1. DATOS DE LA ASIGNATURA

Nombre de la asignatura:	**Arquitectura de Software**
Carrera:	**Ingeniería en Sistemas Computacionales**
Clave de la asignatura:	**ISC-2104**
SATCA	**2-2-4**

2. HISTORIA DEL PROGRAMA

Lugar y fecha de elaboración o revisión	Participantes	Observaciones (cambios y justificación)
Instituto Tecnológico de Culiacán, 30 de abril de 2021.	Dra. María Lucía Barrón Estrada Dr. Ramón Zatarain Cabada Dr. Héctor Rodríguez Rangel M.C. Gloria Ekaterine Peralta Peñuñuri	Elaboración de asignatura: Arquitectura de Software.

3. PRESENTACIÓN

Caracterización de la asignatura

Esta asignatura aporta al perfil del Ingeniero en Sistemas Computacionales competencias profesionales para desarrollar arquitecturas de software que satisfagan requerimientos funcionales e incluyan atributos de calidad aplicando métodos y técnicas que permitan diseñar, implementar y evolucionar sistemas de software del mundo real.

La asignatura permitirá que el estudiante:

- Identifique tendencias actuales y futuras, así como el estado actual de la práctica en esta área.
- Ejecute el proceso de 3 fases (diseño, evaluación y transformación) para el desarrollo de arquitecturas de software para sistemas de cualquier dominio.
- Conforme equipo de trabajo con sus compañeros y asuma diferentes roles en el desarrollo de un proyecto.
- Evalúe el desempeño del equipo de acuerdo a las actividades y roles de cada uno de sus compañeros.
- Asuma y valore un compromiso ético en las diferentes actividades relativas al proceso de desarrollo de software.

Esta asignatura, es la aplicación práctica del conocimiento científico, a través de

los métodos y técnicas adecuados, para la creación y evolución de sistemas de software.

La materia de Arquitectura de Software se relaciona con la materia precedente de Ingeniería de Software.

Requiere de competencias previas como: Manejo de un lenguaje de modelado, dominio en el uso de herramientas CASE, uso de algún Sistema Manejador de Bases de Datos, dominio de algún lenguaje de programación orientado a objetos, aplicación de alguna metodología de desarrollo de sistemas y manejo de las diferentes plataformas operativas.

Intención didáctica.

La asignatura debe tener un enfoque teórico – práctico con el fin de fomentar en el estudiante la habilidad para aplicar la metodología para desarrollo de arquitecturas de software basada en los requerimientos funcionales y considerando los atributos de calidad necesarios en el sistema.

La asignatura se divide en cinco unidades.

En el bloque uno, Introducción a las Arquitecturas de software, se aborda la trascendencia del software a nivel global y la importancia de la arquitectura de software respecto a los atributos de calidad, la clasificación de los modelos arquitectónicos más utilizados de acuerdo al dominio del problema y las tendencias de arquitecturas para dominios específicos.

En el bloque dos, se analizan y clasifican los requerimientos asignando prioridades para que sean utilizados en el diseño de la arquitectura de software que cumpla con las especificaciones de calidad solicitadas.

En el bloque tres, se cubre el proceso de construcción de software incluyendo la vista de componentes de software, sus características y relaciones a un alto nivel de abstracción. Cubre los principios de diseño que gobiernan el propósito, la estructura y evolución de componentes de software. Se presenta la metodología de diseño de un modelo arquitectónico delimitando el contexto del sistema, e identificando abstracciones y componentes que permitan estructurar el modelo.

En el bloque cuatro se evalúa la arquitectura de software respecto a los atributos de calidad utilizando la evaluación basada en escenarios para identificar problemas y posibles soluciones de los mismos.

En el bloque cinco, se muestran y aplican las técnicas de transformación de

arquitecturas de software para mejorar el cumplimiento de los atributos de calidad.

4. COMPETENCIAS A DESARROLLAR:

Competencia general:

Diseñar arquitecturas de software que satisfacen requisitos de calidad aplicando la metodología para el diseño basado en la funcionalidad y generando arquitecturas de software que promueven la disminución del costo de desarrollo y mantenimiento de software para asegurar la entrega a tiempo de nuevos productos de software.

Competencias específicas:	
• Identifica conceptos fundamentales relacionados con la arquitectura de software. • Analiza requisitos funcionales y de calidad de sistemas de software. • Diseña una arquitectura de software usando la metodología para diseño basado en funcionalidad. • Evalúa la arquitectura de software usando el método de evaluación basado en atributos de calidad. • Genera una nueva arquitectura de software que cumpla con los requisitos funcionales aplicando los métodos de transformación de arquitecturas.	**Competencias genéricas:** **Competencias instrumentales** • Capacidad de análisis y síntesis. • Capacidad de organizar y planificar. • Comunicación oral y escrita en su propia lengua. • Solución de problemas. **Competencias interpersonales** • Trabajo en equipo. • Compromiso ético. **Competencias sistémicas** • Capacidad de aplicar los conocimientos en la práctica. • Capacidad de aprender. • Búsqueda del logro. • Capacidad de generar nuevas ideas (creatividad).

