Maestría en Ciencias de la Información y las Comunicaciones

Emphasis

NAME OF THE SUBJECT: Software Engineering

- Mandatory (X): Basic ( ) Complementary ( )
- Elective ( ): Intrinsic ( ) Extrinsic ( )

NUMBER OF ACADEMIC CREDITS: Four (4).

COURSE TYPE: THEORETICAL: ___ PRACTICAL: ___ THEORETICAL-PRACTICAL: _X_
Methodological alternatives:
Master Class (X), Seminar (X), Seminar - Workshop (X), Workshop ( ), Practice (X), Tutored projects (X), Other: _____

Justification

The Software Engineer normally faces the modeling and construction of logical machines to solve problems of process automation or improvement in the processes of organizations from the point of view of information processing. This subject is essential to consolidate in the future Master of the Universidad Distrital the methodological discipline to develop high-quality software.

This academic space places greater emphasis on the knowledge area of software design applied to direct engineering for the development of software-based solutions. Solid training in computer science and computer programming is a necessary requirement to take this subject.

Since this subject requires training in the student methodological discipline and exercising the use of a metamodel to generate software models, the training is oriented to the theoretical-practical exercise based on a classroom project, which the student faces the stage development of functional models based on use cases structural and dynamic of a software-based application. Throughout the semester the understanding of the fundamental concepts of the metamodel will be deepened, in this case, UML (Unified Modeling Language), with the simultaneous application of the concepts to the real problem of analysis and software design of the course project.

In addition, the review of technological issues that complement knowledge of the state of the art of technologies that support software development will be addressed. The exercise of comprehensive reading will be part of the development of the course. Technical readings related to the areas of knowledge of programming language semantics, principles and

attributes of software quality and models of quality assessment of software-based products will be addressed.

Content

General description xxxx.

GENERAL OBJECTIVE
Review with students the basic theoretical elements of management, analysis, design, implementation and testing of software development projects with a practical case study exercise that reinforces the acquisition of knowledge. Likewise, confront the student with

technical literature that places him in the state of the art and exercises in him the discipline of technical updating through continuous reading.

SPECIFIC OBJECTIVES

- Contextualize the student in what he understands the discipline of Software Engineering and understand the semantic levels of specification of programming languages
- Provide conceptual elements that allow controlling the development of a software development project. Review and practically apply some of the most popular software metrics.
- Apply the Unified Modeling Language and a simplified Unified Process Model to a real software development project during the semester.
- Generally review the quality attributes of the software and introduce quality-driven design into the design.

SYNTHETIC PROGRAM:

Unit 1. Introduction
1.1. Software Engineering Concepts
1.2 Knowledge areas of Software Engineering.
1.3. Abstraction levels in engineering. Conceptual differentiation of model, metamodel and meta-metamodel.

Unit 2. Process, Project and Product Metrics
2.1. Analysis of the key components in the development of a project: The process, the human resource and the project.
2.2 Review of the most common product, project and process metrics.
2.3 Estimation vs. Measurement: Function point estimation technique.

Unit 3. Analysis and Object Oriented Design with UML
3.1. Historical context of the evolution of UML.
3.2. Scope of the UML superstructure recommendation.
3.3. Compliance levels of modeling tools with respect to UML.
3.4 UML semantic areas.
3.5. Functional modeling based on use cases: Diagrams and specification approaches. 3.6. Static or structural modeling: Class diagrams, associations and structural collaboration, component diagrams, deployment diagrams.
3.7. Mapping or translation of domain models or business logic to persistence models.
3.8. Dynamic modeling

Unit 4. Software Quality Attributes

4.1. Review of the relevant quality attributes in software development: Performance, Ease of testing, Maintainability, Ease of understanding, Security, Availability, Reliability, Usability, Modifiability, Ease of reuse, Conceptual integrity, Latency, Concurrency.
4.2. Brief review of design driven by quality attributes.

Strategies

METHODOLOGY:

Master classes and conferences will be held on the topics scheduled in the course. Students will reinforce their understanding with technical readings provided by the teacher.

A case study will be used to which the modeling concepts will be applied and which will be gradually assessed by the teacher as part of the evaluation with a final final report and basic programming product.

The case study will have three revision milestones: The first revision will focus on the deliverable of functional modeling, the second on the deliverable of structural modeling and the third on the dynamic model verifiable with the obtained application prototype. The deliverables will be received in the form of specification documents in PDF format accompanied by the engineering models generated in the modeling tool, they are reviewed by the teacher in the electronic document format and then personal advice is provided to clarify doubts and guide methodological adjustments to the delivered models. During the course, a scope prototype negotiation based on use cases will be proposed.

Periodic evaluations on specific topics and readings will be carried out.

