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

Department of Computer Engineering

Software Engineering Components. Part 2. Software modeling. Analysis of software requirements.

The program of the academic discipline (Syllabus)

Details of the academic discipline

Cycle of Higher Education	***First cycle of higher education (Bachelor's degree)***
Field of Study	*12 Information technologies*
Specialty	*121 Software Engineering*
Education Program	*Computer Systems Software Engineering*
Type of Course	*Normative*
Mode of Studies	*Full-time education*
Year of studies, semester	*2 year (3 semester)*
ECTS workload	*4 credits*
Testing and assessment	*Test*
Course Schedule	*Lectures 18 (36 hours), Laboratory9 (18hours)*
Language of Instruction	*English*
Course Instructors	Lecturer: Prof., Dr.Sc. Bogdan Korniyenko, b.korniyenko@kpi.ua Laboratory: Prof., Dr.Sc. Bogdan Korniyenko, b.korniyenko@kpi.ua
Access to the course	https://classroom.google.com

Program of academic discipline

1 Course description, goals and objectives, and learning outcomes

Discipline "Software Engineering Components. Part 2. Software modeling. Analysis of requirements for software" should form in future specialists the theoretical foundations of program architecture design, modeling of systems for which software is created, a modern level of information and programming culture; mastery of the basics of a systematic approach to software modeling, the principles of creating and using the UML modeling language, structural and object-oriented analysis and design of software complexes; creation of technical documentation in accordance with current standards; acquiring practical software design skills.

The purpose of studying the discipline "Software Engineering Components. Part 2. Software modeling. Analysis of requirements for software" is aimed at forming future engineers with a modern level of information and digital culture, mastering the basic principles of creating software products; acquisition of practical skills of independent compilation of professional software and use of modern information technologies to solve various problems of an applied nature. The formation of learning goals and students' understanding of various aspects of the future profession is a necessary component of training a qualified software engineer (Software Engineer), system architect (System Architect), software architect (Software Architect).

The subject of study of the discipline is modern methods, tools and technologies of software development.

According to the requirements of the EP, the discipline "Software Engineering Components" should ensure that applicants acquire competencies and program learning outcomes: PC01-05, PC07, PC08, PC11-13, PLO01-04, PLO06-11, PLO13-20. In particular, after mastering the module "Software Engineering Components. Part 2. Software modeling. Analysis of requirements for software" must demonstrate the following competencies and program learning outcomes:

- the ability to formulate and ensure software quality requirements in accordance with customer requirements, specifications and standards;
- the ability to adhere to specifications, standards, rules and recommendations in the professional field when implementing life cycle processes;
- the ability to apply fundamental and interdisciplinary knowledge to successfully solve software engineering tasks;
- the ability to evaluate and take into account economic, social, technological and environmental factors affecting the field of professional activity;
- the ability to accumulate, process and systematize professional knowledge regarding the creation and maintenance of software and recognition of the importance of lifelong learning;
- the ability to implement phases and iterations of the life cycle of software systems and information technologies based on appropriate models and software development approaches.

According to the results of studying the educational discipline "Software Engineering Components. Part 2. Software modeling. Analysis of requirements for software", the following **knowledge** should be obtained:

- the ability to formulate and ensure software quality requirements in accordance with customer requirements, specifications and standards;
- the ability to carry out the system integration process, apply change management standards and procedures to maintain the integrity, overall functionality and reliability of the software;
- know and apply methods of developing algorithms, designing software and data and knowledge structures;
- have skills in team development, approval, design andrelease of all types of software documentation.

Skills that should be acquired as part of studying the academic discipline "Software Engineering Components. Part 2. Software modeling. Analysis of requirements for software":

- know the main processes, phases and iterations of the software life cyclesoftware;
- to know and be able to use methods and means of collection, formulation andsoftware requirements analysis;
- choose initial data for design, guided by formalrequirements description and modeling methods;
- to be able to calculate the economic efficiency of softwaresystems.

Such a combination of general and special competences, theoretical and practical knowledge, skills and abilities helps to increase the professional level of bachelor's degree holders in order to carry out effective activities in the field of development ofsoftware engineering.

2 Pre-requisites and post-requisites of the discipline (place in the structural and logical scheme of training according to the relevant educational program)

Necessary disciplines: "Algorithms and data structures", "Fundamentals of programming", "Software Engineering Components. Part 1. Introduction to software engineering".

Module "Software Engineering Components. Part 2. Software modeling. Analysis of requirements for software" is necessary for studying the following parts of the discipline "Software Engineering Components", and can also be useful when studying the disciplines "Methodologies and technologies of software development", "Risk management and project quality".

3 Structure of the credit module

A list of the main topics included in the study program of the discipline "Software Engineering Components. Part 2. Software modeling. Analysis of requirements for software":

Section1. Theoretical foundations of modeling.

