TEC | Tecnológico de Costa Rica

Programa del curso IC-6821

Diseño de Software

Escuela de Computación
Carrera de Ingeniería en Computación, Plan 410

TEC | Tecnológico de Costa Rica

I parte: Aspectos relativos al plan de estudios

1 Datos generales

Nombre del curso:	Diseño de Software
Código:	IC-6821
Tipo de curso:	Teórico – práctico
Electivo o no:	No
Nº de créditos:	4
Nº horas de clase por semana:	4
Nº horas extraclase por semana:	8
Ubicación en el plan de estudios:	Curso del quinto semestre de la carrera de Ingeniería en Computación
Requisitos:	IC5821 Requerimientos de Software
Correquisitos:	No los hay
El curso es requisito de:	IC6831 Aseguramiento de la Calidad del Software
Asistencia:	Obligatoria
Suficiencia:	No
Posibilidad de reconocimiento:	Si
Vigencia del programa:	II Semestre, 2012

2 Descripción general

En este curso se estudia el diseño de software, dentro del contexto del proceso de desarrollo de software. Se espera que al finalizar el curso el estudiante sea capaz de diseñar software mediante un proceso de diseño sistemático.

3 Objetivos

General

Valorar un conjunto de métodos, técnicas y herramientas para el diseño y la especificación de un producto de software.

Específicos

Al finalizar el curso el estudiante estará en capacidad de:

- Aplicar técnicas y herramientas orientadas a objetos para la modelación del diseño de software.
- Documentar la toma de decisiones durante la etapa de diseño del software.
- Comprender los diferentes niveles de abstracción en que deben expresarse las soluciones de problemas de diseño.
- Desarrollar destrezas para diseñar la arquitectura de software
- Analizar aspectos de las tecnologías actuales y de las tendencias tecnológicas que influyen en los diseños del software.

4 Contenidos

1) Introducción/vistazo al diseño y arquitectura de software. (0.5 semana)

a) De los requerimientos al diseño.

b) Pasos para diseñar software

c) Artefactos del diseño de software

d) El rol del arquitecto de software

2) El diseño de interfaces de usuario (3.5 semanas)

a) El concepto de utilizabilidad ("usability")

b) Técnicas para diseñar interfaces de usuario (diseño en papel, reutilización de conceptos y diseños conocidos, diseño con usuarios, etc.)

c) Diseño de la parte funcional de la interfaz, considerando diferentes tecnologías, eficiencia y eficacia

d) La importancia de la validación de datos

e) Tipos de componentes para el diseño de interfaces de usuario

f) Software para el diseño de interfaces de usuario

g) Guías de diseño gráfico de la interfaz, considerando diferentes tecnologías

h) Consideraciones para el diseño de reportes

3) Principios de diseño (1 semana)

a) Principio de divide y conquista

b) Principio de incrementar la cohesión

c) Principio de reducción de acoplamiento

d) Principio de mantenimiento del nivel de abstracción alto

e) Principio de incrementar la reusabilidad y reusar lo existente

f) Principio del diseño para la flexibilidad

g) Principio de anticipar la obsolecencia

h) Principio de diseñar para la portabilidad

i) Principio del Diseño para pruebas ("testabilidad")

j) Principio del diseño defensivo

4) El diseño de la arquitectura del software (5 semanas)

a) El entendimiento del problema

b) Identificación de elementos y sus relaciones

c) Descomposición del sistema

d) Diseño con componentes

e) Diseño con servicios

f) Diseño de la integración entre sistemas

g) El papel de los atributos de calidad (requerimientos no funcionales) en la especificación de la arquitectura

h) Especificación de la arquitectura de software (artefactos)

i) Estilos y patrones arquitectónicos

j) Tendencias tecnológicas en arquitectura de software

k) Arquitectura de aplicaciones empresariales

5) El diseño detallado del software (4 semanas)

a) Artefactos del diseño detallado

b) Diseño de los casos de uso

c) Diseño de los componentes y de los servicios

d) Diseño de clases

e) Aplicación de patrones de diseño en el diseño de clases

6) El proceso de revisión del diseño (1 semanas)

a) En relación con los requerimientos funcionales

b) En relación con los atributos de calidad del software

(requerimientos no funcionales)

c) En relación con los requerimientos tecnológicos.

d) Revisión y Control del diseño durante el proceso de construcción de software.

7) Tendencias en el diseño de software (0.5 semana).

