Syllabus of the educational component

Program of the educational course

Software Development

Specialty 113 Applied mathematics	Institute Educational and Scientific Institute of Computer Science and Information Technology
Educational program Intellectual data analysis	Department Computer Mathematics and Data Analysis
Level of education Bachelor's level	Course type Special (professional), Selective
Semester 6	Language of instruction Ukrainian

Lecturers and course developers

Eduard Georgiyovych Fastovskyi

Eduard.Fastovskyi@khpi.edu.ua

Senior teacher of the Department of Computer Mathematics and Data Analysis of the National Technical University "KhPI"

Work experience – 18 years. An author of more than 14 scientific and scientific-methodological paperworks. Leading lecturer of the courses such as "Java Principles and Paradigms" and "Corporate Java Technologies". Experience in IT – 15 years.

More about the lecturer on the department's website

General information

Summary

The course focuses on the study of key aspects of the software development process, including the architectural design, programming, testing, and project management. Students with gain in-depth knowledge and skills that are required for successful implementation of software projects in real-world development environments.

Course objectives and goals

Develop the students' in-depth knowledge and skills in software engineering for effective implementation of software products, including architectural design, programming, testing, and project management.

Format of classes

Lectures, laboratory works, independent work, and consultations. Final control - test.

Competencies

GC 1. Ability to learn and master modern knowledge.
GC 2. Ability to apply knowledge in practical situations.
GC 7. Ability to search, process, and analyze information from various resources.
GC 8. Knowledge and understanding of the subject area and understanding of professional activities.
GC 10. Skills in the use of information and communication technologies.
SC 5. Ability to develop algorithms and data structures, software tools and program documentation.

SC 6. Ability to design databases, information systems and resources.

Learning outcomes

LO 11. Be able to apply modern programming technologies and software development, software implementation numerical and symbolic algorithms.
LO 12. Solve individual engineering problems and/or tasks that arise in at least one subject area: sociology, economy, ecology, and medicine.
LO 13. To use specialized software programs in practical work products and software systems for computer mathematics.
LO 14. Demonstrate the ability to self-study and continue professional development.
LO 15. Be able to organize your own activities and get results within a limited time frame.
LO 16. Demonstrate skills of interaction with other people, ability to work in a team.

Student workload

Overall scope of the course – 150 hrs. (5 credits ECTS): lectures – 32 hrs., laboratory works – 32 hrs., independent work – 86 hrs.

Course prerequisites

«Object-oriented programming», «Analysis of requirements for software systems», «Databases and information systems».

Features of the course, teaching and learning methods, and technologies

Educational materials are available to students on the teacher's OneDrive resource.

Program of the course

Topics of the lectures

Topic 1. Introduction
The purpose of the course, the objectives of the course, the main concepts.
Topic 2. Basics of the unified modeling language (UML)
History. Purpose. Types of diagrams. Behavioral diagrams - purpose and features of their construction.
Topic 3. UML language: Structural diagrams and behavioral diagrams.
Purpose of diagrams and features of their construction.
Topic 4. Basic principles of object-oriented programming and SOLID design
History. Purpose. Types of basic principles.
Topic 5. Introduction to the GoF design patterns.
History. Purpose. Types of patterns. Creational patterns.
Topic 6. GoF design patterns
Structural and behavioral patterns.
Topic 7. Architectural patterns. Client-server architecture.
Purpose. Types of patterns. Features and types of the client-server architecture.
Topic 8. Architectural patterns. Service-oriented architecture. Microservices.
Features of the architectural patterns, advantages and disadvantages.
Topic 9. Architecture of web and server applications.
Design and development.
Topic 10. Architecture of mobile applications.
Design and development.
Topic 11. Project build tools and deployment automation.
Role and configurations of project build tools, particularly Apache Maven, software deployment automation techniques to improve development efficiency and stability.
Topic 12. Source code version control system (VCS)
Purpose. Types. Basic commands. Work with branches: creation strategy, merging, and conflict resolution.
Topic 13. Bug tracking system (bug-trackers)

Basics of using bug trackers to effectively detect, track, and manage bugs in software development.
Topic 14. Introduction to cloud computing
Introduction to the basic principles and architectural components of cloud systems.
Topic 15. Docker: software containerization
Basic concepts of Docker. Creation, configuration and management of containers.
Topic 16. Introduction to software testing
Strategies, types of tests, and use of tools for ensuring code quality.

