National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

Department of Design Automation for Energy Processes and Systems

Automated Design Technologies and Programs Verification
Work program of the discipline (Syllabus)

Course Requisites

Educational level	***Third (PhD)***
Field of knowledge	*12 Information technology*
Specialty	*121 Software Engineering*
Educational program	*Software Engineering*
Course status	*Normative*
Form of study	*Part-time*
Year of study, semester	*2 year, autumn semester*
Number of ECTS credits	*4 ECTS credits (120 hours)*
End-of-semester control / control measures	*Exam, modular test, calendar control*
Timetable	*According to the timetable for the autumn semester of the current academic year http://rozklad.kpi.ua/*
Language of study	*English*
Information about course leader / teachers	Lecturer: Ph.D., Associate Professor, Kuzminykh Valeriy Oleksandrovych, vakuz0202@gmail.com Practical training: Ph.D., Associate Professor, Kuzminykh Valeriy Oleksandrovych, vakuz0202@gmail.com
Course placement	http://campus.kpi.ua/

Curriculum

1. Description of the discipline, its purpose, subject of study and learning outcomes

The discipline "Technologies of automated design and verification of programs" (TAPVP) refers to the normative disciplines for gaining in-depth knowledge of the specialty of the curriculum of the third level of higher education in the specialty 121 Software Engineering. The course helps students understand and master many aspects of information analysis in various control systems, computer and telecommunications systems and networks.

The purpose of the discipline is to form higher competencies in higher education students:

- Ability to develop high-quality and reliable software for complex software packages and systems based on the latest technologies and software development standards.

- Ability to apply formal methods of design, development and research of software systems and technologies in research.

- Ability to conduct experimental studies to evaluate the effectiveness and security of software.

- Ability to develop technical documentation for research projects.

- Ability to perform original research, achieve scientific results that create new knowledge in software engineering and related
interdisciplinary areas, and can be published in leading scientific journals
in information technology and related fields.

- Ability to expand the boundaries of knowledge using the results of original research.

- Ability to ensure continuous self-development and self-improvement, responsibility for the development of others in the professional field, adhering to pedagogical ethics, the rules of academic integrity in scientific and pedagogical activities.
- Ability to use adequate methods of effective interaction with representatives of different groups (social, cultural and professional).
- Ability to work in a team, form positive relationships with colleagues, communicate with the general scientific community and the public in the field of software engineering.
Subject of the discipline: modern methods and technologies of automated design, testing and quality management of software (software). Acquiring, within this area, theoretical knowledge and practical skills harmoniously combine modern achievements and views on the outlined issues.
As a result of studying the discipline, applicants for higher education will acquire the following program learning outcomes:
- Know modern technologies of automated design and verification of programs.
- Be able to apply, develop and improve methods of automated software design.
- Be able to apply, develop and improve software verification methods.
- Know the principles of building scenario models and verification of information analysis scenarios.
- Have advanced conceptual and methodological knowledge in software engineering and at the boundaries of subject areas, as well as research skills sufficient to conduct scientific and applied research at the level of modern world achievements in the field, gaining new knowledge and / or innovation.
- Develop and research conceptual, mathematical and computer models of processes and systems, use them effectively to gain new knowledge and / or create innovative products in software engineering and related interdisciplinary areas.
- Understand the theoretical foundations underlying information research methods
systems and software, research methodologies and computational experiments.
- Be able to formulate and solve problems of optimization, adaptation, forecasting, management and decision-making on processes, tools and resources for software development, implementation, maintenance and operation.

2. Prerequisites and postrequisites of the discipline (place in the structural and logical scheme of education according to the relevant educational program)

Successful study of the program "Technology of automated design and verification of programs" is preceded by the study of "Fundamentals of Programming", "Fundamentals of Operating Systems", "Object-Oriented Programming", "Modern Methodologies and Software Development Technologies" bachelor's and master's degrees. Also needed is the ability to use a computer at the administrator level, the ability to program, basic knowledge of set theory, the ability to build Use-Case models to further analyze and predict future application functionality to be designed and tested.

The theoretical knowledge and practical skills obtained as a result of mastering the discipline "Technologies of automated design and verification of programs" can be useful for research on the topic of the dissertation, as well as for mastering the discipline "Software Reengineering Methods".

