UNIVERSIDAD DE BUENOS AIRES

Facultad de Ciencias Económicas

Departamento de Sistemas

Asignatura: Construcción de aplicaciones informáticas

Código: 654

Plan "1997"

Cátedra: Profesor Cesar A. BRIANO

Carrera: Lic. en Sistemas de Información

Aprobado por Res. Cons. Directivo
(F.C.E.)
Nro.: 1459/11

En caso de contradicción entre las normas previstas en la publicación y las dictadas con carácter general por la Universidad o por la Facultad, prevalecerán éstas últimas.

UNIVERSIDAD DE BUENOS AIRES
FACULTAD DE CIENCIAS ECONÓMICAS

MATERIA

654

CONSTRUCCIÓN DE APLICACIONES INFORMÁTICAS

PLAN 97
Departamento de Sistemas

Carrera de Licenciatura en Sistemas de Información de las Organizaciones

Profesor a Cargo: Lic. César A. Briano

APROBADO POR RESOLUCIÓN CONSEJO DIRECTIVO F.C.E. N° 1459/11

Buenos Aires
2010

En caso de contradicción entre las normas previstas en este programa y las dictadas con carácter general por la Universidad o por la Facultad, prevalecerán estas últimas.

CONSTRUCCIÓN DE APLICACIONES INFORMÁTICAS

A. ENCUADRE GENERAL.

A.1. FUNDAMENTACIÓN.

Esta asignatura está incluida en un grupo de materias que se ocupa de brindar conocimientos en el área de los sistemas y las tecnologías de la información.

Como dice R. Pressman en su libro Ingeniería del Software: "Actualmente el software ha superado al hardware como la clave del éxito de muchos sistemas basados en computadoras. Tanto si se utiliza la computadora para llevar un negocio, controlar un producto o capacitar un sistema, el software es el factor que marca la diferencia."

Durante los primeros años, el software se consideró como un apéndice del hardware. La programación carecía de métodos sistemáticos, lo que generaba muchos errores, atrasos y dificultades en su mantenimiento posterior. Estos graves problemas llevaron a comenzar una disciplina que integra métodos, herramientas y procedimientos para la planificación, análisis, diseño, codificación, prueba y mantenimiento del software.

Dentro de este concepto, el crecimiento en el uso de los sistemas computacionales hace imprescindible la aplicación de conceptos de ingeniería del software para la construcción de aplicaciones de calidad.

El conocimiento de estas técnicas resulta vital en la preparación de un profesional del campo de los sistemas de información, por cuanto estos conocimientos no están solamente relacionados con el desarrollo de la tecnología consecuente, sino con la dirección de proyectos de desarrollo exitosos, en el ámbito de la organización social donde le toque desenvolverse.

El desarrollo de las capacidades necesarias para la dirección de proyectos de desarrollo usando técnicas que aseguren la calidad del software, permitirá a las organizaciones donde estará inserto el profesional dotado de estos conocimientos, reducir de una manera interesante sus gastos en los procesos de desarrollo y lograr una importante ventaja competitiva: un buen software.

A.2. UBICACIÓN DE LA ASIGNATURA EN EL CURRICULUM DE LA CARRERA.

La asignatura debe estar ubicada en la currícula de la carrera en la parte media del "CICLO PROFESIONAL".

En esa etapa, los alumnos ya han adquirido bastante experiencia en el uso del computador como herramienta para el manejo de la información. Por otro lado, también han adquirido teórica y prácticamente, todos los fundamentos de los lenguajes de programación y en los conceptos vinculados con el desarrollo e implementación de sistemas.

Requiere una sólida formación lógica y de los conceptos básicos de lenguajes de programación y sistemas operativos.

Es por ello, que es correlativa de Teoría de los Lenguajes y Sistemas Operativos y Lógica.

A.3. OBJETIVOS DE APRENDIZAJE.

Lograr que los alumnos conozcan los principios básicos y los conceptos fundamentales de la ingeniería de software y desarrollen su aplicación práctica.

Conocer las distintas herramientas, métodos y procedimientos actuales, aplicables a la planificación, análisis, diseño, codificación, prueba y mantenimiento del software, con el objeto de poder conducir y asesorar profesionalmente proyectos de desarrollo de software.

Tomar conciencia de la importancia de establecer criterios de calidad de software.

A.4. PROGRAMA DE CONTENIDOS MÍNIMOS.

Paradigmas de construcción: ciclo de vida, prototipos, incremental y de 4ta. generación. Ingeniería de software. Planificación y control de proyectos de construcción. Fundamentos de diseño. Herramientas de desarrollo. Calidad de software. Estrategias y técnicas de prueba y depuración. Mantenimiento. Gestión de configuración. Análisis y diseño orientado a objetos.

B. ENFOQUE CONCEPTUAL.

B.1. PROGRAMA DE CONTENIDOS ANALÍTICOS.

