МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ СЕМЕНА КУЗНЕЦЯ

ЗАТВЕРДЖЕНО
на засіданні кафедри
інформаційних систем.
Протокол № 1 від 22.08.2023 р.

ПОГОДЖЕНО
Проректор з навчально-методичної роботи

[illegible] ________________ Каріна НЕМАШКАЛО

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ
* 02071211 *

ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

робоча програма навчальної дисципліни (РПНД)

Галузь знань	**12 "Інформаційні технології"**
Спеціальність	**121 "Інженерія програмного забезпечення"**
Освітній рівень	**перший (бакалаврський)**
Освітня програма	**"Інженерія програмного забезпечення"**

Статус дисципліни	**обов’язкова**
Мова викладання, навчання та оцінювання	**англійська**

Розробник:
к.е.н., професор підписано КЕП Ірина ЗОЛОТАРЬОВА

Завідувач кафедри
інформаційних систем [illegible] ________________ Дмитро БОНДАРЕНКО

Гарант програми [illegible] ________________ Олег ФРОЛОВ

Харків
2024

MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
SIMON KUZNETS KHARKIV NATIONAL UNIVERSITY OF ECONOMICS

APPROVED at the meeting of Information Systems Department Протокол № 1 від 22.08.2023 р.	**AGREED** Vice-rector for educational and methodical work [illegible] Karina NEMASHKALO

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ * ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ * 02071211

SOFTWARE ENGINEERING

Program of the course

Study programme	**12 "Information technologies"**
Specialty	**121 "Software engineering"**
Study cycle	**перший (бакалаврський)**
Study programme	**"Інженерія програмного забезпечення"**

Course status	**mandatory**
Language	**English**

Developer: professor	digital signature	Iryna Zolotaryova
Head of Information Systems Department	[illegible]	Dmytro Bondarenko
Programme Guarantor	[illegible]	Oleh Frolov

Kharkiv
2024

INTRODUCTION

The course is devoted to studying methods and tools for managing software projects (work planning and resource management), expert evaluation of intermediate development results during the life cycle (LC) processes, risk assessment of building a software system, and the quality achieved for it. This discipline uses standards governing the processes of the LC, requirements engineering, testing, and quality assurance by checking the indicators on the LC processes and the final product to evaluate them. The course covers theoretical and practical issues of building various software systems to perform software development tasks. The course will be useful for future managers and executors of information systems development projects, primarily in organizing software product design processes using modern tools.

The course "Software Engineering" is studied by applicants for the specialty "Software Engineering" of all forms of study in the third year during the sixth semester.

The purpose of teaching the course "Software Engineering" is to study methodologies for the development of information systems software; application of the UML (Universal Modeling Language) language for modeling and designing information systems; application of software tools to expand and deepen theoretical knowledge and applied skills on basic concepts and definitions in the field of software quality assurance, acquisition of skills in the use of modern information technologies for

Formation of systematic understanding:

Modern concepts, principles, methodologies, and technologies of software development.

Life cycle (LC) of information systems software systems.

Best practices and standards of software engineering (SE) used in the IT industry.

Mastering skills:

Design, development, testing, and implementation of software.

Use of modern software engineering tools.

Application of OOP principles in software design and development.

Formulation and documentation of software requirements.

Ensuring software quality according to specified standards.

Study:

Stages, models, and artifacts of the software lifecycle.

Classes, objects, methods, inheritance, and polymorphism in OOP.

Software engineering standards IEEE, ISO/IEC, CMMI.

Methods and techniques of software testing.

Software quality assurance: standards, methods, metrics.

CASE tools, IDEs, and version control systems.

The objectives of the course are: to form a systematic understanding of the concepts, principles, methodology, and technologies of software development as a set of processes for developing software systems based on the life cycle (LC) of information systems software; study the purpose of the tools of the object approach to software design; software engineering standards; methods of creating requirements for software development; methods and techniques for testing software.

The subject of the course is technologies for the development, testing and operation of software modules of information systems based on modern methodologies and tools of the object-oriented approach to software design, study of methodologies for developing information systems software, the use of the Universal Modeling Language (UML) for software modeling and design, the use of tools - IBM Rational Rose, Scrum, Adile, Canban study of the main provisions for software design based on modern technical, software, instrumental and communication tools.

The object of the course is the processes that reflect various aspects of the creation and maintenance of software products.

The learning outcomes and competencies formed by the course are defined in Table 1.

