Universidad Autónoma del Estado de México
Unidad Académica Profesional Tianguistenco
Licenciatura en Ingeniería de Software

Guía pedagógica:

Arquitectura de software

Elaboró: LSCA Carlos Alberto García Acevedo

Fecha: 30/junio/2015

Fecha de aprobación | H. Consejo académico | H. Consejo de Gobierno

Índice

I. Datos de identificación

Espacio educativo donde se imparte	**Unidad Académica Profesional Tianguistenco**			
Licenciatura	**Licenciatura en Ingeniería de Software**			
Unidad de aprendizaje	**Arquitectura de software**	**Clave**	**L40836**	
Carga académica	**2**	**2**	**4**	**6**
	Horas teóricas	Horas prácticas	Total de horas	Créditos

Período escolar en que se ubica	1	2	3	4	5	6	7	8	9

Seriación	Ninguna	Ninguna
	UA Antecedente	UA Consecuente

Tipo de Unidad de Aprendizaje

Curso		Curso taller	
Seminario	✓	Taller	
Laboratorio		Práctica profesional	
Otro tipo (especificar)			

Modalidad educativa

Escolarizada. Sistema rígido		No escolarizada. Sistema virtual	
Escolarizada. Sistema flexible	✓	No escolarizada. Sistema a distancia	
No escolarizada. Sistema abierto		Mixta (especificar)	

Formación común

Formación equivalente	**Unidad de Aprendizaje**

II. Presentación de la guía pedagógica

Es importante que el docente cuente con una Guía pedagógica para realizar su actividad, la cual tiene como propósito complementar el programa de estudios y al mismo tiempo orientar el proceso y ejecución de su intervención educativa, permitiéndole reflexionar sobre las actividades que va a proponer a los alumnos, los medios didácticos que se utilizará, proponer prácticas y solucionar problemas que se presentan en la utilización de las computadoras y el software utilizados con fines educativos durante las sesiones de aprendizaje.

En la presente Guía pedagógica de la Unidad de Aprendizaje Arquitectura de software se presenta una propuesta didáctica que permitirán al docente desarrollar su programa y enriquecerlo con nuevos conocimientos y actualizaciones, en congruencia con el modelo de competencias y la enseñanza constructivista, a fin de contribuir a la formación de los profesionales de la Carrera de Ingeniería de Software, atendiendo a sus particulares estilos de aprendizaje.

Esta unidad de aprendizaje comprende conocer e identificar los diferentes fundamentos asociados a la arquitectura de software así como establecer criterios para aplicar el patrón de diseño más adecuado en cada caso de construcción, mejora o actualización, fundamentándose en los sistemas basados en componentes establecidos por la ingeniería de software. El entorno de desarrollo de la Unidad de aprendizaje es en el aula y en laboratorio.

Para lograr lo anterior el estudiante debe conocer conceptos y fundamentos de Ingeniería de software, así como contar con técnicas para el análisis y el diseño. También debe ser capaz de aplicar, distinguir y proponer adecuadamente las técnicas para obtener requisitos y especificaciones enfocados a solucionar problemas prácticos así como tener la habilidad de emplearlos para crear, mejorar o actualizar software con una actitud de valoración de la contribución de las diversas disciplinas de desarrollo de programas que buscan simplificar el trabajo humano.

La unidad de aprendizaje pertenece al área curricular Programación e ingeniería de software, núcleo que promueve una relación inter y trans-disciplinaria al agrupar unidades de aprendizaje que definen fundamentalmente la formación práctica o aplicativa de la profesión y el desarrollo de competencias específicas. También provee al alumno de escenarios educativos para la integración, aplicación y desarrollo de los conocimientos, habilidades y actitudes que le permitan el desempeño de las funciones, tareas y resultados requeridos en los diferentes ámbitos de intervención profesional.

Es importante el empeño e interés del alumno para dominar y adquirir habilidades en el manejo de la computadora y del software que se utilizarán en la unidad de aprendizaje, así como las propuestas para integrar conocimientos técnicos, prácticos, disciplinarios, pedagógicos o didácticos en el proceso y ejecución del enriquecimiento educativo, de esta manera el docente, facilitador del aprendizaje, podrá construir programas de estudios de una unidad de aprendizaje por objetivos académicos y profesionales.