UNIDAD TEMATICA I: INTRODUCCIÓN

Objetivos de la unidad: Introducir a los alumnos en las características particulares que tiene la construcción de aplicaciones informáticas y describir, de un modo general, aquellas tareas que deben realizarse para obtener software de calidad

1. La evolución y características del software. Ingeniería de software. Producto, Proceso y Proyecto. Conceptos.

2. Marco teórico para el desarrollo de aplicaciones de software.

3. Actividades protectoras de la calidad de software

UNIDAD TEMATICA II: MODELOS DE DESARROLLO DE SOFTWARE

Objetivos de la unidad: Que el alumno pueda analizar los modelos más utilizados en la construcción de software, sus diferencias, sus ventajas y desventajas y los ejemplos de aplicación.

1. Modelos de desarrollo: ciclo de vida, prototipos, DRA, Evolutivos, etc. Características. Ventajas y Desventajas. Casos de uso.

2. Modelos de desarrollo ágil, características.

UNIDAD TEMATICA III: GESTIÓN DE PROYECTOS

Objetivos de la unidad: Que el alumno comprenda las características, componentes y problemáticas de los proyectos informáticos y analice las herramientas que puede utilizar para una administración eficaz de los mismos.

1. Organización. Distintos Tipos. Ventajas y Desventajas.

2. Planificación, Calendarización y Seguimiento del Proyecto.

3. Métricas. Distintos Tipos. Ventajas y Desventajas. Métricas para la Calidad. Integración dentro del Proceso de Desarrollo.

4. Estimación de recursos del proyecto.

5. Análisis y Gestión de Riesgo.

UNIDAD TEMATICA IV: ANALISIS DE SOFTWARE

Objetivos de la unidad: Que el alumno entienda las características de la etapa de análisis, observando las diversas consideraciones que deben tenerse en cuenta, para comprender las necesidades de los usuarios y las características que debe tener el software a producir.

1. Ingeniería de Requisitos. Características de la etapa. Impacto sobre el resultado final.

2. Validación y Negociación de Requisitos. Especificación.

3. Modelado del Análisis. Distintos Tipos.

UNIDAD TEMATICA V: DISEÑO DE SOFTWARE

Objetivos de la unidad: Que el alumno aprenda sobre los fundamentos que deben considerarse a la hora de diseñar software.

1. Fundamentos del Diseño. Diseño Modular.

2. Diseño de datos. Diseño arquitectónico. Diseño procedimental. Diseño de Interfases.

3. Documentación del diseño.

UNIDAD TEMATICA VI: CODIFICACIÓN

Objetivos de la unidad: Que el alumno evalúe las características de los lenguajes y entornos de desarrollo, comprendiendo las dificultades de la etapa de programación. Que pueda elegir el lenguaje de programación adecuado a las características de la aplicación a construir.

1. Elección del lenguaje. Estándares de programación. El estilo de codificación. Documentación.

2. Entornos de desarrollo de software. Asistentes, Depuradores y Generadores de Código.

3. Programación orientada a Objetos.

4. Programación para web.

UNIDAD TEMATICA VII: TÉCNICAS DE VERIFICACIÓN, VALIDACIÓN Y MANTENIMIENTO

Objetivos de la unidad: Que el alumno conozca las tareas que deben desarrollarse sobre los programas ya construidos para asegurar su correcto funcionamiento y extender su vida útil.

1. Garantía de calidad del software.

2. Técnicas de prueba. Estrategias de prueba. Depuración.

3. Mantenimiento. Reingeniería de Software.

4. Gestión de la Configuración del software.

UNIDAD TEMATICA VIII: CALIDAD

Objetivos de la unidad: Que el alumno analice los conceptos de calidad vinculados con el desarrollo de programas informáticos y comprenda las actividades que deben desarrollarse para asegurar la calidad del producto final.

1. El concepto de calidad.

2. Revisiones de software.

3. El plan de aseguramiento de la calidad.

4. Control de calidad.

5. Marcos de trabajo, estándares y normas, de calidad y madurez, en el desarrollo de software

UNIDAD TEMATICA IX: PRÁCTICA

Objetivos de la unidad: Que el alumno se capacite en la construcción de aplicaciones mediante la ejercitación, práctica y ejecución de programas en entornos reales.

1. Aplicación práctica de las técnicas tratadas desarrollando programas utilizando lenguaje Visual Basic, pudiéndose utilizar otros lenguajes complementarios para el aprendizaje de temas específicos.

B.2. BIBLIOGRAFÍA.

B.2.1. BIBLIOGRAFÍA OBLIGATORIA.

1. **ROGER PRESSMAN**: Ingeniería del Software. 7ma Edición. 2010. Ed. McGraw-Hill.

2. **LUIS JOYANES AGUILAR**: Fundamentos de Programación. 3ra Edición. 2002. Ed. McGraw-Hill.

3. **CESAR A. BRIANO:** Modelos para el Desarrollo de Software, apunte de cátedra.

B.2.2. GUIAS PARTE PRACTICA.

1. **ERICA YONE y JUAN RECIA**: Apuntes teóricos sobre visual Basic. Compilados y editados por Leandro Arturi y Maximiliano A. Lopez

2. **MAXIMILIANO A. LOPEZ, LEANDRO MAURI y otros**: Guía de ejercicios prácticos.

B.2.3. BIBLIOGRAFÍA AMPLIATORIA.