II parte: Aspectos operativos

5 Metodología de enseñanza y aprendizaje

Exposición magistral de los temas y análisis de casos de estudio, ejercicios y proyectos prácticos para afianzar los conocimientos, desarrollar habilidades y destrezas del trabajo en equipo.

6 Evaluación

El curso será evaluado con tres rubros principales: proyectos programados, exámenes parciales y un último rubro que comprende pruebas y tareas cortas.

Proyecto programado (40%): Se asignará un proyecto programado que deberá ser trabajado durante todo el semestre. El proyecto se trabajará grupalmente. La entrega está programada en la semana 19. Adicionalmente se deben hacer presentaciones parciales de avance en las semanas 7 y 14.

Exámenes parciales (40%): Se aplicarán dos exámenes parciales, cada uno correspondiente a 20% del rubro. Los exámenes están programados para en las semanas 9 y 19.

Pruebas y tareas cortas (20%): Adicionalmente se asignarán tareas y cortas y se aplicarán pruebas cortas durante el semestre, éstas serán anunciadas con una semana de antelación.

Proyectos programados (1)	40%
Exámenes parciales (2)	40%
Pruebas y tareas cortas	20%

No es posible eximirse de ninguna evaluación del curso.

7 Bibliografía

Obligatoria

Ashmore, D. 2004. The J2EE Architect's Handbook. DVT Press.

Complementaria

Primer: Agile Model-driven development with UML 2.0. Cambridge University Press.

Liskov, Barbara. 2001. Program Development in Java: Abstraction, Specification, and Object-oriented Design. Addison-Wesley.

Braude, Eric J. 2004. Software Design: from programming to architecture. John Wiley & Sons.

Lethbridge, Timothy C., Laganiere, Robert. 2nd Ed. 2004. Object-Oriented Software Engineering: Practical Software Development using UML and Java. McGrawHill.

Albin, Stephen. 2003. The Art of Software Architecture: Design Methods and Techniques. John Wiley & Sons.

Booch. Análisis y Diseño Orientado a Objetos Con Aplicaciones. Addison-Wesley/.

Budgen, David. Software Design, 2nd. Ed. Addison-Wesley, 2003

Fowler, M. UML Gota a Gota. Addison-Wesley, 1997.

IEEE Standard for Software Reviews, 1997.

Gorton, Ian. Essential Software Architecture. Springer-Verlag, 2006.

Jacobson; Booch; Rumbaugh. El Proceso Unificado de Desarrollo de

Software. Addison-Wesley, 2000.

Jacobson; Booch; Rumbaugh. El Lenguaje Unificado de Modelado. Addison-Wesley, 1999.

Nielsen. Usability Engineering. Morgan Kaufmann. 1993

Nielsen. Designing Web Usability. New Riders. 2000

Sommerville. Ingeniería de Software. Addison-Wesley, 2000

Fairley, Richard,Ingeniería de Software, McGraw-Hill/Interamericana de México, S.A. de C.V., 1988.

Pressman,Ingeniería del Software, McGraw- Hill / interamericana de México, S.A. de C.V

James Rumbaugh, Objetct-Oriented Modeling and Design, Prentice Hall, 1991

Larman, Craig. UML y Patrones. 2da edición. Prentice Hall. 2003.

www.microsoft.com

www.sun.com

8 Profesor

Diego Munguía Molina tiene estudios de Ingeniería en Computación (ITCR) y es egresado de la maestría en Ciencias Cognoscitivas (UCR), actualmente se encuentra desarrollando su tesis de graduación. Profesionalmente se ha desarrollado como arquitecto de software, acumulando experiencia en ingeniería de software, computación de alto rendimiento e integración de sistemas. Ha laborado como docente para la institución desde el 2012 en la Escuela de Computación impartiendo los cursos de Introducción a la Programación, Taller de Programación, Lenguajes de Programación, Compiladores e Intérpretes, Programación Orientada a Objetos y Diseño de Software.

Medio oficial electrónico: TEC-Digital (www.tec-digital.itcr.ac.cr)
Correos electrónicos: dmunguia@itcr.ac.cr
Oficina de Ingeniería en Computación, SIUA

Teléfono oficina: 24313987
Horario de consulta: K y J 1:00pm-6:00pm

Datos del asistente del curso:
Marco Álvarez Vega es estudiante avanzado de la carrera de Ingeniería en Computación en la SIUA.