Topic1.1. Purpose, subject area and tasks of the discipline.
Topic1.2. Concept of subject area and hierarchy of modeling levels.
Topic1.3. Concept of visual languages and visual models.
Topic1.4. Concept of model graph and diagram.
Topic1.5. Entity-relationship diagrams.
Topic1.6. Diagrams of functional modeling.
Topic1.7. Data flow diagrams.

Section2. Modern technologies of object-oriented analysis and design of information systems.

Topic2.1. Concepts of object-oriented analysis and design.
Topic2.2. Methodology of object-oriented programming.
Topic2.3. Basic principles and conceptual foundations of SADT and CASE technologies.
Topic2.4. Structural model of the subject area.
Topic2.5. Functional-oriented and object-oriented methodologies of domain description (IDEF0).
Topic2.6. Functional methods of data flows.
Topic2.7. Object-oriented technique.

Section3. Structure of the UML language.

Topic3.1. Purpose of the UML language.
Topic3.2. The general structure of the UML language.
Topic3.3. Basic semantic constructions of the language.
Topic3.4. Features of the graphic representation of UML language diagrams.
Topic3.5. Graphical notation of the use case diagram.
Topic3.6. Basics of structural modeling: classes.
Topic3.7. Architectural modeling of component diagrams.

Section4. Identification of project requirements.

Topic4.1. Identification of requirements and needs.
Topic4.2. Work with users.
Topic4.3. Work by interested persons.
Topic4.4. Techniques for identifying requirements and needs.
Topic4.5. Classification of requirements.
Topic4.6. Documenting requirements using use cases.
Topic4.7. Prioritization of requirements.

Section5. Specification of software requirements. Specification languages. Criteria for good requirements.

Topic5.1. System functions.

Topic5.2. Data requirements.

Topic5.3. Requirements for external interfaces.

Topic5.4. Criteria of quality requirements. Quality attributes.

Topic5.5. Specification languages.

Topic5.6. Change management process in traditional and Agile development.

Topic5.7. The document is a concept. Requirements specification.

4 Educational resources and materials

Basic:

1. Pressman, Roger (2010) *Software Engineering: A Practitioner's Approach*, McGraw Hill, New York, NY.
2. Sommerville, Ian (2011) *Software Engineering*, Addison-Wesley, Boston, MA.
3. Stephens, Rod (2015) *Beginning Software Engineering*, Wrox.
4. Tsui, Frank , Orlando Karam and Barbara Bernal (2013) *Essentials of Software Engineering*, Jones & Bartlett Learning , Sudbury, MA.
5. Pfleeger, Shari (2001) *Sofwtare Engineering: Theory and Practice*, Prentice Hall, Upper Saddle River, NJ.

Supplementary:

1 Larman, C. (2005) *Applying UML and Patterns: AnIntroduction to Object-Oriented Analysis and Design and iterative Development*, Pearson

2 Ambler, S. (2002) *Agile Modeling: Effective Practices for Extreme Programming and the Unified Process*, NewYork, John Wiley & Sons.

3 Bass, D.L., Clements, D.P. andKazman, D.R. (2012) *Software Architecture in Practice*, 3rd edn, Upper Saddle River, NJ, Addison Wesley

4 Beck, K. (2004) *Extreme Programming Explained: Embrace Change*, Upper Saddle River, NJ, Addison Wesley

5 Clemens Szyperski (2002) Component Software: Beyond object-oriented programming, Addison-Wesley

6 John Cheesman & John Daniels (2000) UML Components: A simple process for specifying component-based software (The component software series) Addison-Wesley

7 Rob Pooley, Perdita Stevens (2006) *Using UML Software Engineering with Objects and Components*, second edition. Addison-Wesley