3. The content of the discipline

Discipline "Technology of automated design and verification of programs" provides

study the following topics:

Topic 1. Modern technologies of automated design and verification of programs

Topic 2. Testing software code

4. Training materials and resources

Basic literature:

1. Голышев Л.К. Прикладной системный анализ и проектирование компьютерных информационных систем. Учебное пособие..- ГП «Информационно-аналитическое агентство.- Киев.-2008.-с.315
2. Основи системного аналізу: підручник / М.З. Згуровський, Н.Д. Панкратова. - К. : Видавнича група BHV, 2007. - 544 с.
3. Сучасні технології автоматизованого проектування та верифікації програм. Тестування програмного забезпечення. Методичні вказівки до виконання лабораторних робіт. [Електронне видання] / Уклад.: Я.Ю. Дорогий, О.О. Дорога-Іванюк. – К.: НТУУ «КПІ», 2020. – 87 с.
4. Кузьміних В.О., Тараненко Р.А. Основи управління ІТ проектами // Навч. посіб. для студ. 122 «Комп'ютерні науки» / КПІ ім. Ігоря Сікорського. Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 9 від 29.05.2019 р.) – Київ : КПІ ім. Ігоря Сікорського, 2019. – 76 с.

Additional literature:

6.Винниченко, И. Автоматизация процессов тестирования / И. Винниченко. - СПб.: питерб 2005. - 203с. - ISBN 5-469-00798-7.
7. Кузьміних В.О., Отрох С.І., Воронько М.П., Тараненко Р.А. Fundamentals of IT project management // Навч. посіб. для студ. 122 «Комп'ютерні науки» / КПІ ім. Ігоря Сікорського. Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 7 від 27.02.2020 р.) – Київ : КПІ ім. Ігоря Сікорського, 2019. – 74 с.
8. Фатрелл Р., Шафер Д., Шафер Л. Управление программными проектами. Пер. с англ. – М.: Вильямс, 2017.
9. Marik B. "When Should a Test Be Automated?", Testing Foundations, 1998, pp.1-20.
10. Success with Test Automation [Електронний ресурс] – Режим доступа: https://www.prismnet.com/~wazmo/succpap.htm. (Дата звернення: 20.01.2020).

Educational content

5. Methods of mastering the discipline (educational component)

№	Type of educational activity	Description
Topic 1. Modern technologies of automated design and verification of programs		
1	Lecture 1. Modern technologies of automated design and verification of programs	General characteristics of the discipline. Basic terms and concepts.
2	Lecture 2. Characteristics of areas of knowledge in software engineering - SWEBOOK.	SWEBOK (Software Engineering Body of Knowledge) - international standard ISO / IEC TR 19759 from 2015. Engineering requirements. Software design.
3	Lecture 3. Characteristics of knowledge areas in software engineering - SWEBOOK.	Software testing. Methods and tools. Software quality.
4	Lecture 4. Life cycle standard and models.	Life cycle characteristics of the ISO / IEC 12207 standard. Formation of applied life cycle models. Types of software lifecycle models.

5	Lecture 5. Methodologies for planning the development of software systems.	Waterfall technique and Agile technique. Advantages and disadvantages. SCRUM and KANBAN. Features of use and opportunities.
Topic 2. Testing software code		
6	Lecture 6. Methods of verification and testing of programs Verification and validation of programs.	Testing, verification and validation are differences in concepts. Methods of testing and verification processes and their place in different life cycle models. Specification languages. Methods of proof, verification and testing of programs. Technological processes of verification in the project.

№	Title of Laboratory work	Hours
1	Created a model of information flows of systems in the form of workflow diagrams.	2
2	Description of automated information systems. Description of the interaction of elements and functions of systems.	2
3	Description of the system design process	2

6. Self-study

№	Topic for self-study
1	Study of architecture, visual interfaces and tools of CASE system Visual Paradigm.
2	Requirements analysis and development of UML diagrams of software system conceptual design
3	Development of UML-diagrams of logical level of software system design: modeling of static aspects.
4	Development of UML-diagrams of logical level of software system design: modeling of dynamic aspects.
5	Development of UML-diagrams of the physical level of software system design.
6	Methods of proof, verification and testing of programs. Specification languages. Software systems testing. . Testing the user interface.

Policy and control

7. Policy of academic discipline (educational component)

Attending classes

It is obligatory to attend laboratory classes, because they carry out control measures on the assessments on which the rating is formed.

Control measures missed

There are no hours in the workload of teachers to accept student arrears, but with the good will of the teacher, if students received low scores in the defense of laboratory work or tests, they have an attempt to increase them at the end of the semester. The time and place of additional classes are determined by the teacher.

Academic integrity

The policy and principles of academic integrity are defined in Section 3 of the Code of Honor of the National Technical University of Ukraine "Kyiv Polytechnic Institute named after Igor Sikorsky". Details: https://kpi.ua/code.

Norms of ethical behavior

Norms of ethical behavior of students and employees are defined in Section 2 of the Code of Honor of the National Technical University of Ukraine "Kyiv Polytechnic Institute named after Igor Sikorsky". Details: https://kpi.ua/code.

