684/18

Aprobado por resolución N°113 del CFI de fecha 04.07.2017

FACULTAD DE INGENIERIA

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

Programa de Taller de Programación

1. NOMBRE DE LA UNIDAD CURRICULAR

Taller de Programación

2. CRÉDITOS

15 créditos

3. OBJETIVOS DE LA UNIDAD CURRICULAR

El objetivo de la unidad curricular es integrar conocimientos adquiridos en unidades previas de programación e incorporar conocimientos avanzados para la construcción de sistemas de software de mediano y gran porte, aplicando conceptos de la orientación a objetos.

Particularmente, la unidad curricular se enfocará en:

- Profundizar en el uso de herramientas conceptuales para el análisis y diseño de sistemas orientados a objetos.
- Aplicar una metodología básica para el uso de dichas herramientas.
- Poner en evidencia problemas que surgen en la construcción de sistemas de software y plantear herramientas para su solución.
- Introducir un lenguaje de programación orientado a objetos adecuado al objetivo de la unidad.

4. METODOLOGÍA DE ENSEÑANZA

La enseñanza estará realizada fundamentalmente en modalidad de taller, o sea: centrada en laboratorios asistidos por un docente. Se dictarán 2 horas semanales de clase, incluyendo clases de monitoreo por parte del docente y presentaciones teóricas periódicas. Además, cada alumno deberá dedicar un promedio de 13 horas semanales para estudio y realización de trabajos de laboratorio.

5. TEMARIO

La unidad curricular se basa fuertemente en un trabajo de laboratorio grupal orientado al desarrollo de un sistema de software. El desarrollo apuntará a utilizar de manera integral los conocimientos obtenidos en unidades previas así como a abordar aspectos habituales del desarrollo, dentro de los que se encuentran:

1. Análisis y diseño de software con UML
2. Diseño de software
    a. Patrones de Diseño
    b. Atributos de Calidad (distribución, seguridad, etc.)
3. Tecnologías
    a. Implementación de sistemas
    b. Desarrollo de interfaces de usuario de escritorio y web
    c. Comunicaciones remotas (ej. web services)
    d. Persistencia de datos
4. Verificación de software

6. BIBLIOGRAFÍA

Tema	Básica	Complementaria
Análisis y diseño de software con UML	(1)(2)	(7)(8)
Diseño de software	(2)(3)	
Tecnologías	(4)(5)	(9)(10)
Verificación de software	(6)	(7)

6.1 Básica

1. Larman, Craig (2001). Applying UML and Patterns (2a Ed). Prentice Hall. ISBN: 9780130925695.
2. Fowler, Martin (2003). UML Distilled (3a Ed). Addison Wesley. ISBN: 9780321193681.
3. Gamma, Erich et al (1995). Design Patterns. Addison-Wesley. ISBN: 0201633612.
4. Eckell, Bruce (2002). Thinking in Java (3a Ed). Prentice Hall. ISBN: 0131002872.
5. Sitio de Java. http://www.oracle.com/technetwork/java
6. Myers, Glenford; Sandler, Corey; Badgett, Tom (2011). The Art of Software Testing (3a Ed). Wiley John and Sons. ISBN: 1118031962.

6.2 Complementaria

7. Ghezzi, Carlo et al (2002). Fundamentals of Software Engineering (2a Ed). Prentice Hall. ISBN: 0138204322.
8. Object Management Group (2001). UML Specification. http://www.omg.org/uml
9. Flanagan, David (2014). Java in a Nutshell (6a Ed). O'Reilly. ISBN: 0596002831.
10. Weiss, Mark (2011). Data Structures & Algorithm Analysis in Java. Peachpit Press. ISBN: 0201357542.

7. CONOCIMIENTOS PREVIOS EXIGIDOS Y RECOMENDADOS

7.1 Conocimientos Previos Exigidos: Conocimientos básicos de programación estructurada, estructuras de datos y algoritmos, y desarrollo orientado a objetos.

7.2 Conocimientos Previos Recomendados: Ninguno

ANEXO A
Para todas las Carreras

Esta primera parte del anexo incluye aspectos complementarios que son generales de la unidad curricular.

A1) INSTITUTO

Instituto de Computación

A2) CRONOGRAMA TENTATIVO

El trabajo de laboratorio estará dividido en tareas de 3-4 semanas de duración. Cada tarea consistirá en una iteración completa (análisis, diseño, implementación y verificación) de desarrollo de un conjunto definido de funcionalidades. Todas las tareas estarán relacionadas entre sí, siendo cada tarea una iteración del desarrollo de un mismo sistema de software.