Topics of the workshops

Practical classes within the course are not provided.

Topics of the laboratory classes

Topic 1. Environment setup
Download, installation and initial setup of required software.
Topic 2-3. Analysis and design of the software application.
Choosing a subject area for application development. Analysis and design of functionality using the UML language.
Topic 3. Work with a bug tracker
Creation of a list of tasks for development in the selected bug tracker.
Topic 4. Preparational steps before development
Choosing application architecture. Selection of the necessary programming languages and technology stack for application implementation.
Topic 5. Work with a source code version control system
Selection and configuration of the selected system.
Topic 6. Project structure design
Creation of the project structure, taking into account the features of the selected project build tool, breakdown into modules. Connection of necessary libraries and frameworks.
Topic 7-10. Application implementation
Application development.
Topic 11-12. Project retrospective and refactoring
Analysis of the implemented code for compliance with SOLID principles and the possibility of using GoF templates. Refactoring according to the analysis results.
Topic 13-14. Application testing and bug fixing
Application testing. Bugs creation in bug tracker. Bug fixing.
Topic 15-16. Application containerization
Application deployment (and/or its separate modules) in Docker.

Self-study

The course involves the implementation of an individual calculational task. You can choose one of the following options:

- Creation of a UML model for a specific project: the student can choose or present their own project and develop a complete UML model, including class diagrams, sequences, interactions, etc.;
- Analysis and optimization of code according to SOLID principles: the student can select a specific code (for example, their own or an existing project) and make changes to comply with SOLID principles;
- Development of a microservice according to an architectural pattern: a student can develop a simple microservice using the architectural pattern that was discussed in the lectures;
- Creation of a Docker container for an application: a student can take an application (can be a simple web service or a console application) and create a Docker container to deploy and run it;
- Creation and an implementation of a test strategy: the student can develop a test strategy for a specific project, including unit tests, integration tests and acceptance tests.

Non-formal education

In particular, individual topics of this component may be taken into account in case of successful completion of the following courses:

Topic 2, 3. “Basics of the unified modeling language (UML)”
https://www.udemy.com/course/unified-modeling-language-uml-course-uml-diagram-software-enginnering/?couponCode=ST21MT61124

Topic 4. “Basic principles of object-oriented programming and SOLID design”
https://www.udemy.com/course/solid-principles-cj/

Topic 5, 6. “Introduction to the GoF design patterns”
https://www.udemy.com/course/gof-design-patterns-learnit/

Topic 7. “Architectural patterns. Client-server architecture”
https://www.udemy.com/course/web-development-learnit/

Topic 8. “Architectural patterns. Service-oriented architecture. Microservices”
https://www.udemy.com/course/understanding-microservices-architecture/

Topic 12. “Source code version control system (VCS)”
https://www.udemy.com/course/git-expert-4-hours/

Topic 13. “Bug tracking system (bug-trackers)”
https://www.udemy.com/course/bug-tracking-with-jira-jira-for-softwareqa-testers/

Topic 14. “Introduction to cloud computing”
https://www.udemy.com/course/introduction-cloud-computing/

Topic 15. “Docker: software containerization”
https://www.udemy.com/course/diveintodocker/

Topic 16. “Introduction to software testing”
https://www.udemy.com/course/introduction-to-software-testing-or-software-qa/
https://www.udemy.com/course/software-testing-fundamentals-w/