Procedure for appealing the results of control measures

Students have the opportunity to raise any issue related to the procedure of control measures and expect that it will be considered in accordance with predefined procedures

8. Types of control and rating system of assessment of learning outcomes (RSO)

The grade from the discipline is set according to a multi-point system, followed by a traditional one.

The maximum number of points in the discipline is 100.

The tables below provide information on the percentage contribution of controls to the semester rating.

Type of semester control	*Percentage contribution of types of control to the semester rating*
Laboratory work 1	*20*
Laboratory work 2	*20*
Laboratory work 3	*20*
Exam	*40*
Total	*100%*

Rating scale (R):

The sum of weight points of control measures (individual tasks, computer workshop, modular control work and credit) during the semester is:

R = 20 + 20 + 20 + 40 = 100 points

Thus, the rating scale of the credit module is 100 points.

Prerequisite for admission to the exam is enrollment in all laboratory work, as well as a starting rating (rc) of at least 35 points. In order for a student to receive appropriate grades (ECTS and traditional), his RD rating is translated according to the table:

Number of pints	*Grades*
100-95	Excellent
94-85	Very good
84-75	Good
74-65	Satisfactorily
64-60	Sufficiently
Less 60	Unsatisfactory
Admission conditions are not met	Not allowed

Table of correspondence of rating points to grades on the university scale:

Number of pints	*Grades*
100-95	Excellent
94-85	Very good
84-75	Good
74-65	Satisfactorily
64-60	Sufficiently
Less 60	Unsatisfactory
Admission conditions are not met	Not allowed

9. Additional information on the discipline (educational component)

List of theoretical questions for the exam

1. Name the main existing methods of software design.

2. Name the main notations of structural analysis and design (SADT).

3. Model-oriented approach.

4. Name the areas of knowledge of SWEBOK software development engineering.

5. Give the basic concepts of SWEBOK.

6. What models are created as part of the use of MDA.

7. SCRUM methodology.

8. Basic concepts of SCRUM methodology.

9. Kanban methodology.

10. Basic concepts of Kanban methodology.

11. Software life cycle.

12. The main stages of LC software.

13. What is the principle of organization of the cascade model.

14. The main phases, advantages, disadvantages and cases of using the cascade model.

16. How are the requirements for the software product?

17. Name the main methods of verification of requirements, give a description of each of them.

18. Diagram of UML usage options.

19. The main elements of the diagram of options for the use of UML, their graphical notation.

20. Types of relationships and graphical representation of the UML usage diagram.

21. UML class diagram. The concept of "class", its characteristics.

22. Diagram of UML classes. Basic relations between classes in UML, their graphic notation.

23. Diagram of UML classes. The difference between aggregation and composition.

24. UML object diagram. The concept of "object".

25. Parameterized class (template).

28. UML state diagram. Automatic in UML, prerequisites for its existence.

29. UML state diagram. The concept of "state", a graphical representation of the state in UML.

30. UML state diagram. The main types of states, examples of their graphic notation.

31. UML state diagram. Security conditions. Parallel transition, types of parallel transition.

32. UML activity diagrams. Status of actions, transitions and tracks on activity diagrams, graphic marking.

33. UML activity diagrams. Examples (with graphical notation) of the main types of states in the diagram
activities.

34. UML sequence diagram. Object life line, graphic notation.

35. UML sequence diagram. Management focus, graphic notation.

36. Messages on the UML sequence diagram. Examples of basic graphics messages designation.

37. Deployment diagram in UML. The main elements.

38. Synchronization diagram in UML. The main elements of the timing chart.

39. Software quality requirements.

40. Standards in quality engineering. CMM / CMMI. ISO standards. Six. SIGMA. ITIL.

41. Software certification in Ukraine.

42. Quality metrics.

43. Preliminary quality assessment based on statistical methods.
44. Preliminary assessment of the complexity of the program at the stage of development of specifications of program requirements.
45. Preliminary assessment of the complexity of the program at the stage of determining the architecture.
46. Classification of quality models.
47. ISO 9126 quality model.
48. Verification process.
49. Validation process.
50. Collective analysis processes. Inspections. Code review. Audits.
51. The concept of defect, failure and failure.
52. Quantitative assessments in testing. Testing efficiency.
53. Types of testing. Testing levels.
54. Testing by the methods of "white box", "gray box" and "black box".
55. Testing techniques. Testing process.
56. Testing documentation. Analysis of test results.
57. Management of the testing process.
58. Creating a testing team.
59. Documentation of tests.
60. Defect reports. Testing log.

Work program of the discipline (Syllabus):

Developed by Ph.D., Associate Professor, Kuzminykh Valeriy Oleksandrovych

Approved *by department _____ (protocol № ___ from _____)*

Resolved *by Methodical commission of the faculty (protocol № __ from _____)*