1.- DATOS DE LA ASIGNATURA

Nombre de la asignatura:	Estilos y Patrones de arquitectura y de diseño de software
Carrera:	Ingeniería en Sistemas Computacionales
Clave de la asignatura:	ARC-1305
(Créditos) SATCA[1]	2-2-4

2.- PRESENTACIÓN

Caracterización de la asignatura.

La arquitectura del software manifiesta las decisiones tempranas del diseño del software, que influirán en el desarrollo del sistema, su distribución y futura evolución. En términos generales, la arquitectura define la estructura que deberá tener el sistema para poder cumplir con los requerimientos del cliente. Por su parte, los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

En esta asignatura, el estudiante obtendrá herramientas para diseñar soluciones de software para problemas complejos, de forma tal que dichas soluciones sean robustas, y cumplan con criterios de calidad tales como mantenibilidad y escalabilidad.

Intención didáctica.

En la primera unidad se abordan los aspectos introductorios de los estilos y patrones de arquitectura y diseño de software, empezando por las definiciones de estilo y patrón, y las diferencias entre ambos conceptos, y su importancia para la construcción de software con calidad. Se finaliza esta unidad estudiando diversos estándares relativos a la arquitectura y el diseño de software, así como los aspectos básicos a tener en cuenta para la documentación patrones de arquitectura y diseño de software.

En la segunda unidad se estudian los principales estilos arquitectónicos de software, que darán elementos al alumno para elegir adecuadamente el estilo

arquitectónico según las características de la solución de software que se pretenda desarrollar.

En la tercera unidad se estudian los principales patrones de arquitectura para el software, clasificándolos según el tipo de problema que se busca resolver con dichos patrones.

La cuarta unidad aborda los patrones de diseño. Se estudian los principales tipos de patrones. Se analizarán las características de dichos patrones, de forma tal que el estudiante sea capaz de elegir el patrón adecuado según el problema de diseño que se requiera resolver.

La quinta unidad busca que el estudiante ponga en práctica los conocimientos adquiridos en unidades previas, al definir la arquitectura y elaborar el diseño de un sistema de software, tomando en cuenta las características de dicho sistema para decidir el mejor estilo arquitectónico, así como los patrones de arquitectura y de diseño a emplear. Se espera que el alumno pueda modelar el diseño del sistema usando los patrones elegidos en dichos modelos.

En la sexta unidad se busca que el alumno logre llevar a la implementación la arquitectura y diseño propuesto, de forma que esté conciente de las implicaciones del uso de estilos y patrones de arquitectura y diseño desde la concepción del software hasta su puesta en operación.

3.- COMPETENCIAS A DESARROLLAR

Competencias específicas:	**Competencias genéricas**
Conocer y entender las diferencias entre los conceptos de estilo y patrón de arquitectura y de diseño de software. Conocer la importancia del uso de estilos y patrones de arquitectura y diseño para la construcción de software de calidad. Conocer y aplicar estándares de arquitectura y diseño de software. Documentar adecuadamente los patrones de arquitectura y diseño empleados en la construcción de	<u>Competencias instrumentales:</u> ● Capacidad de análisis y síntesis. ● Capacidad de organizar y planificar. ● Comunicación oral y escrita. ● Habilidad para buscar y analizar información proveniente de fuentes diversas. ● Solución de problemas. <u>Competencias interpersonales:</u> ● Trabajo en equipo.

sistemas de software. Ser capaz de decidir adecuadamente el mejor estilo arquitectónico a emplear en la construcción de sistemas de software. Conocer y saber discernir entre diversos patrones arquitectónicos a emplear para la solución de problemas comunes en la construcicón de sistemas de software. Concoer y saber discernir entre diversos patrones de diseño a emplear para la solución de problemas comunes en el diseño de sistemas de software. Aplicar patrones de arquitectura y diseño durante el modelado de sistemas de software. Implementar correctamente los patrones de arquitectura y diseño empleados en la definición y diseño de una aplicación de software.	● Capacidad crítica y autocrítica. ● Capacidad de venta de ideas. <u>Competencias sistémicas:</u> ● Capacidad de aplicar los conocimientos en la práctica. ● Habilidades de investigación. ● Capacidad de aprender. ● Capacidad de generar nuevas ideas (creatividad). ● Habilidad para trabajar en forma autónoma. ● Búsqueda del logro.