Learning outcomes and competencies formed by the course

Learning outcomes	Competencies
LO 01	GC3, GC5, GC6, SC8, SC10
LO 02	GC2, GC8, GC9, GC10, GC11, GC12, SC5, SC9
LO 03	GC7, SC11, SC12
LO 04	SC4, SC5, SC12
LO 06	GC1, GC2, SC10, SC11
LO 07	GC1, GC2, SC6, SC7, SC8, SC11
LO 12	SC1, SC2, SC3
LO 13	SC8
LO 14	GC2, GC3, SC2, SC3, SC13
LO 16	SC15

LO 01 Analyze, purposefully search for, and choose information and reference resources and knowledge necessary for solving professional tasks, taking into account modern achievements of science and technology.
LO 02 To know the code of professional ethics, understand the social significance and cultural aspects of software engineering and adhere to them in professional activities.
LO 03 To know the software life cycle's basic processes, phases, and iterations.
LO 04 To know and apply professional standards and other regulatory documents in the field of software engineering.
LO 06 Ability to select and use a software development methodology appropriate to the task.
LO 07 To know and apply in practice the fundamental concepts, paradigms, and basic principles of functioning of language, tools, and computing tools of software engineering.
LO 12 Apply in practice effective approaches to software design;
LO 13 To know and apply methods of algorithm development, software design, and data and knowledge structures.
LO 14 Apply in-practice software tools for domain analysis, design, testing, visualization, measurement, and documentation of software.
LO 16 Have skills in team development, coordination, design, and production of all types of software documentation.

GC 1 Ability to think abstractly, analyze, and synthesize.
GC 2 Ability to apply knowledge in practical situations.
GC 3 Ability to communicate in the state language both orally and in writing.
GC 5 Ability to learn and master modern knowledge.
GC 6 Ability to search, process, and analyze information from various sources.
GC 7 Ability to work in a team.
GC 8 Ability to act based on ethical considerations.
GC 9 The desire to preserve the environment.
GC 10 Ability to act in a socially responsible and conscious manner.
GC 11 The ability to exercise their rights and responsibilities as a member of society, to understand the values of civil (free democratic) society and the need for sustainable development, the rule of law, human and civil rights, and freedoms in Ukraine.
GC 12 Ability to preserve and enhance the moral, cultural, scientific values and achievements of society based on an understanding of the history and patterns of development of the subject area, its place in the general system of knowledge about nature and society, and the development of society, technology, and technology, to use various types and forms of physical activity for active recreation and healthy lifestyle.
SC 01 Ability to identify, classify, and formulate software requirements.
SC 02 Ability to participate in the design of software, including modeling (formal description) of its structure, behavior, and processes of functioning.
SC 03 Ability to develop architectures, modules, and components of software systems.
SC 04 Ability to formulate and ensure software quality requirements by customer requirements, terms of reference, and standards.
SC 05 Ability to comply with specifications, standards, rules, and recommendations in the professional field when implementing life cycle processes.
SC 06 Ability to analyze, select, and apply methods and tools to ensure information security (including cybersecurity).
SC 07 Knowledge of data information models, and ability to create software for data storage, extraction, and processing.
SC 08 Ability to apply fundamental and interdisciplinary knowledge to successfully solve software engineering problems.
SC 09 Ability to evaluate and take into account economic, social, technological, and environmental factors affecting the field of professional activity.
SC 10 Ability to accumulate, process, and systematize professional knowledge of software development and maintenance and recognize the importance of lifelong learning.
Ability to implement phases and iterations of the life cycle of software systems and information technologies based on appropriate software development models and approaches.
SC 11 Ability to carry out the process of system integration, and apply change management standards and procedures to maintain the integrity, overall functionality, and reliability of the software.
SC 12 Ability to reasonably select and master tools for software development and maintenance.
SC 13. Ability to reasonably choose and master tools for software development and maintenance.
SC 15. Ability to use technologies and tools for distributed data processing and parallel computing in software development.

COURSE CONTENT

Topic 1. Software life cycle models. Documentation of the automated software system (APS). Vision & Scope, SRS, User Stories.

An overview of the main models of ZhC PZ (waterfall, iterative, spiral, incremental). Advantages and disadvantages of each model. Selection of the optimal model of residential and commercial software for a specific project. Professional standards and other regulatory documents in the field of software engineering.

APS documentation. Types of APS documentation (technical task, specification of requirements, operational documentation). Rules for processing APS documentation. Use of CASE tools for APS documentation.