Katy Peralta 9/4/19 09:55
Deleted: .

5. COMPETENCIAS PREVIAS

Desarrollar soluciones de software, considerando los aspectos del modelo de negocios, mediante la aplicación de la metodología adecuada a la naturaleza del problema.

6. TEMARIO

Unidad	Temas	Subtemas
1	Introducción a las Arquitecturas de software	1.1 Importancia del Software en el ámbito mundial 1.1.1 Problemas generados por el software 1.1.2 Problemas por la falta de software 1.2 Definición de Arquitectura Software 1.3 Modelos de Arquitecturas de software. 1.4 Estructuras y vistas arquitectónicas 1.5 Importancia de la Arquitectura de Software
2	Requerimientos para el diseño de la Arquitectura de software	2.1 Clasificación de requerimientos 2.1.1 Requerimientos de software 2.1.1.1 Requerimientos funcionales 2.1.1.2 Requerimientos de calidad 2.1.2 Requerimientos de hardware 2.1.3 Otros requerimientos 2.2 Especificación de requisitos 2.2.1 Requerimientos en documentos 2.2.2 Requerimientos en entrevistas y stakeholders 2.2.3 Requerimientos en el negocio 2.3 Captura de requerimientos 2.4 Asignación de prioridades 2.5 Documentación de requerimientos

3	Diseño de la arquitectura basado en la funcionalidad (QASAR)	3.1 Definición de Actores 3.2 Definición del contexto del sistema 3.3 Identificación de arquetipos 3.4 Descomposición del sistema 3.5 Instancias del sistema 3.6 Documentación del diseño arquitectónico
4	Evaluación de la arquitectura de software	4.1 Tipos de evaluación 4.2 Métodos de evaluación de la arquitectura 4.2.1 Escenarios 4.2.2 Simulación y prototipos 4.2.3 Modelo matemático 4.2.4 Experiencia 4.3 Evaluación de la arquitectura basada en escenarios 4.3.1 Selección del atributo a evaluar 4.3.2 Desarrollo de expedientes de escenarios 4.3.3 Análisis de impacto 4.3.4 Documentación de los resultados de la evaluación
5	Transformación de arquitecturas de software	5.1 El proceso de transformación 5.2 Estilos de arquitectura 5.3 Patrones de arquitectura 5.4 Aplicación de patrones de diseño 5.5 Distribución de requerimientos

7. **SUGERENCIAS DIDÁCTICAS PARA EL DESARROLLO DE COMPETENCIAS ESPECÍFICAS**

Competencias específicas	**Actividades de aprendizaje**

Unidad I. Introducción a las Arquitecturas de software Identifica conceptos fundamentales relacionados con la arquitectura de software.	• Investigar y discutir en clase sobre algún tema de la unidad. Ejemplos: - Evolución del software - Fallas del software - Necesidades de software - Evolución de los métodos de diseño de software - Contrastar la problemática de los sistemas grandes contra los pequeños. - Describir algunos modelos de arquitecturas de software ampliamente utilizados
Unidad 2. Requerimientos para el diseño de la Arquitectura de software Analiza requisitos funcionales y de calidad de sistemas de software.	• Investigar en empresas regionales los métodos que utilizan para la recolección de requerimientos de software. • Analizar y clasificar requerimientos de software • Asignar prioridades a los requerimientos de software
Unidad 3. Diseño de la arquitectura basado en la funcionalidad (QASAR) Diseña una arquitectura de software usando la metodología para diseño basado en funcionalidad.	• Trabajar en equipo para elaborar el diseño de una arquitectura de software basada en un conjunto de requisitos funcionales y de calidad. El diseño deberá seguir la metodología propuesta obteniendo: - Contexto del sistema - Casos de uso - Instancias del sistema - Identificación de arquetipos - Modelo arquitectónico propuesto - Modelo de clases - Modelo de datos
Unidad 4. Evaluación de la arquitectura de software Evalúa la arquitectura de software usando el método de evaluación basado en atributos de calidad.	• Generar el conjunto de escenarios para la evaluación de diferentes requisitos de calidad • Incorporar un conjunto nuevo de requisitos funcionales y evaluar la arquitectura del proyecto definida en la práctica anterior respecto a los requisitos de calidad. Obtener datos, que permitan objetivamente tomar decisiones acerca de la arquitectura.

Unidad 5. Transformación de arquitecturas de software Genera una nueva arquitectura de software que cumpla con los requisitos funcionales aplicando los métodos de transformación de arquitecturas.	• Transformar la arquitectura (el número de veces que sea necesario) hasta obtener una arquitectura que satisfaga los requisitos establecidos. • Presentar los diferentes modelos y adecuaciones propuestos en la transformación, así como su evaluación y documentación de las versiones. • Exponer en clase el proyecto completo.