4.- HISTORIA DEL PROGRAMA

Lugar y fecha de elaboración o revisión	**Participantes**	**Observaciones (cambios y justificación)**
Instituto Tecnológico de Hermosillo. Febrero 2013	Dr. Oscar Mario Rodríguez Elias M.C. Julio Flores	Jornadas Curriculares de la Academia de Sistemas y Computación para el diseño de la especialidad de la carrera de Ing. en Sistemas Computacionales en el Instituto Tecnológico de Hermosillo.

5.- OBJETIVO(S) GENERAL(ES) DEL CURSO (competencias específicas a desarrollar en el curso)

Conocer y aplicar adecuadamente los principales estilos y patrones de arquitectura y diseño que han sido propuestos para la solución de problemas comunes en sistemas de software, con el fin de construir software que cumpla con estándares de calidad.

6.- COMPETENCIAS PREVIAS

- Conocer los conceptos y generalidades de la arquitectura de software, así como los roles de un arquitecto de software.

- Identificar y Analizar necesidades de información para su representación, tratamiento, tratamiento y automatización para la toma de decisiones.

- Diseñar esquemas de bases de datos para generar soluciones al tratamiento de información.

- Desarrollar soluciones de software utilizando programación concurrente, programación de eventos, que soporte interfaz gráfica e incluya dispositivos móviles.

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

7.- TEMARIO

Unidad	Temas	Subtemas
1	**Introducción a los estilos y patrones arquitectónicos y de diseño de software**	1.1. Definición de estilos y patrones 1.2. Diferencia entre estilo y patrón 1.3. Estilos y patrones arquitectónicos 1.4. Estilos y patrones de diseño 1.5. Importancia del uso de estilos y patrones 1.6. Estándares arquitectónicos 1.7. Documentación de patrones arquitectónicos y de diseño
2	**Estilos arquitectónicos**	2.1. Arquitecturas midelware 2.2. Arquitecturas orientadas a servicios 2.3. Arquitecturas orientadas a aspectos 2.4. Arquitecturas orientadas a agentes 2.5. Arquitecturas dirigidas por modelos

		2.6. Líneas de productos de software 2.7. Documentación de la arquitectura
3	**Patrones arquitectónicos**	3.1. De estructura 3.1.1. Capas 3.1.2. Pipas y filtros 3.1.3. Blackboard 3.2. De distribución 3.2.1. Broker 3.2.2. Cliente servidor 3.2.3. Peer to Peer 3.3. De sistemas interactuvos 3.1. Modelo-Vista- Controlador 3.2. Controlador- Supervisor 3.3. Presentación- abstracción-control 3.4. De sistemas adaptables 3.4.1. Microckernel 3.4.2. Reflection
4	**Patrones de diseño**	4.1 Documentación de los patrones de diseño 4.2. Tipos de patrones 4.2.1. Básicos 4.2.2. Creación de objetos 4.2.3. De colecciones 4.2.4. De estructura 4.2.5. De comportamiento 4.2.6. De concurrencia 4.3. Patrones comunes 4.3.1. Interfaces y clases abstractas 4.3.2. Métodos de acceso 4.3.3. Observador (publisher - subscriber) 4.3.4. Decorador 4.3.5. Método Factory 4.3.6. Singleton 4.3.7. Command 4.3.8. Adaptador 4.3.9. Método Template 4.3.10. Iterator 4.3.11. Composite 4.3.12. Estado (State) 4.3.13. Proxy
5	**Diseñando con**	5.1. Identificación del problema

	patrones	5.2. Identificación de posibles patrones 5.3. Elección del patrón a utilizar 5.4. Introducción del patrón en el modelo de diseño 5.5. Documentación de la arquitectura y el diseño
6	**Implementando con patrones**	6.1. Implementación del modelo 6.2. Documentación del código 6.3. Implementando para reutilizar 6.4. Implementando para evolucionar 6.5. Implementando para escalar

8.- SUGERENCIAS DIDÁCTICAS (desarrollo de competencias genéricas)
El profesor debe:

• Fomentar el uso de las tecnologías de información y comunicación.
• Proyección de videos sobre sistemas de calidad para reflexionar y elaborar resúmenes y conclusiones sobre ellos.
• Realizar visitas a empresas que para observar los roles del Arquitecto de software, los modelos utilizados y alcances de las actividades de arquitectura de software.
• Realizar viajes de prácticas a empresas que apliquen ingeniería del software con el fin de conocer que normas tendrían que usar si deciden crear una empresa desarrolladora de software.
• Solicitar al estudiante que realice investigaciones en diversas fuentes de información sobre patrones de arquitectura y diseño de software.
• Relacionar los contenidos de esta asignatura con los obtenidos en las demás del plan de estudios y de la especialidad, reforzando la importancia de tener una visión y práctica interdisciplinaria para alcanzar las metas académicas, profesionales y empresariales.
• Elaborar un conjunto de casos prácticos.
• Cuando los temas lo requieran, utilizar medios audiovisuales para una mejor comprensión del estudiante.
• Integrar equipos de trabajo en donde se compartan conocimientos y experiencias académicas y laborales.
• Discutir en grupo la información generada por los equipos de trabajo.
• Propiciar el uso de las diferentes fuentes de información, tanto de índole primaria como secundaria.
• Elaboración de rúbricas.

9.- SUGERENCIAS DE EVALUACIÓN

- Solución de casos prácticos solicitados durante las actividades, así como sus conclusiones de forma escrita.
- Reportes de investigación de campo.
- Reportes de prácticas
- Ejercicios realizados.
- Tareas.
- Exposición.
- Participación en clase.
- Proyecto.
- Exámenes teóricos y/o prácticos.

10.- UNIDADES DE APRENDIZAJE

Unidad 1: Introducción a los estilos y patrones arquitectónicos y de diseño de software

Competencia específica a desarrollar	Actividades de Aprendizaje
Conocer y entender las diferencias entre los conceptos de estilo y patrón de arquitectura y de diseño de software. Conocer la importancia del uso de estilos y patrones de arquitectura y diseño para la construcción de software de calidad. Conocer y aplicar estándares de arquitectura y diseño de software. Documentar adecuadamente los patrones de arquitectura y diseño empleados en la construcción de sistemas de software.	1.1 Investigar los conceptos de estilo y patron de arquitectura y diseño de software especificando su diferencia. 1.2 Investigar sobre un caso de estudio que ejemplifique los problemas ocasionados por una mala arquitectura o diseño de software. 1.3 Discutir en grupo sobre los problemas de una mala arquitectura o mal diseño del software. 1.4 Investigar y elaborar un cuadro comparativo sobre los diversos estándares existentes relativos a la arquitectura y diseño del software. 1.5 Elaborar una plantilla con los elementos principales a considerar en la documentación de la arquitectura del software. 1.6 Elaborar una plantilla con los elementos principales a considerar para

	la documentación de los patrones de arquitectura y diseño del software.

Unidad 2: Estilos arquitectónicos

Competencia específica a desarrollar	**Actividades de Aprendizaje**
Ser capaz de decidir adecuadamente el mejor estilo arquitectónico a emplear en la construcción de sistemas de software. Documentar adecuadamente los patrones de arquitectura y diseño empleados en la construcción de sistemas de software.	2.1 Investigar sobre ejemplos de sistemas de software construidos con base en cada un de los distintos estilos de arquitectura. 2.2 Elaborar un cuadro sinóptico que identifique las principales diferencias y similitudes entre los distintos estilos de arquitecturas de software, así como el tipo de problema que busca resolver cada estilo. 2.3 Elaborar un diagrama que muestre la evolución histórica de los estilos de arquitecturas de software. 2.4 Investigar sobre el concepto de línea de productos de software. 2.5 Discutir en clase el concepto de líneas de productos de software y sus implicaciones para las empresas de desarrollo. 2.6 Elaborar una plantilla que especificque los principales elementos que debe contener la documentación de la arquitectura de un sistema de software.

