MODULE OUTLINE

1. GENERAL INFORMATION

SCHOOL	SCHOOL OF SCIENCE AND TECHNOLOGY		
PROGRAM COURSE	INFORMATICS		
LEVEL OF STUDY	UNDERGRADUATE		
MODULE CODE	PLI-42	**YEAR OF STUDY**	4th
MODULE TITLE	Special Issues in Software Engineering		
INDEPENDENT TEACHING ACTIVITIES *in case credits are awarded for separate components/parts of the course, e.g. in lectures, laboratory exercises, etc. If credits are awarded for the entire course, give the weekly teaching hours and the total credits*		**HOURS**	**CREDIS**
Weekly teaching hours * 32 weeks		16-18	20 ECTS
COURSE TYPE *Background knowledge, Scientific expertise, General Knowledge, Skills Development*	Scientific expertise Optional mandatory instead of PLI23, PLI40, PLI47		
PREREQUISITE MODULES:	No		
LANGUAGE OF INSTRUCTION AND EXAMS	GREEK		
THE MODULE IS OFFERED TO ERASMUS STUDENTS	No (due to annual duration of the module)		
MODULE WEBSITE (URL)	https://www.eap.gr/education/undergraduate/computer-science/topics/#eidik_them_log Each module has its own space in the Learning Management System of EAP (http://study.eap.gr), with controlled access (use of code) for students and teaching staff.		

2. LEARNING OUTCOMES

Learning Outcomes

- *The course learning outcomes, specific knowledge, skills and competences of an appropriate (certain) level, which students will acquire upon successful completion of the course, are described in detail. It is necessary to consult:*

FORMAL SPECIFICATIONS

Upon successful completion of the module, the student will be able to:

- Describe and explain the term “software specifications”.
- Reference three categories of specifications depending on the degree of formality: formal, semi-formal and non formal.
- Indicate the major semi-formal specification production techniques.

- State the features of formal specifications' languages
- Have background knowledge on the mathematical tools that are the basis for formal specifications' languages, such as the propositional calculus, the predicate calculus, logic functions and the set theory.
- Explain and apply the axiomatic and algebraic specifications to describe a piece of software and its functions using equations, recurrence relations and regular expressions.
- Write and interpret Z schemata, as well as basic symbols and operators
- Specify using Z schemata the invariance or the change of a software module state, operations, error handling procedures and data manipulation functions.
- Create complex shapes using the Z notation language and specify complex software operations using functions, sequences and multi-sets.
- Give the formal definition of Petrinets and describe their basic concepts, rules and properties.
- Identify and design Petrinets of three different categories: treaty-event, position-transition and networks of discrete items..
- Use effectively tools for designing Petrinets and their practical application.
- Overturn myths about the application of formal specifications.
- Enumerate and apply the 10 commandments of proper practical application of formal techniques.

SOFTWARE VERIFICATION AND VALIDATION
Upon successful completion of the module,the student will be able to:

- Discern software verification and validation (V&V) activities and understand their position in the software lifecycle.
- Discern the static informal V&V techniques from the dynamic informal V&V techniques
- Describe the static V&V techniques (static analysis, walkthrough, review, and more) and know their objectives, application points and the respective advantages of each technique.

- Describe and explain the three phases of testing: unit testing, integration testing and validation testing.
- Describe and explain the three phases of testing: unit testing, integration testing and validation testing.
- Explain and effectively apply the most important techniques for designing test cases for the functional testing of software (black box testing) such as: equivalence partitioning, boundary value analysis, cause-effect graph.
- Explain and effectively apply the most important techniques for designing test cases for the structural testing of software (white box testing) such as: basic path testing, loop testing.
- Explain and effectively apply the most important techniques for designing test cases for testing the interfaces between the components that make up a software system

- Know the most important issues for the practical application of various software testing techniques, such as top-down integration testing and bottom-up, alpha and beta testing, performance testing, recovery testing.andmore.

- Explain and effectively apply the most important debugging methods following a successful testing phase in order to correct the identified bugs, such as brute force techniques, backtracking and techniques for locating the cause of the bug.
- Design and apply performance testing (load & stress testing) for multi-user applications and web applications in particular.

SOFTWARE MANAGEMENT AND SOFTWARE QUALITY
Upon successful completion of the module, the student will be able to:

- Know what software quality is and how it is assured.
- Discern the differences in the process of software quality assurance compared to other products.
- Know the prevalent models of software quality assurance and effectively implement the ISO9126. standard.
- Know what are the internal and external software quality metrics, which is the procedure of measuring them and to what extent they are interrelated.
- Know and effectively apply the Halstead metrics and interpret the meaning of the results.
- Know the fundamental problem and objectives of the Human Computer Interaction field, as well as the basic interaction modelling theories (such as the human processor model, the keystroke level model, Fitts, Hick–Hyman and Power Law of Practice).
- Know and effectively apply the most widely used evaluation methods (heuristic evaluation, cognitive walkthrough, user observation, questionnaires, performance measurement).
- Know and effectively apply the basic techniques for collecting, analysing and interpreting quantitative data in the context of the usability evaluation of an interactive application.
- Select the appropriate combination of usability evaluation methods depending on the type of the interactive application and the stage of its development.

