# SYLLABUS OF A MODULE

| Polish name of a module | **Inżynieria oprogramowania** |
|---|---|
| English name of a module | **Software engineering** |
| ISCED classification - Code | 0613 |
| ISCED classification - Field of study | *Software and applications development and analysis* |
| Languages of instruction | *English* |
| Level of qualification: | *1 – BSc (EQF 6)* |
| Number of ECTS credit points | *6* |
| Examination: | *EW – exam written* |
| Available in semester: | *A – Autumn only* |

## Number of hours per semester:

| Lecture | Tutorials | Laboratory | Seminar | E-learning | Project |
|---|---|---|---|---|---|
| 30 | 0 | 30 | 0 | 0 | 0 |

## MODULE DESCRIPTION

### Module objectives

O1. To familiarize students with the course of the software production process, starting from the strategic phase, through discovering user requirements to the final stages, i.e. user testing and maintenance.

O2. To acquire practical skills in software design by students.

## PRELIMINARY REQUIREMENTS FOR KNOWLEDGE, SKILLS AND OTHER COMPETENCES

1. Knowledge of the mathematics and programming fundamentals.
2. Knowledge of the most popular programming paradigms: procedural and object-oriented.
3. The ability to use various sources of information including instructions and technical documentation.
4. The ability to work individually and in a group.

5. The ability to properly present their own actions.

**LEARNING OUTCOMES**

LO 1 – The student has knowledge of actions taken at all stages of software development, i.e. requirements engineering, analysis, design, verification and validation of software.

LO 2 – The student has skills to use appropriate notation and software engineering techniques in the process of creating computer systems.

**MODULE CONTENT**

| Type of classes – lecture | Number of hours |
|---|---|
| **Lec 1** - Basic concepts and goals of software engineering. | **2** |
| **Lec 2** - Models of software development process. | **2** |
| **Lec 3** - Requirements engineering process. | **2** |
| **Lec 4** - Introduction to UML. | **2** |
| **Lec 5** - UML – structural diagrams. | **2** |
| **Lec 6** - UML – behavioral diagrams. | **2** |
| **Lec 7** - **8**- Methods for identifying classes and objects in the project. | **4** |
| **Lec 9** - Typical computer system architectures. | **2** |
| **Lec 10** - Introduction to design patterns. | **2** |
| **Lec 11** - Discussion of selected design patterns. | **2** |
| **Lec 12**- Software verification and validation process. | **2** |
| **Lec 13** - Test automation methods. | **2** |
| **Lec 14** - Agile programming techniques. | **2** |
| **Lec 15** - Management basics of software projects. | **2** |
| **Sum** | **30** |
| **Type of classes– laboratory.** | **Number of hours** |
| **Lab 1** - Familiarization with the concepts of software engineering. | **2** |
| **Lab 2**- Familiarization with the CASE tool used in the laboratory. | **2** |

| | |
|---|---|
| **Lab 3**- Requirements specification for a sample project. | **2** |
| **Lab 4**- Preparation of use cases based on requirements specification. | **2** |
| **Lab 5**- Use case scenarios, alternative scenarios, exceptions. | **2** |
| **Lab 6** - Activity diagrams for use cases. | **2** |
| **Lab 7** - Class identification based on user stories. | **2** |
| **Lab 8** - Class relationships: generalization, association, aggregation and composition. | **2** |
| **Lab 9 - 10** - Preparation of the class diagram for a sample project. | **4** |
| **Lab 11** - Creating documentation for a given source code. | **2** |
| **Lab 12** - The use of selected UML diagrams in software design. | **2** |
| **Lab 13** - Software architecture compliant with the Model-View-Controller pattern. | **2** |
| **Lab 14** - Sample implementations of selected design patterns. | **2** |
| **Lab 15** - Software testing - unit tests. | **2** |
| **Sum** | **30** |

## TEACHING TOOLS

| |
|---|
| **1. -** lectures with multimedia presentations |
| **2. -** laboratory exercises with provided instructions |
| **3. -** computer lab with CASE software installed |

## WAYS OF ASSESSMENT ( F – FORMATIVE, S – SUMMATIVE

| |
|---|
| **F1. -** assessment of preparation for laboratory exercises |
| **F2. -** assessment of the ability to apply the acquired knowledge while doing the exercises |
| **F3. -** evaluation of reports on the implementation of exercises covered by the curriculum |
| **F4. -** assessment of activity during classes |
| **S1. -** assessment of the ability to solve the problems posed and the manner of presentation obtained results - pass mark * |
| **S2. -** assessment of mastery of the teaching material being the subject of the lecture - exam |

*) in order to receive a credit for the module, the student is obliged to attain a passing grade in all laboratory classes as well as in achievement tests.

**STUDENT'S WORKLOAD**

| L. p. | Forms of activity | Average number of hours required for realization of activity |
|---|---|---|
| 1. | **Contact hours with teacher** | |
| 1.1 | Lectures | 30 |
| 1.2 | Tutorials | |
| 1.3 | Laboratory | 30 |
| 1.4 | Seminar | |
| 1.5 | Project | |
| 1.6 | Consulting teacher during their duty hours | |
| 1.7 | Examination | 3 |
| | Total number of contact hours with teacher: | 63 |
| 2. | **Student's individual work** | |
| 2.1 | Preparation for tutorials and tests | |
| 2.2 | Preparation for laboratory exercises, writing reports on laboratories | 34 |
| 2.3 | Preparation of project | |
| 2.4 | Preparation for final lecture assessment | |
| 2.5 | Preparation for examination | 23 |
| 2.6 | Individual study of literature | 30 |
| | Total number of hours of student's individual work: | 87 |
| | Overall student's workload: | 150 |
| **Overall number of ECTS credits for the module** | | **6** ECTS |
| Number of ECTS points that student receives in classes requiring teacher's supervision: | | 2.6 ECTS |
| Number of **ECTS** credits acquired during practical classes including laboratory exercises and projects: | | 2.2 ECTS |

**BASIC AND SUPPLEMENTARY RESOURCE MATERIALS**

| |
|---|
| 1. Gamma et al.: Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995 |
| 2. Miles R., Hamilton K.: Learning UML 2.0, O'Reilly Media, 2006 |
| 3. Pressmann R., Maxim B.: Software Engineering: A Practitioner's Approach, McGraw-Hill Education, 2019 |
| 4. Sommerville I.: Software Engineering, Pearson, 2015 |
| 5. McConnell S.: Code Complete: A Practical Handbook of Software Construction, Microsoft Press, 2004 |
| 6. Bruegge B., Dutoit A.: Object-Oriented Software Engineering Using UML, Patterns, and Java, Pearson, 2009 |

**MODULE COORDINATOR ( NAME, SURNAME, E-MAIL ADDRESS)**

| |
|---|
| Robert, Dyja, robert.dyja@icis.pcz.pl |