8. SUGERENCIAS DIDÁCTICAS PARA EL DESARROLLO DE COMPETENCIAS GENÉRICAS.

El profesor debe:

- Ser conocedor de la disciplina que está bajo su responsabilidad, conocer su origen y desarrollo, así como el estado actual en la práctica para considerar este conocimiento al abordar los temas.
- Desarrollar la capacidad para coordinar y trabajar en equipo.
- Orientar el trabajo del estudiante y desarrollar en él la autonomía, el trabajo cooperativo y la toma de decisiones.
- Mostrar flexibilidad en el seguimiento del proceso formativo y propiciar la interacción entre los estudiantes.
- Tomar en cuenta el conocimiento de los estudiantes como punto de partida y como obstáculo para la construcción de nuevos conocimientos.
- Fomentar actividades grupales que propicien la comunicación, el intercambio argumentado de ideas, la reflexión, la integración y la colaboración de y entre los estudiantes. Ejemplo: al socializar los resultados de las investigaciones y las experiencias prácticas solicitadas como trabajo extra clase.
- Observar y analizar fenómenos y problemáticas propias del campo ocupacional. Ejemplo: el proyecto propuesto que se realizará durante el curso.
- Propiciar el desarrollo de capacidades intelectuales relacionadas con la lectura, la escritura y la expresión oral. Ejemplos: trabajar las actividades prácticas a través de guías escritas, redactar reportes e informes de las actividades realizadas durante el desarrollo del proyecto, exponer al grupo las conclusiones obtenidas durante las observaciones.
- Propiciar el desarrollo de actividades intelectuales de inducción-deducción y análisis-síntesis, que encaminen hacia la investigación.
- Desarrollar actividades de aprendizaje que propicien la aplicación de los conceptos, modelos, técnicas y metodologías que se van aprendiendo en el desarrollo de la asignatura.

- Proponer problemas que permitan al estudiante la integración de contenidos de la asignatura y entre distintas asignaturas, para su análisis y solución.
- Cuando los temas lo requieran, utilizar medios audiovisuales para una mejor comprensión del estudiante.
- Propiciar el uso de las nuevas tecnologías en el desarrollo de la asignatura.

9. SUGERENCIAS DE EVALUACIÓN

Diagnóstica	Formativa	Sumativa

• Evidencia: Auto-presentación.
  • Fotografía.
  • Nombre completo
  • Antecedentes académicos
  • Hábitos de estudio.
  • Hobbies o pasatiempos.
  • Motivos para estudiar esta carrera.
  • Intereses profesionales
	• Rúbrica para evaluar el desempeño de los estudiantes en equipo durante el desarrollo del proyecto semestral y las actividades en clase.
	• Tabla de resultados por equipos y del grupo.
		• 45% proyecto semestral (Rúbrica de proyecto en tres etapas)
		• 15% Trabajo de investigación y tareas (Rúbrica de trabajos y tareas)
		• 30% examen teórico-práctico (Respuestas de cuestionario)
		• 10% participación en clase (Lista de cotejo de actividades)

10. FUENTES DE INFORMACIÓN

Libro de texto

1. Jan Bosch, Design and Use of Software Architecture, Addison Wesley, 2000.
2. David Garlan, Mary Shawn. An introduction to Software Architecture. Technical report CMU-CS-94-166
3. The 4+1 View Model of Software Architecture (white paper). Philippe Kruchten. IEEE Software 12 (6) November 1995, pp. 42-50
4. Software Architecture in Practice (3rd Edition) (SEI Series in Software Engineering)by Len Bass, Paul Clements and Rick Kazman Addison-Wesley Professional; 3 edition (September 25, 2012)

Lecturas complementarias

1. Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives (2nd Edition) by Nick Rozanski and Eóin Woods, Addison-Wesley Professional; 2 edition (October 25, 2011)
2. Software Architecture: Foundations, Theory, and Practice by R. N. Taylor, N. Medvidovic and E. M. Dashofy, Wiley; 1 edition (January 9, 2009)
3. The Process of Software Architecting, Peter Eeles, Peter Cripps
4. Documenting Software Architectures: Views and Beyond (2nd Edition) by Paul Clements, Felix Bachmann, Len Bass and David Garlan, Addison-Wesley Professional; 2 edition (October 5, 2010)
5. Architecture Complete: A Programmer's Guide to Software Architecture and Design Michael Gualtieri , Jason Darrow, Wrox; 1 edition (August 28, 2012)
6. Mary Shaw and David Garlan. Software Architecture: Perspectives on an Emerging Discipline, Prentice-Hall, 1996.

11. PERFIL DEL PROFESOR QUE IMPARTIRÁ LA MATERIA

- Ingeniero en Sistema computacionales, Licenciado en Informática o carrera afín con experiencia práctica en desarrollo de sistemas de software.
- Maestro en ciencias de la computación
- Doctor en ciencias de la computación