8 Christopher Fox (2006) *Introduction to Software Engineering Design*. Addison Wesley

Educational content

5 Methodology

Sections and topics	Hours			
	Total	Including		
		Lectures	Practical work	Self-study
Section1. Theoretical foundations of modeling. *Topic1.1. Purpose, subject area and tasks of the discipline.* *Topic 1.2. Concept of subject area and hierarchy of modeling levels.* *Topic 1.3. Concept of visual languages and visual models.* *Topic 1.4. Concept of model graph and diagram.* *Topic 1.5. Entity-relationship diagrams.* *Topic 1.6. Diagrams of functional modeling.* *Topic 1.7. Data flow diagrams.*	18	4	4	10
Section2. Modern technologies of object-oriented analysis and design of information systems. *Topic 2.1. Concepts of object-oriented analysis and design.* *Topic 2.2. Methodology of object-oriented programming.* *Topic 2.3. Basic principles and conceptual foundations of SADT and CASE technologies.* *Topic 2.4. Structural model of the subject area.* *Topic 2.5. Functional-oriented and object-oriented methodologies of domain description (IDEF0).* *Topic 2.6. Functional methods of data flows.* *Topic 2.7. Object-oriented technique.*	26	6	4	16
Section3. Structure of the UML language. *Topic 3.1. Purpose of the UML language.* *Topic 3.2. The general structure of the UML language.* *Topic 3.3. Basic semantic constructions of the language.* *Topic 3.4. Features of the graphic representation of UML language diagrams.* *Topic 3.5. Graphical notation of the use case diagram.* *Topic 3.6. Basics of structural modeling: classes.* *Topic 3.7. Architectural modeling of component diagrams.*	28	8	4	16
Section4. Identification of project requirements. *Topic 4.1. Identification of requirements and needs.* *Topic 4.2. Work with users.* *Topic 4.3. Work by interested persons.* *Topic 4.4. Techniques for identifying requirements and needs.* *Topic 4.5. Classification of requirements.* *Topic 4.6. Documenting requirements using use cases.* *Topic 4.7. Prioritization of requirements.*	24	6	4	14
Section5. Specification of software requirements. Specification languages. Criteria for good requirements. *Topic 5.1. System functions.* *Topic 5.2. Data requirements.*	26	12	4	10

Topic 5.3. Requirements for external interfaces. *Topic 5.4. Criteria of quality requirements. Quality attributes.* *Topic 5.5. Specification languages.* *Topic 5.6. Change management process in traditional and Agile development.* *Topic 5.7. The document is a concept. Requirements specification.*				
Total hours in semester	120	36	20	64

Laboratory works:

The purpose of conducting laboratory classes is for students to consolidate theoretical knowledge and acquire the necessary practical skills for working with modern technologies for software engineering.

- Laboratory work #1: Stage of identifying the problem;
- Laboratory work #2: The stage of problem analysis;
- Laboratory work #3: Determining the boundaries of the system;
- Laboratory work #4: Usage scenarios;
- Laboratory work # 5: Project limitations.

6 Self-study

- preparation for lectures by studying the previous lecture material;
- preparation for laboratory work with the study of the theory of laboratory work with an oral answer to the given questions of the section;
- preparation of results of laboratory work in the form of a protocol.

Attendance Policy and Assessment

7 Attendance Policy

During classes in an academic discipline, students must adhere to certain disciplinary rules:
- extraneous conversations or other noise that interferes with classes are not allowed;
- the use of mobile phones and other technical means is not allowed without the teacher's permission.

Laboratory works are submitted personally with a preliminary check of theoretical knowledge, which is necessary for the performance of laboratory work. Validation of practical results includes code review and execution of test tasks.

8 Monitoring and grading policy

Current control: survey on the subject of the lesson
Calendar control: conducted twice a semester as a monitoring of the current status of meeting the syllabus requirements.
Semester control:test
Conditions for admission to semester control: enrollment of all laboratory work

System of rating points and evaluation criteria

The student's rating in the discipline consists of the points he receives for:
1. performance and defense of 5 laboratory works;
2. execution of 2 modular control works (MCW).

Laboratory works:
"excellent", a complete answer to the questions during the defense (at least 90% of the required information) and a properly prepared protocol for laboratory work - 10 points;
"good", a sufficiently complete answer to the questions during the defense (at least 75% of the required information) and a properly prepared protocol for laboratory work - 8 points;
"satisfactory", incomplete answer to the questions during the defense (at least 60% of the required information), minor errors and a properly prepared protocol for laboratory work - 6 points;
"unsatisfactory", an unsatisfactory answer and/or an improperly prepared protocol for laboratory work - 0 points.

Modular Control Works:
"Excellent", full answer (not less than 90% of the information you need) - 25 points;
"Good", a full answer (not less than 75% of the information you need), or a complete answer with minor mistakes - 20 points;
"Satisfactory", incomplete answer (but not less than 60% of the information you need) and minor mistakes - 16 points;
"Unsatisfactory", unsatisfactory response (incorrect problem solution), requires mandatory re - writing at the end of the semester - 0 points.

The maximum sum of weighted points of control measures during the semester is:

$$R=5*Rlab+2*Rmcw=5*10+2*25=100.$$

Table1 — Correspondence of rating points to grades on the university scale

Score	*Grade*
100-95	Exellent
94-85	Very good
84-75	Good
74-65	Satisfactory
64-60	Sufficient
below 60	Fail
Course requirements are not met	Not graded

Syllabus of the course:

designed by Professor of the Department of Information Systems and Technologies, Bogdan Korniyenko

adopted by Department of Computer Engineering (protocol № 10, 25.05.2022)

approved by the methodical commission of FICT (protocol № 10, 09.06.2022)