Software development project planning. Determination of goals, tasks, and stages of the project. Assessment of resources required for project implementation, taking into account economic, social, technological, and environmental factors.

Project risk management. Identification and analysis of project risks. Development of a risk response plan. Monitoring and control of project risks.

Topic 2. Software Requirements as a field of knowledge in software engineering. Requirements management and communication with SWEBOK tasks.

Concept and classification of software requirements: Functional and non-functional requirements. Qualitative and quantitative characteristics of requirements.

Collection and analysis of requirements. Methods of gathering requirements (interviews, surveys, observations, document analysis). Analysis of completeness, correctness, consistency, and priority of requirements. Use of CASE tools for requirements management.

Documentation of requirements. Types of documents describing the requirements (requirement specification, technical task). Rules for processing documents with requirements. Using templates and standards to document requirements.

Requirements management. Planning and control of the process of collection, analysis, documentation, and verification of requirements. Change and verification of requirements. Management of conflicts and risks related to requirements. Analysis and selection of methods and means of ensuring information security.

Connection with other knowledge areas of SWEBOK. Impact of requirements on other stages of the software life cycle. Ensuring software compliance with requirements.

Topic 3. Features of definition and analysis of business requirements. Definition of requirements as a stage of software development. Problems of managing the process of developing software requirements.

Definition of business requirements. The process of defining, analyzing, and formulating requirements for a software product based on business needs and its goals. Application of fundamental and interdisciplinary knowledge to solve problems.

Analysis of business requirements. The process of analyzing business requirements, including determining the structure of requirements, their priorities and relationships, as well as the development of functional requirements.

Stage of software development. The importance of defining requirements at the early stages of software development, methods, and approaches to this process.

Problems of managing the requirements development process. Problems that may arise during the requirements development process and how to solve them.

Key aspects of requirements management: requirements change management, requirements conflict management, risk management, and others.

Topic 4. Object-oriented approach to software design. UML language.

Basic concepts of object-oriented approach: basic concepts of object-oriented approach, classes, objects, inheritance, polymorphism, and others.

Modeling process using UML. The main elements of UML (Unified Modeling Language): class diagrams, interaction diagrams, sequence diagrams, activity diagrams, and others, as well as their application in the software modeling process.

Software development using UML. Software development techniques and approaches using UML, such as Agile, Rational Unified Process (RUP), Scrum, etc.

Management of the software development process using UML. Problems and methods of managing the software development process using UML

Topic 5. Requirements analysis process. UML Use Case Diagram.

Requirements analysis process. The process of gathering, analyzing, and documenting software requirements. Definition of functional and non-functional requirements, coordination of requirements with stakeholders, as well as formulation of generalized requirements and requirements for the project.

Requirements analysis methods. Prototyping, research method, UML (Use Case Diagrams), etc.

Diagram of UML use cases (Use Case Diagrams). The concept of UML use case diagrams, their structure, and elements such as actors, use cases, inclusions, extensions, etc.

Examples of using UML use case diagrams to model and analyze software requirements.

Topic 6. Methods of object analysis and modeling. Advanced requirements analysis. UML activity diagram.

Methods of object analysis and modeling. Object analysis methods, scenario analysis methods, and others.

Advanced requirements analysis. Use case diagram analysis, convention analysis, constraint analysis, etc.

UML activity diagram (Activity Diagrams), its structure, elements, purpose, and methods of activity analysis and modeling.

Examples of using the UML activity diagram to model various processes and activities in software.

Topic 7. Management of changes in software requirements. The main tasks of requirements management. Requirements tracing.

Software requirements change management: identifying, documenting, and tracking requirements changes and their impact on the development process.

The main tasks of requirements management: definition, analysis, documentation, storage, and tracking of requirements during the life cycle of software development. Non-functional requirements, taking into account issues of environmental protection, sustainable development, social responsibility, values of a democratic society, the rule of law, human and citizen rights, by the laws of Ukraine, understanding of the historical context and laws of the development of society, technology, and technology.

Requirements Tracing: The process of tracing requirements from source documents and analyzing their impact on various stages of software development.

Requirements Change Management Processes: requirements change management processes and techniques, such as Agile, Scrum, and Waterfall, as well as tools and practices for their implementation.

Topic 8. Software design stage. UML state diagram.

Description of the software design phase, including requirements analysis, architectural design, detailed design, modeling, and testing.

Statechart Diagrams UML (Statechart Diagrams): its structure, elements, purpose, and methods of analysis and modeling of system states.

