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

Department of Computer Engineering

Software Engineering Components. Part 3. Software architecture.

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 (4 semester)*
ECTS workload	*5 credits*
Testing and assessment	*Exam*
Course Schedule	*Lectures 18 (36 hours), Laboratory18 (36 hours)*
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

Studying the discipline "Software Engineering Components. Part 3. Software architecture" occupies an important place in the structure of knowledge acquisition in the educational program "Computer systems software engineering". The software design process involves developing an architecture to reduce system complexity through abstraction and separation of powers. Software architecture combines different points of view on the system, which is a strong argument for the need and feasibility of developing a software architecture. This discipline forms a systematic approach to software design and the engineering worldview of an IT specialist.

The purpose of studying the discipline "Software Engineering Components. Part 3. Software modeling. Analysis of requirements for software" is theoretical and practical training of students, which should ensure their acquisition of basic knowledge in the field of modern design technologies and the study of software architecture, acquisition of practical skills in the implementation of software systems, the basics of modeling and analysis of software systems, analysis of development, specification and requirements management.

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 3. Software architecture" 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 3. Software architecture", the following **knowledge** should be obtained:

- the ability to participate in software design, including modeling (formal description) of its structure, behavior and functioning processes;
- the ability to adhere to specifications, standards, rules and recommendations in the professional field when implementing life cycle processes;
- the ability to evaluate and take into account economic, social, technological and environmental factors affecting the field of professional activity.

Skills that should be acquired as part of studying the academic discipline "Software Engineering Components. Part 3. Software architecture":

- design, develop and test software of various types;
- develop technical documentation for software;
- know and apply professional standards and other legal documents in the field of software engineering;
- be able to develop a human-machine interface;
- to know, analyze, choose, competently apply the means of ensuring information security (including cyber security) and data integrity in accordance with the applied tasks being solved and the software systems being created.

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", "Software Engineering Components. Part 2. Software modeling. Analysis of requirements for software", “Object-oriented programming”.

Module "Software Engineering Components. Part 3. Software architecture " 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 3. Software architecture":

Section1. Variants of software system architectures.

Topic1.1. The concept of software architecture.

Topic1.2. Architectural samples, reference models and variants of architectures.

Topic1.3. Architectural structures and representations.

Topic1.4. Relations between structures.

Topic1.5. Architecture based on ports.

Topic1.6. Architecture of independent components.

Topic1.7. Service-oriented architectures.

Section2. Software architecture design. Design methodology. Modularity.

Topic2.1. Modules, modular interface approach.

Topic2.2. Rationale for modularity.

Topic2.3. The internal characteristic of the module is connectivity.

Topic2.4. The connection of modules is an external characteristic of the module.

Topic2.5. Multi-layered architecture

Topic2.6. Logical structure of multilayer architecture.

Topic2.7. Design patterns of multi-layered architecture.

Section3. Architecture of Web applications.

Topic3.1. Components of a Web application.

Topic3.2. The architecture of a multi-level Web application.

Topic3.3. Architecture of a mixed Web application.

Topic3.4. Architecture of a multi-level Web application based on CORBA technology.

Topic3.5. Web application architecture with OLE DB, ADO and ODBC interfaces.

Section4. Design patterns.

Topic4.1. The concept of pattern design.

Topic4.2. Types of patterns.

Topic4.3. Generating patterns.

Topic4.4. Structural patterns.

Topic4.5. Behavioral patterns.

Topic4.6. The concept of a socket.

Topic4.7. Functions of sockets.

Section5. Techniques of object-oriented design.

Topic5.1. The main principles of object-oriented architecture (OOA).

Topic5.2. Advantages of OOA.

Topic5.3. Concepts and elements of OOA.

Topic5.4. The concept of COM technology.

Topic5.5. Problems that are solved with the help of COM technology.

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 andPatterns: AnIntroductiontoObject-OrientedAnalysisandDesignanditerativeDevelopment*, Pearson

2 Ambler, S. (2002) *AgileModeling: EffectivePracticesforExtremeProgrammingandtheUnifiedProcess*, NewYork, JohnWiley&Sons.