General Competences

Taking into consideration the general competences that students/graduates must acquire (as those are described in the Diploma Supplement and are mentioned below), at which of the following does the course attendance aim?

Search for, analysis and synthesis of data and information by the use of appropriate technologies,
Adapting to new situations
Decision-making
Individual/Independent work
Group/Team work
Working in an international environment
Working in an interdisciplinary environment
Introduction of innovative research
Project planning and management
Respect for diversity and multiculturalism
Environmental awareness
Social, professional and ethical responsibility and sensitivity to gender issues
Critical thinking
Development of free, creative and inductive thinking ……
(Other…….citizenship, spiritual freedom, social awareness, altruism etc.) …….

Search for, analysis and synthesis of data and information by the use of appropriate technologies
Decision-making
Individual/Independent work

Project planning and management Development of free, creative and inductive thinking

3. MODULE CONTENT

The aim of the course is to offer advanced knowledge and skills related to Software Engineering (Software Technology). In some aspects, it is a continuation of the course PLI11 and to a lesser extent of the course PLI24, especially with respect to the topics described in the book "Software Technology II". Subjects of the module: 1. Formal Specifications 2. Software Validation and verification 3. Software Management & Quality with special emphasis on Human Computer Interaction

4. TEACHING METHODS--ASSESSMENT

<table>
<tr><td>MODES OF DELIVERY
Face-to-face, in-class lecturing, distance teaching and distance learning etc.</td><td>Distance education with five Group Counseling Meetings (OSS) during the academic year on weekends.</td></tr>
<tr><td>USE OF INFORMATION AND COMMUNICATION TECHNOLOGY
Use of ICT in teaching, Laboratory Education, Communication with students</td><td>We use :
● Video conferencing tools (webex),
● Presentation software (powerpoint, google slides),
● E-learning platforms (study.eap.gr)
● Specialized software in the subjects taught(C and/or Java compilers, Microsoft project, Keystroke Level Model Form Analyzer to estimate task completion times, HP Petri Sim to implement petri net modeling).

Additionally, the students use office automation tools, web browsers and e-reader for digital books.</td></tr>
<tr><td>MODULE DESIGN
Description of teaching techniques, practices and methods: Lectures, seminars, laboratory practice, fieldwork, study and analysis of bibliography, tutorials, Internship, Art Workshop, Interactive teaching, Educational visits, projects, Essay writing, Artistic creativity, etc</td><td>
<table>
<tr><th>Activity</th><th>Annual Workload</th></tr>
<tr><td>5 OSS (* 4 hours)</td><td>20</td></tr>
<tr><td>Preparation of Assignments (5 assignments * 10 hours)</td><td>50</td></tr>
<tr><td>Examination</td><td>3</td></tr>
<tr><td>Individual study</td><td>439-503</td></tr>
<tr><td>Total module workload (hours)</td><td>512-576</td></tr>
</table>
</td></tr>
</table>

The study hours for each learning activity as well as the hours of selfdirected study are given following the principles of the ECTS.	
STUDENT PERFORMANCE EVALUATION/ASSESSMENT METHODS *Detailed description of the evaluation procedures.* *Language of evaluation, assessment methods, formative or summative (conclusive), multiple choice tests, short- answer questions, open-ended questions, problem solving, written work, essay/report, oral exam, presentation, laboratory work, other……etc.* *Specifically defined evaluation criteria are stated, as well as if and where they are accessible by the students*	Elaboration of written assignments during the academic year, the average of the grades of which participates in the formation of the final grade of module by 30%, if there is a passable in the final or repetitive examinations. In the final written exams the grade of the written assignments participates in the formation of the final grade of module by 70%. All the criteria are posted, both in each written assignment (in the LMS study.eap.gr), as well as in the general regulation of HOU at: https://www.eap.gr/education/study-regulations/

(6) SUGGESTED BIBLIOGRAPHY

- *Suggested bibliography:*

HOU Publications:

1. *Volume AD: Formal Specifications, EAP, Patras. PLI42 / 1*
2. *Volume II: Software Validation, EAP, Patra 2003. PLI42 / 2*
3. *Volume DG: Software Management and Quality, EAP, Patra 2003. PLI42 / 3*
4. *Accompanying Text (37 pages) for the course PLI 42: Christodoulou S., Xenos M., Performance Control*

Web Applications

5. *"Accompanying Text (79 pages) for the course PLI 42: Tselios N., Xenos M., "Human Computer Interaction"*
6. *Accompanying Text (50 pages) for the course PLI 42: Vassiliadis V., Xenos M., "Project and Financial analysis"*
7. *Additional digital (and multimedia) material is located inside the platform study (study.eap.gr).*

Additional digital material (and multimedia) is available through study platform.

-Related scientific Journals:

1) IEEE Transactions on Software Engineering
2) ACM Transactions on Programming Languages and Systems
3) ACM Transactions on Computer-Human Interaction (TOCHI)
4) International Journal of Human-Computer Studies
5) IEEE Transactions on Human-Machine Systems
6) Interacting with computers