Al finalizar cada tarea se realizará una defensa grupal de la misma en sala de máquinas, y una evaluación individual de la tarea que podrá ser escrita, oral, etc. A continuación se presenta un ejemplo de cronograma para tres tareas de cuatro semanas cada una.

Semana 1	Tecnologías (2 hs. de clase) Análisis y diseño de software con UML y Laboratorio (1 hs.)
Semana 2	Tecnologías (2 hs. de clase) Análisis y diseño de software con UML y Laboratorio (1 hs.)
Semana 3	Tecnologías (2 hs. de clase) Diseño de software y Laboratorio (1 hs.)
Semana 4	Verificación de software (2 hs. de clase) Laboratorio (1 hs.)
Semana 5	Defensas grupales (1 hs. de clase) Evaluación individual
Semana 6	Tecnologías (2 hs. de clase) Análisis y diseño de software con UML y Laboratorio (1 hs.)
Semana 7	Tecnologías (2 hs. de clase) Diseño de software y Laboratorio (1 hs.)
Semana 8	Tecnologías (2 hs. de clase) Laboratorio (1 hs.)
Semana 9	Tecnologías (2 hs. de clase) Laboratorio (1 hs.)
Semana 10	Defensas grupales (1 hs. de clase) Evaluación individual
Semana 11	Tecnologías (2 hs. de clase) Análisis y diseño de software con UML y Laboratorio (1 hs.)
Semana 12	Tecnologías (2 hs. de clase) Diseño de software y Laboratorio (1 hs.)
Semana 13	Tecnologías (2 hs. de clase) Laboratorio (1 hs.)
Semana 14	Tecnologías (2 hs. de clase) Laboratorio (1 hs.)
Semana 15	Defensas grupales (1 hs. de clase) Evaluación individual

Aprobado por resolución N°113 del CFI de fecha 04.07.2017

A3) MODALIDAD DEL CURSO Y PROCEDIMIENTO DE EVALUACIÓN

La evaluación del curso se basa en la realización del laboratorio grupal. Cada tarea constará de entregas obligatorias de ejercicios así como defensas en máquina. Las entregas y defensas realizadas durante el curso tendrán un puntaje asignado y un nivel de suficiencia definido. El puntaje final será eliminatorio para aquellos estudiantes que no obtengan el nivel de suficiencia. De la misma manera, la insuficiencia global de una de las tareas será suficiente para la insuficiencia del curso.
La evaluación será sobre un máximo de 100 puntos. Del resultado obtenido en la evaluación surgirán dos posibilidades:

- Exoneración del curso: el estudiante aprueba totalmente el curso (se logra acumulando como mínimo 60 puntos)
- Insuficiencia en el curso: el estudiante reprueba la unidad, debiendo inscribirse nuevamente en el curso (en caso de no llegar a 60 puntos)

Si bien el laboratorio es grupal, la evaluación podrá realizarse individualmente en caso de ser necesario. En este caso, se podrá determinar la insuficiencia del trabajo de un estudiante en base a los resultados de las evaluaciones grupales, al seguimiento del grupo que hace el docente de monitoreo a lo largo del curso, evaluación individual (oral u escrita).

A4) CALIDAD DE LIBRE

Esta unidad curricular no adhiere a la calidad de libre.

A5) CUPOS DE LA UNIDAD CURRICULAR

No tiene cupo

ANEXO B para la(s) carrera(s) Ingeniería en Computación (plan 97)

B1) ÁREA DE FORMACIÓN

Actividades integradoras: Talleres, Pasantías y Proyectos

B2) UNIDADES CURRICULARES PREVIAS

Para el Curso: los estudiantes deben cumplir alguna de las siguientes condiciones:

1. Curso de Programación 4 y Examen de Programación 3
2. Examen de Programación 4

Para el Examen: No aplica

Aprobado por resolución N°113 del CFI de fecha 04.07.2017

ANEXO B para la(s) carrera(s) Ingeniería en Computación (plan 87)

B1) ÁREA DE FORMACIÓN

No corresponde

B2) UNIDADES CURRICULARES PREVIAS

Para el Curso: Curso de Programación 4.

Para el Examen: No aplica

APROB. RES. CONSEJO DE FAC. ING.
21/6/18 06126-002277-17
061100-000970-10