Software design processes and methods using the UML state diagram. Examples of using a UML state diagram to model various system states and transitions in software.

Topic 9. System architecture design. UML class and component diagrams.

Description of the system architecture design process. Methods and processes of system architecture design, such as defining the main components, their relationships, and others.

UML class diagram (Class Diagram): its structure, elements, purposes and methods of class analysis and modeling.

Processes and methods of system architecture design using the UML class diagram. Examples of using the UML class diagram to model different classes and their relationships in software.

UML component diagram (Component Diagram): its structure, elements, and purpose.

Defining the system integration process, and applying change management standards and procedures to maintain the integrity, overall functionality, and reliability of the software.

Topic 10. Software developers. Work in a team.

The main functions of a software developer: are analysis of requirements, design, implementation, testing, and support of software.

Basic skills and knowledge that software developers must have, such as programming, algorithmization, software architecture, testing, version control, communication, and others.

Organization and teamwork. Methods and approaches to organizing the work of developers in a team, such as Agile, Scrum, Waterfall, etc., as well as problems and approaches to solving them.

Communication and team communication. Methods and approaches to communication and team communication, such as meetings, joint planning, information sharing, etc. Approaches to the accumulation, processing, and systematization of professional knowledge regarding the creation and maintenance of software, lifelong learning.

Tasks and responsibilities of software developers in a team.

The list of laboratory studies by the course is given in the table. 2.

Table 2

List of laboratory studies

Topic name	Content
Topic 1-3.	Business analysis (analysis of the subject area). 1) Description of the current situation (AS IS) in UML format (use case diagram). 2) Description of business problems of the customer/clients. 3) Description of interested parties/end users. 4) Subject area dictionary (glossary). 5) Analysis of business requirements in the form of a goal map and the form of an impact map: Impact Mapping.
Topic 4.	Analysis of existing solutions (analogs, competitors). Comparative analysis by criteria.
Topic 5-7.	Creating a solution concept. Description of solution functions. Detailing of functional and non-functional requirements. Justification of the toolkit for software development and support.
Topic 8-10.	Designing a solution. System architecture design. User interface prototyping. Drawings of interface screens/pages.

The list of self-studies by the course is given in the table. 3.

Table 3

List of self-studies

Topic name	Content
Topic 1 - 10	Studying lecture material
Topic 1 - 10	Preparation for laboratory classes
Topic 1 - 10	Preparation for the exam

The number of hours of lectures, laboratory studies, and hours of self-study is given in the technological card of the course.

TEACHING METHODS

In the process of teaching the course, in order to acquire certain learning outcomes, to activate the educational process, it is envisaged to use such learning methods as:

Verbal (lecture (Topic 1, 3, 5, 7, 9, 10), problem lecture (Topic 8), lecture-visualization (Topic 2, 4, 6)).

In-person (demonstration (Topic 1-10)).

Laboratory work (Topic 1-4), project method (Topic 1-4).

ASSESSMENT FORMS AND METHODS

The university uses a 100-point accumulative system for evaluating the learning outcomes of students of higher education.

The current assessment is carried out during lectures and laboratory classes and is aimed at checking the level of preparedness of the higher education applicant to perform specific work and is evaluated by the sum of points scored:

– for courses with a form of semester assessment examination (exam): the maximum amount is 60 points; the minimum amount that allows a student of higher education to pass an exam is 35 points.

The final assessment includes semester assessment and certification of the student of higher education.

Semester control is carried out in the form of a semester exam/

The maximum number of points that a student of higher education can receive during the examination (examination) is 40 points. The minimum amount for which the exam is considered passed is 25 points.

The final assessment by course is determined by summing the points for the current and final control.

During the teaching of the course, the following assessment measures are used:

Current assessment: defense of laboratory work (40 points), written control work (testing) (20 points).

Semester assessment: Exam (40 points).

More detailed information on the assessment system is provided in technological card of the course.

An example of an exam card and assessment criteria.

Simon Kuznets Kharkiv National University of Economics
First (bachelor) level of higher education
Specialty "Software Engineering"
Educational and professional programme "Software Engineering"
Semester 6
Course "Software Engineering"

EXAM CARD No. 1

Task 1
Development of the automated module "According to the movement of personnel at the enterprise".

The developed automated module is proposed for implementation at private and public enterprises to increase the productivity of personnel department specialists. Achieving the set goal is carried out by implementing an automated solution to the following tasks:

hiring employees;
relocation of employees;
dismissal of employees;
analysis of the personnel status of the enterprise.