	Hours			Teacher hours / week	Student hours / week	Total Hours Student / semester	Academic credits
Type of course	DW	CW	AW	(DW + CW)	(DW + CW +AW)	X 16 weeks	
Theoretical-Practical	4	2	6	6	12	192	4

Direct Presential Work (DW): classroom work in plenary session with all students.
Mediated-Cooperative Work (CW): Teacher tutoring work to small groups or individually to students.
Autonomous Work (AW): Student work without the presence of the teacher, which can be done in different instances: in work groups or individually, at home or in a library, laboratory, etc.)

Resources

RESOURCES :

- Conference room
- Videobeam and personal computer.
- Software modeling tool using UML 2
- Software development tools

BIBLIOGRAPHY:

- Pfleeger, Shari Lawrence. “Ingeniería de software. Teoría y práctica”. Prentice Hall. Primera Edición. 2002
- Pressman, Roger S. “Ingeniería del Software. Un Enfoque Práctico”. Séptima Edición. Mc. Graw Hill. 2010
- Sommerville, Ian. ”Software Engineering”. Séptima Edición. Addison Wesley. 2004 LEN BASS P. C., AND KAZMAN, R. “Software Architecture in Practice”. Addison-Wesley, 1998.
- Taylor, Richard N.; Medvidovic, Nenad; Dashofy, Eric M. “Software Architecture. Foundations, Theory and Practice”. John Wiley & Sons Inc. 2010.
- Weitzenfeld, Alfredo. “Ingeniería de Software Orientada a Objetos con UML, JAVA e Internet”. Thomson. 2005
- Sommerville, Ian. “Software Process Models”. Computer Science Handbook. Second Edition. Editor-in – Chief Allen B. Tucker. Chapman&Hall/CRC in cooperation with ACM. 2004.
- Ghezzi, Carlo; Mehdi, Jazaheri and Mandrioli, Dino. “Software Qualities and Principles”. Computer Science Handbook. Second Edition. Editor-in –Chief Allen B. Tucker. Chapman&Hall/CRC in cooperation with ACM. 2004.
- Kapfhammer, Gregory. “Software Testing”. Computer Science Handbook. Second Edition. Editor-in – Chief Allen B. Tucker. Chapman&Hall/CRC in cooperation with ACM. 2004.
- Martin, Robert C. “Object-Oriented C++ Applications Using The Booch Method”. Prentice Hall.1.995.
- Meyer, Bertrand. “Object Oriented Software Construction”. Prentice Hall, Englewood Cliffs, N.J. 1.988.
- Coad, Peter y Yourdon, Edward. “Object Oriented Analysis”. Prentice Hall Cliffs, N.J. 1.990
- Coad, Peter y Yourdon, Edward. “Object Oriented Modeling and Design”. Prentice Hall Cliffs, N.J. 1.991
- Tanenbaum, Andrew S. “Sistemas Operativos Modernos”.Prentice Hall.1.992
- Newman, Alexander et al. “Using JAVA”. Que Corporation. 1996.
- W. , Richard Stevens. “UNIX Network Programming”. Prentice Hall. 1990
- Collins, Dave. “Designing Object Oriented Interfaces”. Benjamin /Cummings Publishing Company.Inc. 1.995.
- Rumbaugh, James; Blaha, Michael; Premerlani, William; Frederick, Eddy y Lorensen, William. “Modelado y Diseño Orientado a Objetos. Metodología OMT”. Prentice Hall. 1996
- Booch, Grady. “Análisis y Diseño Orientado a Objetos”. Addison-Wesley/Diaz de Santos. 1996
- Baker, Seán.”CORBA Distributed Objects”. Addison-Wesley.1997 Booch, Grady; Rumbaugh, James; Jacobson, Ivar; “El Lenguaje Unificado de Modelado”. Addison Wesley y ACM Press Books. 1999
- Booch, Grady; Rumbaugh, James; Jacobson, Ivar; “El Proceso Unificado de Desarrollo de Software”. Addison Wesley y ACM Press Books. 1999

  Larman, Craig;”UML y Patrones. Introducción al Análisis y Diseño Orientado a Objetos”. Prentice Hall. 1999 Fowler, Kendall; Scott, Kendall; “UML Gota a Gota”. Addison Wesley Longman., Pearson y Prentice Hall. 1999

Course Schedule

Week /Unid	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1. Unit 1	X	X														
2. Unit 2			X	X	X	X										
3. Unit 3							X	X	X	X	X	X				
4. Unit 4													X	X	X	X

Evaluation

	EVALUATION	DATE	PERCENTAGE
FIRST **GRADE**	Readings and presentations	All semester	20%
SEGUNDA **NOTA**	Exercises	All semester	20%
TERCERA **NOTA**	Project	Week 14	30%
CUARTA **NOTA**	Exam	Week 16	30%