3 Bass, D.L., Clements, D.P. andKazman, D.R. (2012) *SoftwareArchitectureinPractice*, 3rd edn, UpperSaddleRiver, NJ, AddisonWesley

4 Beck, K. (2004) *ExtremeProgrammingExplained: EmbraceChange*, UpperSaddleRiver, NJ, AddisonWesley

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 RobPooley, PerditaStevens (2006) *Using UML SoftwareEngineeringwithObjectsandComponents*, secondedition. Addison-Wesley

8 ChristopherFox (2006) *IntroductiontoSoftwareEngineeringDesign*. AddisonWesley

Educational content

5 Methodology

Sections and topics	Hours			
	Total	including		
		Lectures	Practical work	Self-study
Section1. Variants of software system architectures. *Topic 1.1. The concept of software architecture.* *Topic 1.2. Architectural samples, reference models and variants of architectures.* *Topic 1.3. Architectural structures and representations.* *Topic 1.4. Relations between structures.* *Topic 1.5. Architecture based on ports.* *Topic 1.6. Architecture of independent components.* *Topic 1.7. Service-oriented architectures.*	28	4	8	16

Section2. Software architecture design. Design methodology. Modularity. *Topic 2.1. Modules, modular interface approach.* *Topic 2.2. Rationale for modularity.* *Topic 2.3. The internal characteristic of the module is connectivity.* *Topic 2.4. The connection of modules is an external characteristic of the module.* *Topic 2.5. Multi-layered architecture* *Topic 2.6. Logical structure of multilayer architecture.* *Topic 2.7. Design patterns of multi-layered architecture.*	32	6	10	16
Section3. Architecture of Web applications. *Topic3.1. Components of a Web application.* *Topic 3.2. The architecture of a multi-level Web application.* *Topic 3.3. Architecture of a mixed Web application.* *Topic 3.4. Architecture of a multi-level Web application based on CORBA technology.* *Topic 3.5. Web application architecture with OLE DB, ADO and ODBC interfaces.*	30	8	6	16
Section4. Design patterns. *Topic 4.1. The concept of pattern design.* *Topic 4.2. Types of patterns.* *Topic 4.3. Generating patterns.* *Topic 4.4. Structural patterns.* *Topic 4.5. Behavioral patterns.* *Topic 4.6. The concept of a socket.* *Topic 4.7. Functions of sockets.*	26	6	6	14
Section5. Techniques of object-oriented design. *Topic 5.1. The main principles of object-oriented architecture (OOA).* *Topic 5.2. Advantages of OOA.* *Topic 5.3. Concepts and elements of OOA.* *Topic 5.4. The concept of COM technology.* *Topic 5.5. Problems that are solved with the help of COM technology.*	34	12	6	16
Total hours in semester	150	36	36	78

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: Development of .asmx web service. Service testing;
- Laboratory work #2: Development of a client (web server application) for .asmx web service. Program testing;
- Laboratory work #3: Implementation of a WCF service using Visual Studio;
- Laboratory work #4: Implementation of a WCF service as a REST service using standard bindings. Development of web and mobile service clients;

• Laboratory work # 5: Web API web service development;

• Laboratory work #6:Development and use of own service for implementation of dependencies in ASP.NETCore application.

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:exam
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 6 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 - 5 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 - 4 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 - 3 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) - 10 points;
"Good", a full answer (not less than 75% of the information you need), or a complete answer with minor mistakes - 8 points;

"Satisfactory", incomplete answer (but not less than 60% of the information you need) and minor mistakes - 6 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:

$$R=6*Rlab+2*Rmcw=6*5+2*10=50.$$

Exam:
Admission to the exam is subject to passing all laboratory work, writing both modular test papers, and a starting rating of at least 17 points.
At the exam, students perform a written test. Each ticket contains two theoretical questions (tasks). Each question (task) is valued at 25 points.

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)