trainees it is important to get not only knowledge of the specific language features, but also skills of using this language for solving problems, therefore the choice of language directly depends on the direct subject of study, degree of audience (bachelor, masters) and design features of a particular class of software systems.

In particular, the use of OOP paradigms adds another level of abstractions that require an understanding of the basics of class design and the skills to integrate them into existing software products. Based on this, the most important component of the curriculum of "Software Engineering" is the consideration of various programming paradigms so that in the future, graduates can choose the most appropriate type of programming language to solve specific practical problems.

For example, the study of functional programming languages (Common Lisp, Scheme, Haskell, and others) and Prolog-type logic programming languages allows students to solve problems in the field of AI. Familiarity with the basics of scripting languages such as Python, used when working with web applications, as well as with the principles of event-oriented programming, allows you to understand how to create applications that interact with the user.

It is important to note that the development of universal applications for solving complex interdisciplinary tasks (in particular, the creation of intelligent systems for particularly important sectors of the economy) makes it possible to discover in practice the advantages of various languages and programming tools.

Mastering universal programming languages (C, C ++, etc.) allows you to gain experience that can be used to solve problems in the field of system programming and create various problem-oriented and subject-oriented specialized tools to support the development of a wide class of intelligent and informational systems (like CASE-tools, intelligent packages, Workbench-systems, etc.).

In general, based on the experience of building competence-oriented models of graduates (bachelors and masters) in the field of PT, it is necessary to provide solutions to the following important tasks in the organization of the modern educational process:

- conceptual understanding of the programming process and instilling computational and logical thinking skills for the implementation of specific tasks in the form of a program and / or a software system;
- selection and use of various programming paradigms, including OO-oriented, functional, logical, environment-oriented, and various types of programming languages, as well as the development of competence in the use of a particular language;
- application of new approaches from the field of AI by presenting typical knowledge from the field of programming in the form of plans for solutions [8, 9], to help students who have difficulty in moving from a method or algorithm for solving a problem to its software implementation.

A significant role in solving the above problems is currently assigned to the organization of the educational process on a single conceptual and methodological basis,

which is carried out by attracting methods of ontological engineering and the technology of teaching IES [3, 4] and others.

4 Formation of Professional and Universal Competencies in Software Engineering

Let us consider some features of professional and universal competences represented in professional standards [10], which are formed taking into account the scientific and educational appearance of the department in the field of PT. The main focus is traditionally placed on the ontologies of such disciplines as "Cybernetic Systems Programming Technology (Software Project Management)", "Software Systems Design and Architecture", "Designing of systems based on knowledge", "Dynamic Intelligent Systems", etc., within which students receive basic theoretical knowledge and practical techniques, typical for the development of traditional software systems and intellectual systems of various architectural typologies, including: life cycle and methodology design and development of software systems; various software system architectures; modeling in languages such as UML, testing, verification and certification of software; CASE-tools, workbench-systems and other types of tools that allow to automate the process of developing software systems, etc.

It is important that within the framework of these courses and research projects students implement individual educational projects, during the course of which the following professional competencies are acquired:

- requirements elicitation (for software, information software and user interface);
- selecting of a life cycle model;
- using metrics to measure project properties;
- selecting of the style (approach) to the software implementation and the rules (standard) of the programming style;
- estimation of the project functions coverage by its program implementation;
- designing of user interface and interface specifications;
- justification of the interface compliance specification to the needs of the potential user;
- documenting model, information, operational, functional, architectural and other aspects of the project;
- documenting solutions to test cases and selected real-world problems;
- documentation of trial operation and commercial implementation.

5 An Example of the Implementation of the Ontological Approach to the Organization of the Educational Process on Intellectual Systems

Starting from 2008, the Department of Cybernetics has already gained considerable experience in developing and using teaching IES and web-based IES for individual

courses / disciplines that operate on the basis of the applied ontology “Intelligent Systems and Technologies” (a detailed description of the methods and tools for developing of tutoring IES can be found in [3, 4]).