Unidad 3: Patrones arquitectónicos

Competencia específica a desarrollar	**Actividades de Aprendizaje**
Conocer y saber discernir entre diversos patrones arquitectónicos a emplear para la solución de problemas comunes en la construcicón de sistemas de software.	3.1 Investigar qué son los patrones de arquitectura, para qué sirven, y cómo se documentan. 3.2 Elaborar un mapa conceptual que

Documentar adecuadamente los patrones de arquitectura y diseño empleados en la construcción de sistemas de software.	describa y clasifique los diversos patrones arquitectónicos. 3.3 Analizar y discutir en grupo el tipo de problema que busca resolver cada patrón arquitectónico. 3.4 Proponer al menos un ejemplo de sistema en el que pudiera ser aplicado cada tipo de patrón arquitectónico.

Unidad 4: Patrones de diseño

Competencia específica a desarrollar	**Actividades de Aprendizaje**
Conocer y saber discernir entre diversos patrones de diseño a emplear para la solución de problemas comunes en el diseño de sistemas de software. Documentar adecuadamente los patrones de arquitectura y diseño empleados en la construcción de sistemas de software.	Diseñar un mapa mental a cerca del proceso de documentación de los Patrones de diseño que incluya lo siguiente: Tipos de patrones básicos y comunes. Elaborar un cuadro sinóptico a cerca de diferencias y semejanza entre los diferentes tipos de patrones. Analizar casos reales y generar un ensayo donde se reflejen las conclusiones

Unidad 5: Diseñando con patrones

Competencia específica a desarrollar	**Actividades de Aprendizaje**
Aplicar patrones de arquitectura y diseño durante el modelado de sistemas de software. Documentar adecuadamente los patrones de arquitectura y diseño empleados en la construcción de sistemas de software.	Generar un diagrama de flujo del proceso de diseño con patrones explicando cada uno de sus elementos. Implementar el diseño con patrones en un proyecto desarrollado anteriormente.

Unidad 6: Implementando con patrones

Competencia específica a desarrollar	Actividades de Aprendizaje
Implementar correctamente los patrones de arquitectura y diseño empleados en la definición y diseño de una aplicación de software.	Realizar las implementaciones correspondientes en el proyecto. Exponer el resultado de las implementaciones.

11.- FUENTES DE INFORMACIÓN

Fuentes impresas (libros)

Básicas:

- **Robert Hanmer**, ***Pattern-Oriented Software Architecture For Dummies***, **For Dummines, 2013.**
- **Eddie Burris, Programming in the Large with Design Patterns, Pretty Print Press, 2012.**
- **Partha Kuchana, *Software Architecture Design Patterns in Java*, AUERBACH, Boca Raton, USA, 2004.**
- **Elisabeth Freeman, Eric Freeman, Bert Bates and Kathy Sierra*, Head First Design Patterns* - O'Reilly, 2004.**
- **Jason McC. Smith, *Elemental Design Patterns*, Addison-Wesley, 2012.**
- **Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns. Elements *of Reusable Object-Oriented Software* - Addison Wesley, 1994.**
- **Larman, *UML y Patrones. Introducción al análisis y diseño orientado a objetos* - Prentice Hall, 2004.**

Complementarias:

- **Buschmann, Meunier, Rohnert, Sommerlad, Stal, *Pattern Oriented Software Architecture Volume 1: A System of Patterns* - Wiley, 1996.**
- **Schmidt, Stal, Rohnert, Buschmann, *Pattern Oriented Software Architecture Volume 2: Patterns for Concurrent and Networked Objects* - Wiley, 2000**
- **Kircher, Jain, *Pattern Oriented Software Architecture Volume 3: Patterns for Resource Management* - Wiley, 2004.**
- **Buschmann, Henney, Schmidt, *Pattern Oriented Software Architecture Volume 4: A***

Pattern Language for Distributed Computing **- Wiley, 2007.**

- **Buschmann, Henney, Schmidt, *Pattern Oriented Software Architecture Volume 5: On Patterns and Patterns Language* - Wiley, 2007.**
- ***AntiPatterns. Refactoring Software, Architectures and Projects in Crisis* - Brown, Malveau, McCormick Mowbray - Wiley**
- ***Patterns in Java* - Mark Grand - Wiley**