Course materials and recommended reading

Basic literature

1 R. Martin Clean architecture. The art of software development. - Fabula, 2019. - 368 p.
https://www.yakaboo.ua/ua/chista-arhitektura.html
2 I. Borodkina, G. Borodkin. Software engineering. Study guide for students of higher educational institutions. - Center of educational literature, 2020. - 204 p.
http://library.kpi.kharkov.ua/files/new_postupleniya/inprza.pdf (1 manual)
3 R. Melnik. Web application programming (frontend and backend). - Lviv Polytechnic, 2018. - 248 p.
https://www.yakaboo.ua/ua/programuvannja-veb-zastosuvan-front-end-ta-bek-end.html
4 Eric Freeman, Elizabeth Robson, Burt Bates, Kathy Sierra. Head First. Патерни проєктування. - Фабула, 2020. - 672 p. https://www.yakaboo.ua/ua/head-first-paterni-proektuvannja-pdf.html
5 Richardson C. Microservices Patterns With examples in Java. - Manning, 2018. - 520 p.
https://www.amazon.com/Microservices-Patterns-examples-Chris-Richardson/dp/1617294543
6 Freeman E., Robson E.. Head First Design Patterns, 2nd Edition. - O’Reilly, 2021. - 669 p.
https://www.amazon.com/Head-First-Design-Patterns-Object-Oriented/dp/149207800X
7 Miell I., Sayers H. Aidan. Docker in Practice 2nd Edition. - Manning Publications Co., 2019. - 384 p.
https://balka-book.com/razrabotka_programnogo_obespecheniya-366/docker_in_practice-111324
8 Oggl B., Kofler M. Docker: Practical Guide for Developers and Devops Teams. - Rheinwerk Computing, 2023. - 492 p. https://balka-book.com/razrabotka_programnogo_obespecheniya-366/docker-practical-guide-for-developers-and-devops-teams-263487

9 Newman S. Building Microservices: Designing Fine-Grained Systems 2nd Edition, - O'Reilly, 2021. - 500 p. https://balka-book.com/razrabotka_programnogo_obespecheniya-366/building_microservices_designing_fine_grained_systems-114667

Additional literature

1 R. Martin. Clean code. - Fabula, 2019. - 448 p. https://www.yakaboo.ua/ua/chistij-kod.html
2 Fowler M., UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3rd Edition. - Print2print, 2016. - 208 p. https://balka-book.com/uml_shablonyi_proektirovaniya_programmnogo_obespecheniya-331/uml_distilled_a_brief_guide_to_the_standard_object_modeling_language_3rd_edition-32903
3 Gamma E. Design Patterns: Elemens of Reusable Object-Oriented Software. - Addison-Wesley Professional, 1995. - 396 p. https://balka-book.com/ua/razrabotka_programnogo_obespecheniya-366/design_patterns_elements_of_reusable_object_oriented_software-14417

Assessment and grading

Criteria for assessment of student performance, and the final score structure

100% of the final grade consists of:

- control works — 20% of the semester grade
- laboratory works — 30% of the semester grade
- independent work — 30% of the semester grade
- individual assignments — 20% of the semester grade

Evaluation scale

Total points	National	ECTS
90–100	Excellent	A
82–89	Good	B
75–81	Good	C
64–74	Satisfactorily	D
60–63	Satisfactorily	E
35–59	Unsatisfactorily (requires additional learning)	FX
1–34	Unsatisfactorily (requires repetition of the course)	F

Norms of academic integrity and course policy

The student must adhere to the Code of Ethics of Academic Relations and Integrity of NTU «KhPI»: to demonstrate discipline, good manners, kindness, honesty, and responsibility. Conflict situations should be openly discussed in academic groups with a lecturer, and if it is impossible to resolve the conflict, they should be brought to the attention of the Institute's management.
Regulatory and legal documents related to the implementation of the principles of academic integrity at NTU «KhPI» are available on the website: http://blogs.kpi.kharkov.ua/v2/nv/akademichna-dobrochesnist/

Approval

Approved by	Date, signature 29.08.2024		Head of the Department Olena AKHIIEZER
	Date, signature 29.08.2024		Guarantor of the Educational Program Olena AKHIIEZER