1. **MICROSOFT,** Visual Basic 6, Manual del Programador. 2003. Ed. McGraw-Hill.

2. **IAN SOMMERVILLE**: Ingeniería de Software. 7ma Edición. 2005. Pearson Education.

3. **SHARI LAWRENCE PFLEEGER**: Ingeniería de Software. 3ra Edición. 2006. Prentice Hall.

4. **GLENFORD MYERS**: El arte de probar el software. 1979. Ed. El Ateneo.

C. METODOLOGÍA.

C.1. METODOLOGÍA DE CONDUCCIÓN DEL APRENDIZAJE.

C.1.1. DESARROLLO DE LA ASIGNATURA.

En el dictado de las clases teóricas, debe hacerse permanente referencia a su aplicación práctica, así como ejercitar abundantemente cada tema. La naturaleza práctica del contenido actual y futuro de la asignatura, hace imprescindible que se enfatice el entendimiento práctico de cada unidad temática.

En cuanto a los lenguajes a utilizar, debe analizarse constantemente la conveniencia de cada uno, coordinando con la materia Teoría de los Lenguajes y Sistemas Operativos el dictado de estos temas.

Debe darse prioridad a la formación lógica del alumno, teniendo en cuenta que a través del tiempo puede tener que utilizar otros lenguajes y técnicas pero la lógica será siempre lo más importante no sólo en los temas de la asignatura sino también en los demás de la carrera y en la actividad profesional.

C.1.2. DINÁMICA DEL DICTADO DE LAS CLASES.

La metodología adoptada para el dictado de las clases es la siguiente:

Las clases deberán ser dictadas en los días, horarios y aulas asignadas.

Es importante destacar que los alumnos deberán leer para cada clase la bibliografía indicada en el programa de clases y por el profesor a cargo del curso tanto para las clases teóricas como para las prácticas. De igual manera, se deberán entregar los ejercicios prácticos indicados en las clases prácticas.

Durante el dictado de la clase, los alumnos podrán solicitar a los docentes la aclaración de dudas surgidas a partir de sus explicaciones.

En el caso de las clases se prevé que algunos de los temas sean dictados directamente sobre computadoras, usando una notebook con cañón o bien dictando la clase en el Gabinete de Computación de la Facultad.

C.1.3. TRABAJOS PRÁCTICOS.

A criterio del docente, podrán efectuarse diferentes tipos de trabajos prácticos:

Trabajos Prácticos de Ejercitación.

Consistirán en la realización de problemas y ejercicios que se entregarán en las clases prácticas.

Trabajos Prácticos de Integración.

Consistirá en el desarrollo de un software pequeño, aplicando todas las herramientas, métodos y procedimientos desarrollados en las clases teóricas y prácticas. Con entregas parciales de avance, la presentación final será entregada en la semana de la última evaluación práctica.

Trabajos de Investigación.

Opcionalmente, cada panel de alumnos deberá realizar un trabajo de investigación a lo largo del cuatrimestre. Los temas serán asignados por el profesor en las primeras clases y el resultado del mismo será expuesto antes del segundo parcial teórico (ya que estos temas formarán parte del temario del mismo) y presentado en forma escrita. El profesor de cada curso podrá optar por reemplazar estos trabajos por la realización de "papers" sobre temas de actualidad, temas sin bibliografía suficiente, casos especiales, análisis comparativos, etc.

C.2. METODOLOGÍA DE EVALUACIÓN.

C.2.1. NORMAS DE EVALUACIÓN.

El criterio es que la evaluación del alumno es permanente.

Se tomarán 2 parciales teóricos y 2 prácticos. Las fechas de exámenes parciales serán anunciadas la primera semana de clases. Los exámenes parciales y sus recuperatorios pueden ser orales o escritos.

A criterio del docente, uno de los parciales prácticos puede ser reemplazado por un ejercicio integral (desarrollo de un software pequeño) realizado por paneles y a realizar a lo largo del cuatrimestre.

C.2.2. RÉGIMEN DE APROBACIÓN DE LA MATERIA.

Para la aprobación de la materia deberán tener **TODOS** los parciales aprobados, teniendo la posibilidad de recuperar sólo **UNO** teórico y **UNO** práctico, en la fecha indicada en el cronograma.

Además los alumnos deberán **aprobar los trabajos prácticos que se indiquen**, como condición para la aprobación de la materia.

Se colocará una nota adicional, en función del concepto que le merezca el alumno a la cátedra.

La nota final surgirá como ponderación de parciales, recuperatorios, concepto, trabajos prácticos y trabajos de investigación de acuerdo al criterio del profesor a cargo del curso.

Sólo corresponderá la calificación de ausente cuando el alumno no haya rendido ningún parcial o los que haya rendido estén aprobados y abandone la materia.

C.2.3. REQUISITOS PARA ALUMNOS LIBRES.

Los exámenes para los alumnos libres comprenderán dos exigencias.

- Realización de una prueba escrita práctica con problemas y ejercicios.
- Interrogatorio o evaluación escrita sobre aspectos teóricos de la materia.

La aprobación de cada uno de ellos, es condición para pasar a la exigencia siguiente.