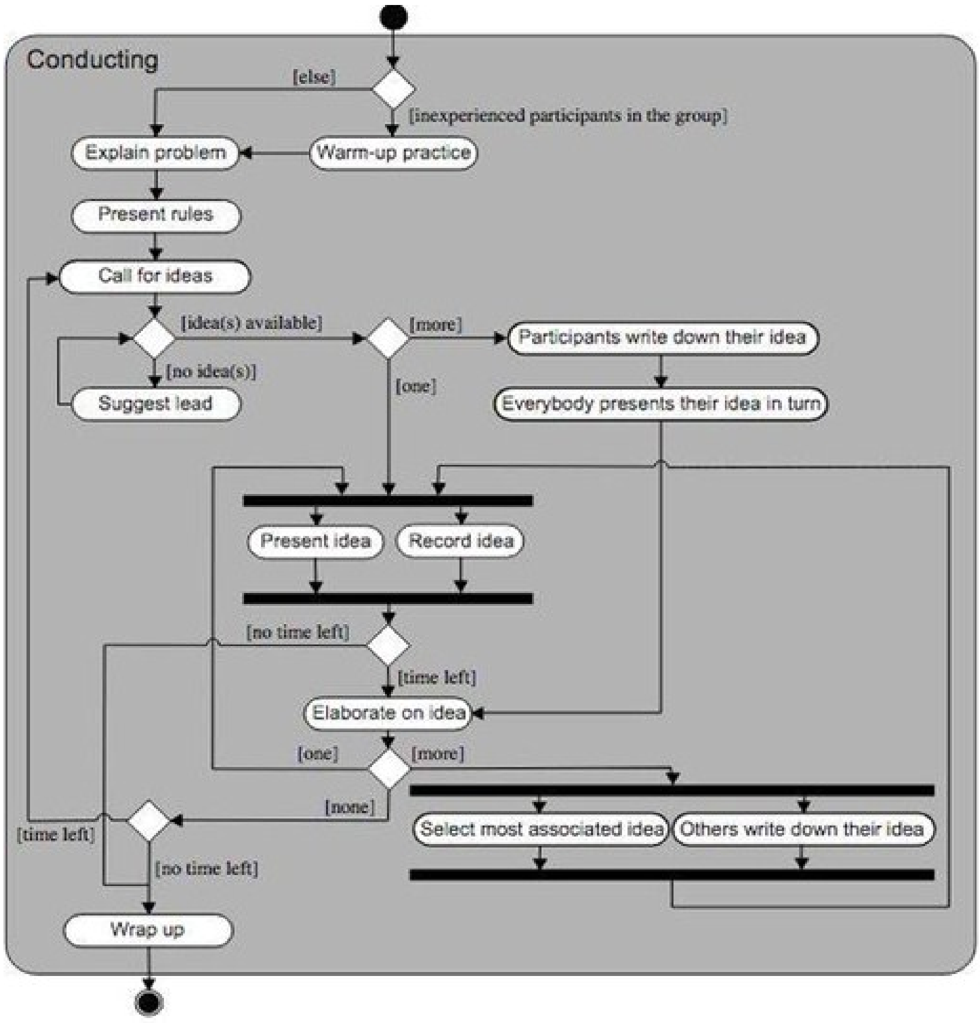
In the developed module, it is necessary to provide for the identification of system users. The end user must be able to fully perform operations with data on workers, positions, divisions, and articles of layoffs, to keep track of hired, transferred, and dismissed employees.

Based on this information, analyze the personnel situation of the enterprise, as well as form information on hired, transferred, and dismissed employees for the period with the possibility of printing them.

Task:
build a diagram of use cases for a given subject area;
describe two use case scenarios from among those shown in the use case diagram
build an activity chart,
construct a class diagram.

Task 2

On the given diagram, indicate the type of diagram, names of structural elements and designations, and describe in detail what exactly the diagram displays.



Include in the report file: 1. Constructed diagrams: Use-case, activity, and classes; a table with a description of scenarios of use cases; 2. description of the diagram and its structural elements.

Protocol No. ____ dated "___"__________20___ was approved at the meeting of the Information Systems Department

Examiner PhD, Prof. Zolotaryova I.O.
Chief of the Department Ph.D., Assoc. Prof. Bondarenko D.O.

Assessment criteria

The final mark for the exam consists of the sum of the marks for the completion of all tasks, rounded to a whole number according to the rules of mathematics.

