Course Title	Software Reuse				
Course Code	WSS553				
Course Type	Specialization (Elective)				
Level	Master (2nd Cycle)				
Year / Semester	2 or 3				
Teacher's Name	Achilleas Achilleos, PhD				
ECTS	10	Lectures / week	3	Laboratories/week	0
Course Purpose	The aim of this course is to provide students with critical understanding of the technology, issues and challenges of software reuse at various levels. Specific focus in the course is dedicated to software reuse in web-based systems accessible via mobile devices. The course will enable students to practice software reuse at various levels, with different programming languages and on different platforms. In specific, the use of Java and HTML5 technologies will provide the capability to experience and practice software reuse on both desktop and mobile platforms, as well as at different levels such as object-oriented programming, component-based software development, middleware- based development, WS*-stack services, REST services and model- driven engineering. Finally, management of code repositories is introduced at the last week. In overall, the objective of the course is to enhance critical awareness, promote practical thinking and reasoning to solve practical problems through the reuse of software systems.				
Learning Outcomes	Upon successful completion of the course students will be able to: • Understand the concepts, principles and methods of software reuse and argue on the importance of software reuse in building modern software systems. • Outline and describe the different levels of software reuse: object-oriented programming, component-based engineering, middleware, WS*-stack services, REST services and model-driven engineering. • Identify, analyse and reuse open-source software tools in practice and at different software reuse levels. • Gain theoretical knowledge and analytical skills to develop applications by employing reuse methods at code, component, design and models levels. • Distribute effectively the results of their work to other developers using software repositories in order to promote software reuse. • Describe and explain the concept of open-source software development and argue on the importance of software licensing.				
Prerequisites	**None.**		Corequisites	**None.**	
Course Content	**1. Introduction to Software Reuse (2 Weeks).**				

*One hour out of the three is normally devoted to laboratory-based programming exercises.

	- Software Reuse Key Concepts. Levels and Types of Software Reuse. The Software Reuse Landscape. Software Reuse Approaches. Reuse Benefits, Issues and Economics. **2. Object Oriented Programming and Component Based Software Engineering (2 Weeks).** - Revisiting key concepts of Object Oriented Programming (OOP). Practical example of reuse through OOP. Reuse through the Java Collections Framework. Introduction to the principles and concepts of Component Based Software Engineering (CBSE). JavaBeans: Software Reuse at the level of CBSE. Practical example of reuse through JavaBeans. **3. Design Patterns: Reusing Best Practices to Solve Common Design Problems (4 Weeks).** - Design Principles and Patterns. Design Patterns: Concepts and Types. Building Successful Mobile Applications using Design Patterns. **4. Software Reuse via the notion of a Middleware (1 Week).** - Motivation, definition and the role of a middleware. Examining a simple middleware architecture: RPC. Challenges in middleware design. Example: HTML5 Context Middleware (H5CM). **5. Service Reusability (2 Weeks).** - Motivation, History and Concepts. The Web Service Model. Web Service Standards - WS*-stack (WSDL, SOAP, XML, UDDI). RESTful Services. REST Motivation, Definition and Principles. REST Vocabulary and Concepts. REST Vs. WS*- stack. **6. Model Driven Engineering (1 Week).** - Introduction to the notion of models reuse. Unified Modelling Language and Domain Specific Modelling. Model-driven engineering and MDA architecture. Models transformation and code generators. **7. Software Repositories (1 Week).** - Definition. Reusing Software Assets. Requirements and Advantages of a Software Repository. The Software Repository Model. Main functions of a Software Repository. Version Control Systems. Creating and Managing a Software Repository. Open-Source Software Development. Software Licensing.
Teaching Methodology	The methodology followed in this course is structured around lectures and laboratory exercises, so that students gain theoretical knowledge as well as practical skills. The taught part of course is delivered to the students with the help of computer presentations. Presentations are available through the e-learning system for students to use in combination with the textbooks. Furthermore, theoretical principles are explained by means of specific examples and solution of specific problems using practical examples. The code for these software reuse examples and exercises is also made available in the e-learning system.

*One hour out of the three is normally devoted to laboratory-based programming exercises.

	Lectures are supplemented with supervised computer laboratories, which include demonstrations of taught concepts and experimentation with related technologies to solve specific problems via exercises. Hence, during laboratory sessions, students apply their gained knowledge and identify the principles taught in the lecture sessions by means of working on different tasks and solving domain-specific problems. The course includes a midterm test that involves both theoretical and critical thinking questions, as well as practical software reuse and programming exercises. The midterm test is undertaken using the e-learning system. Also, a course project is assigned to the students since this is a practical-oriented course. Finally, the course assessment is completed by means of a three-hours final exam at the end of the semester.
Bibliography	**Textbooks:** 1. Michel Ezran, Maurizio Morisio, Colin Tully, "Practical Software Reuse" (Practitioner Series), Paperback: 216 pages, Publisher: Springer; 1st edition (April 2, 2002), Language: English, ISBN-10: 1852335025, ISBN-13: 978-1852335021. 2. John Vlissides, Ralph Johnson, Richard Helm, Erich Gamma, "Design Patterns: Elements of Reusable Object-Oriented Software", Publisher: Addison-Wesley Professional, Release Date: October 1994, ISBN: 0201633612. **References:** 1. "Why Software Reuse has Failed and How to Make It Work for You", Douglas C. Schmidt, Available Online: Link. 2. "Design patterns, the big picture, Part 1: Design pattern history and classification", Jeff Friesen, JavaWorld | Nov 21, 2012, Available Online: Link. 3. "Design patterns, the big picture, Part 2: Gang-of-four classics revisited", Jeff Friesen, JavaWorld | Dec 26, 2012, Available Online: Link. 4. "Design patterns, the big picture, Part 3: Beyond software design patterns", Jeff Friesen, JavaWorld | Nov 21, 2012, Available Online: Link. 5. "Mobile UI Design Patterns – A Deeper Look At The Hottest Apps Today", Dominik Pacholczyk, UXPin, Available Online: Link.
Assessment	• Midterm Test: 20% • Course Project: 30% • Final Exam: 50%
Language	English.

*One hour out of the three is normally devoted to laboratory-based programming exercises.