The ontology model in accordance with c [4] is presented in the form of a semantic network in which vertices reflect various elements of courses / disciplines (sections, topics, subtopics, definitions, etc.), including a model of targeted competencies and a set of models of various learning influences on the identification of knowledge and the skills of the trainees, and the edges show the different types of connections (relationships) between course elements, competencies and learning influences.

Automated support for the construction of applied ontologies for each course / discipline based on this model is carried out using special tools that function as part of the AT-TECHNOLOGY workbench [3].

From a methodological point of view, it is important to note that by creating the ontology “Intelligent Systems and Technologies” (which currently has about 900 vertices from 6 courses / disciplines) it was possible to build a single ontological knowledge and skills space, which allows:

- implement the training cycle in accordance with the curriculum and teaching materials (lectures, practical exercises, etc.);
- provide a fully functional construction of competence-oriented models of students for the entire period of study (bachelor, master), and as a final result the formation of models of future professionals, including individual psychological portraits);
- compare the current competencies of the trainees with the target ones, identify the so-called “problem areas” [3, 4] in knowledge and plan the training effects in the form of solving specific practical tasks for each student in order to achieve a higher level of competencies, etc .;
- show great intellectual learning opportunities [4], namely: individual planning of the methodology for studying specific training courses; intellectual analysis of learning tasks; intellectual decision support, etc.

An example of the implementation of the processes of automated construction of competence-oriented models of specialists in the field of knowledge engineering based on the ontology "Intelligent Systems and Technologies" is given in [11].

6 Conclusion

Thus, the ontological approach integrates heterogeneous knowledge, data, documents (FSES standards) and others allows you to effectively organize the educational process in the field of "Software Engineering" using the most modern technologies in the development and use of intelligent systems of various classes.

References

1. Vel'bitskiy I.V. Tekhnologiya programmirovaniya. – Kiev: Tekhnika (1984)
2. Lavrishcheva E.M. Programmnaya inzheneriya. Paradigmy, tekhnologii i CASE-sredstva / Uchebnik dlya vuzov. – M.: Izdatel'stvo Yurayt (2016)

3. Rybina G.V. Intellektual'nye sistemy: ot A do Ya. Seriya monografiy v trekh knigakh. Kniga 1. Sistemy, osnovannye na znaniyakh. Integrirovannye ekspertnye sistemy. – M.: Nauch-tekhlitizdat (2014)
4. Rybina G.V. Intellektual'naya tekhnologiya postroeniya obuchayushchikh integrirovannykh ekspertnykh sistem: novye vozmozhnosti // Otkrytoe obrazovanie. 21(4) pp.43-57 (2017)
5. Kibernetika: lyudi, gody, sobytiya. Kollektivnaya monografiya / Pod red. G.V. Rybinoy. – M.: Radiotekhnika (2013)
6. M.S. Farooq, S.A. Khan, F. Ahmad, S. Islam, A. Abid, An Evaluation Framework and Comparative Analysis of the Widely Used First Programming Languages. PLoS ONE 9(2): e88941 (2014)
7. R. Sebesta, Concepts of Programming Languages, 11th Edition, Pearson (2015)
8. S. Lucci and D. Kopec, Artificial Intelligence in the 21st Century, 2nd ed. Dulles, Virginia: Mercury Learning & Information (2015)
9. Schweikert C. Programming Languages with Plan Knowledge Representation for Learning // In Proceeding of the Int'l Conf on Artificial Intelligence (ICAI 16): CSREA Press. P. 389 (2016)
10. Portal Federal'nykh gosudarstvennykh obrazovatel'nykh standartov vysshego obrazovaniya www.fgosvo.ru
11. Rybina G.V., Sergienko E.S. Nekotorye podkhody k avtomatizirovannomu postroeniyu kompetentnostno-orientirovannykh modeley spetsialistov v oblasti inzhenerii znaniy s ispol'zovaniem obuchayushchikh integrirovannykh ekspertnykh sistem // Informatsionno-izmeritel'nye i upravlyayushchie sistemy. 15(7) pp.3-15 (2017)