The algorithm for solving each task includes separate stages that differ in complexity, time-consumingness, and importance for solving the task. Therefore, individual tasks and stages of their solution are evaluated separately from each other as follows:

The first task is evaluated from 0 to 30 points according to the following components:

10 points – correct and complete construction of the use case diagram and correct and complete description of the scenarios of the two use cases;

10 points – correct and complete construction of the class diagram;

10 points - correct and complete the construction of the activity diagram.

If the parts of the task described above are not fully completed, 1 point is deducted from the maximum score.

The second task is evaluated from 0 to 10 points according to the following components:

10 points – a correct and complete description of the components of the given diagram. If parts of the task are not fully completed, 1 point is deducted from the maximum score.

RECOMMENDED LITERATURE

Main

1. Fundamental Approaches to Software Engineering. 22nd International Conference, FASE 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6–11, 2019, Proceedings
2. UML-Based Software Product Line Engineering with Smarty. Published by Springer International Publishing, 2023, https://doi.org/10.1007/978-3-031-18556-4
3. I. Borodkina, G. Borodkin, Software engineering. Guide for students of higher educational institutions - K., "Center for educational literature" - 2018.-204 p.
4. Software Engineering at Google: Lessons Learned from Programming Over Time by Titus Winters (Author), Tom Manshreck (Author), Hyrum Wright (Author). O'Reilly Media; 1st edition (April 7, 2020).
5. Modern Software Engineering. Doing What Works to Build Better Software Faster. David Farley. 2021. Addison-Wesley Professional.
6. Maxwell K. John Five levels of leadership.: Trans. from English T. Kuripko. – Kh.: Ranok Publishing House: “Fabula”, 2019. – 304 p.
7. Joseph Heagney. Basics of project management. Plot. - 2020. - 272 p.
8. Gregory M. Horine. Project Management Absolute Beginner's Guide 5th Edition, 2022 https://a.co/d/7oJCpuT
9. Richard Newton. Project management from A to Z. Alpina Publisher, - 2018. - 180 p.
10. Postil S. D. UML. unified language of modeling information systems / S. D. Postil: State University. fisk. services of Ukraine. - Irpin: State University fisk. services of Ukraine, 2019. - 321 p.
11. Martin Fowler. UML. Basics 3rd edition / Martin Fowler, Kendall Scott - M.: Simbol-plus, 2018. - 192 p.

12. Robert M. Pure architecture: the art of software development" / Robert Martin, Fabula, 2019. - 416 p.

13. Modern information technologies and systems [Electronic resource]: monograph / N. G. Aksak, L. E. Gryzun, O. V. Shcherbakov [and others]; in general ed. V. S. Ponomarenko — Electron. text data (22.9 MB). — Kharkiv: HNEU named after S. Kuznetsia, 2022. — 270 p. - Access mode:http://www.repository.hneu.edu.ua/handle/123456789/29233.

14. Yuriy Skorin, Iryna Zolotaryova. Enhancing the effectiveness of usability testing for user interfaces // International scientific journal “Computer systems and information technologies”. No. 3. – Khmelnitskyi: Khmelnytskyi National University, 2023. – P. 65–74. - Access mode:http://www.repository.hneu.edu.ua/handle/123456789/30712.

15. SWEEBOK (Software Engineering Body of Knowledge)https://www.computer.org/education/bodies-of-knowledge/software-engineering.

16. ISO/IEC/IEEE 90003:2018 Software engineeringhttps://www.iso.org/standard/74348.html

17. ISO/IEC/IEEE 12207:2017 Systems and software engineering Software life cycle processes https://www.iso.org/standard/63712.html

Information resources

Requirements management tools and standards:

- Doors Next
- ReqView
- Vision Requirements Management
- ISO/IEC/IEEE International Standards
- Volere Process
- Mastering the Requirements Process

UML tools:

- UML Resource Center at IBM Rational
- Agile Modeling (http://www.agilemodeling.com/), also refer to Extreme Programming (http://www.extremeprogramming.org/).
- http://www.objectsbydesign.com/
- A nice page on how to choose a UML modeling tool
- OMG's UML resource page:http://www.omg.org/uml/
- Unified Modeling Language (UML)
- Common Warehouse Metamodel (CWM)
- IBM Technical Journals
- http://www.holub.com/goodies/uml/index.html
- http://www.smartdraw.com/resources/centers/uml/uml.htm
- https://github.com/Alliedium/awesome-software-engineering?tab=readme-ov-file