San Pablo Catholic University (UCSP)
Undergraduate Program in Computer Science
SILABO

CS391. Software Engineering III (Mandatory)

2020-I

1. General information		
1.1 School	:	Ciencia de la Computación
1.2 Course	:	CS391. Software Engineering III
1.3 Semester	:	7^{mo} Semestre.
1.4 Prerrequisites	:	CS292. Software Engineering II. (6^{th} Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Virtual
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation

Software development requires the use of best development practices, IT project management, equipment management And efficient and rational use of quality assurance frameworks, these elements are key and transversal during the whole productive process. The construction of software contemplates the implementation and use of processes, methods, models and tools that allow to achieve the realization of the quality attributes of a product.

4. Summary

1. Software Evolution 2. Software Project Management 3. Software Project Management 4. Software Processes 5. Estándares ISO/IEC

5. Generales Goals

- Understand and implement the fundamental concepts of project management and software equipment management.
- Understand the fundamentals of project management, including its definition, scope, and need for project management in the modern organization.
- Students have to understand the fundamental concepts of CMMI, PSP, TSP to be adopted in software projects.
- Describe and understand quality assurance models as a key framework for the success of IT projects.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: Software Evolution (12)	
Competences:	
Content	**Generales Goals**
• Software development in the context of large, preexisting code bases – Software change – Concerns and concernlocation – Refactoring • Software evolution • Characteristics of maintainable software • Reengineering systems • Software reuse – Code segments – Libraries and frameworks – Components – Product lines	• Identify the principal issues associated with software evolution and explain their impact on the software lifecycle [Familiarity] • Estimate the impact of a change request to an existing product of medium size [Usage] • Use refactoring in the process of modifying a software component [Usage] • Discuss the challenges of evolving systems in a changing environment [Familiarity] • Outline the process of regression testing and its role in release management [Familiarity] • Discuss the advantages and disadvantages of different types of software reuse [Familiarity]
Readings: Pressman and Maxim (2015), Sommerville (2017)	

UNIT 2: Software Project Management (10)	
Competences:	
Content	**Generales Goals**
• Team participation – Team processes including responsabilities for task, meeting structure, and work schedule – Roles and responsabilities in a software team – Team conflict resolution – Risks associated with virtual teams (communication, perception, structure) • Effort estimation (at the personal level) • Risk – The role of risk in the lifecycle – Risk categories including security, safety, market, financial, technology, people, quality, structure and process • Team management – Team organization and decision-making – Role identification and assigment – Individual and team performance assessment • Project management – Scheduling and tracking – Project management tools – Cost/benefit analysis	• Discuss common behaviors that contribute to the effective functioning of a team [Familiarity] • Create and follow an agenda for a team meeting [Usage] • Identify and justify necessary roles in a software development team [Usage] • Understand the sources, hazards, and potential benefits of team conflict [Usage] • Apply a conflict resolution strategy in a team setting [Usage] • Use an ad hoc method to estimate software development effort (eg, time) and compare to actual effort required [Usage] • List several examples of software risks [Familiarity] • Describe the impact of risk in a software development lifecycle [Familiarity] • Describe different categories of risk in software systems [Familiarity] • Demonstrate through involvement in a team project the central elements of team building and team management [Usage] • Describe how the choice of process model affects team organizational structures and decision-making processes [Familiarity] • Create a team by identifying appropriate roles and assigning roles to team members [Usage] • Assess and provide feedback to teams and individuals on their performance in a team setting [Usage] • Using a particular software process, describe the aspects of a project that need to be planned and monitored, (eg, estimates of size and effort, a schedule, resource allocation, configuration control, change management, and project risk identification and management) [Familiarity]
Readings: Pressman and Maxim (2015), Sommerville (2017)	

UNIT 3: Software Project Management (8)	
Competences:	
Content	**Generales Goals**
• Software measurement and estimation techniques • Software quality assurance and the role of measurements • Risk – Risk identification and management – Risk analysis and evaluation – Risk tolerance (e.g., risk-adverse, risk-neutral, risk-seeking) – Risk planning • System-wide approach to risk including hazards associated with tools	• Track the progress of some stage in a project using appropriate project metrics [Usage] • Compare simple software size and cost estimation techniques [Usage] • Use a project management tool to assist in the assignment and tracking of tasks in a software development project [Usage] • Describe the impact of risk tolerance on the software development process [Assessment] • Identify risks and describe approaches to managing risk (avoidance, acceptance, transference, mitigation), and characterize the strengths and shortcomings of each [Familiarity] • Explain how risk affects decisions in the software development process [Usage] • Identify security risks for a software system [Usage] • Demonstrate a systematic approach to the task of identifying hazards and risks in a particular situation [Usage] • Apply the basic principles of risk management in a variety of simple scenarios including a security situation [Usage] • Conduct a cost/benefit analysis for a risk mitigation approach [Usage] • Identify and analyze some of the risks for an entire system that arise from aspects other than the software [Usage]
Readings: Pressman and Maxim (2015), Sommerville (2017)	

UNIT 4: Software Processes (12)

Competences:

Content	**Generales Goals**
• System level considerations, i.e., the iteraction of software with its intended enviroment • Introduction to software process models (e.g., waterfall, incremental, agile) – Activities with software lifecycles • Programming in the large vs. individual programming • Evaluation of software process models • Software quality concepts • Process improvement • Software process capability maturity models • Software process measurements	• Describe how software can interact with and participate in various systems including information management, embedded, process control, and communications systems [Usage] • Describe the relative advantages and disadvantages among several major process models (eg, waterfall, iterative, and agile) [Usage] • Describe the different practices that are key components of various process models [Usage] • Differentiate among the phases of software development [Usage] • Describe how programming in the large differs from individual efforts with respect to understanding a large code base, code reading, understanding builds, and understanding context of changes [Usage] • Explain the concept of a software lifecycle and provide an example, illustrating its phases including the deliverables that are produced [Usage] • Compare several common process models with respect to their value for development of particular classes of software systems taking into account issues such as requirement stability, size, and non-functional characteristics [Usage] • Define software quality and describe the role of quality assurance activities in the software process [Usage] • Describe the intent and fundamental similarities among process improvement approaches [Usage] • Compare several process improvement models such as CMM, CMMI, CQI, Plan-Do-Check-Act, or ISO9000 [Usage] • Assess a development effort and recommend potential changes by participating in process improvement (using a model such as PSP) or engaging in a project retrospective [Usage] • Explain the role of process maturity models in process improvement [Usage] • Describe several process metrics for assessing and controlling a project [Usage] • Use project metrics to describe the current state of a project [Usage]

Readings: Pressman and Maxim (2015), Sommerville (2017)

UNIT 5: Estándares ISO/IEC (6)	
Competences:	
Content	**Generales Goals**
• ISO 9001:2001. • ISO 9000-3. • ISO/IEC 9126. • ISO/IEC 12207. • ISO/IEC 15939. • ISO/IEC 14598. • ISO/IEC 15504-SPICE. • IT Mark. • SCRUM. • SQuaRE. • CISQ.	• Learn and apply correctly standards and international standards . [Usage]
Readings: Sommerville (2017), Pressman and Maxim (2015)	

8. Methodology

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

Pressman, Roger S. and Bruce Maxim (Jan. 2015). *Software Engineering: A Practitioner's Approach.* 8th. McGraw-Hill.

Sommerville, Ian (Mar. 2017). *Software Engineering.* 10th. Pearson.