El estudiante amplía su visión al percibir cómo la industria está actualmente empleando diversas técnicas para desarrollar software en un entorno tan cambiante y exigente. Al mismo tiempo valora y aplica los conocimientos y habilidades adquiridos en su carrera hasta este punto para resolver diferentes situaciones prácticas mientras va haciéndose más consciente del relevante papel que desempeñan hoy las técnicas de arquitectura en el desarrollo, así como la misma Ingeniería de software.

III. Ubicación de la unidad de aprendizaje en el mapa curricular

Núcleo de formación:	Sustantivo
Área Curricular:	Programación e ingeniería de software
Carácter de la UA:	Obligatoria

IV. Objetivos de la formación profesional.

Objetivos del programa educativo:

Formar profesionistas con los conocimientos, habilidades y actitudes necesarios para contribuir en cualquiera de los procesos de la Ingeniería de Software para proponer soluciones de calidad al manejo automatizado de información dentro de las organizaciones, aplicando un enfoque sistemático, disciplinado y cuantificado en la formulación, planeación, análisis, diseño, implantación y mantenimiento de software, así como la generación de conocimiento, metodologías y métricas en torno a la Ingeniería de Software .

Objetivos del núcleo de formación:

Núcleo de formación Sustantivo.

Desarrollar en el alumno/a el dominio teórico, metodológico y axiológico del campo de conocimiento donde se inserta la profesión.

Objetivos del área curricular o disciplinaria:

Programación e ingeniería de software:

Emplear las técnicas de diseño necesarias para formular y expresar algoritmos computacionales para los cuales existe solución, estructurando en forma eficiente la representación elegida para la información de que se trate en la construcción de programas en forma correcta y con base en metodologías existentes.

Comprender las diferentes filosofías, conceptos, metodologías y técnicas utilizadas para la construcción de sistemas de software, considerando sus requerimientos, análisis y modelado, diseño, validación, verificación y calidad.

Analizar los diferentes elementos que inciden en la creación de productos de software desde una perspectiva de desarrollo industrial, incluyendo aspectos de eficiencia del proceso de creación, uso de herramientas automatizadas para su desarrollo, robustez, adaptabilidad, análisis de costos y tiempos y comercialización, entre otros.

V. Objetivos de la unidad de aprendizaje.

Aplicar los fundamentos asociados a la arquitectura de software de gran escala, marcos de referencia y patrones de diseño para la construcción de sistemas basados en componentes.

VI. Contenidos de la unidad de aprendizaje, y su organización.

<table>
<tr><td colspan="3">Unidad 1. Importancia de la arquitectura para el diseño y desarrollo de software.</td></tr>
<tr><td colspan="3">Objetivo:
Argumentar y evaluar la contribución de la Arquitectura de software, así como las principales disciplinas de Diseño existentes, para el desarrollar, mejorar o actualizar programas, comprendiendo e identificando los requerimientos específicos de su aplicación y reconociendo los rasgos que los categorizan.</td></tr>
<tr><td colspan="3">Contenidos:
1.1 Definiciones prácticas de software y arquitectura
1.2 Importancia de la arquitectura para el desarrollo de software
1.3 Relación entre la arquitectura y el diseño</td></tr>
<tr><td colspan="3">Métodos, estrategias y recursos educativos</td></tr>
<tr><td colspan="3">Discusión y análisis, lluvia de ideas, trabajo en equipos, mapa conceptual, cuadro sinóptico, método de caso, ABP, síntesis, exposición y clase magistral.</td></tr>
<tr><td colspan="3">Actividades de enseñanza y de aprendizaje</td></tr>
<tr><td>Inicio</td><td>Desarrollo</td><td>Cierre</td></tr>
<tr><td>1.1. Lluvia de ideas e investigación personal. Elabora un diagrama con los principales técnicas de arquitectura y cómo pueden ser afines con software.</td><td>1.1. Clase magistral. Anotación de los puntos más relevantes expuestos de la necesidad de la industria del software de contar con el apoyo de otras disciplinas para mejorar sus desarrollos. Entrega síntesis.</td><td>1.1. En equipos de trabajo elabora mapa conceptual de la necesidad inicial de la industria de adecuar técnicas usadas por otras disciplinas en el desarrollo de software de calidad.</td></tr>
<tr><td>1.2. Método de caso. Discusión en grupos pequeños sobre las técnicas de la arquitectura que puedan emplearse a la industria del software. Entrega reporte.</td><td>1.2. Debate donde se analizan y proponen diferentes puntos de vista sobre las principales técnicas y métodos arquitectónicos que apoyan a la industria del software. Entrega conclusiones.</td><td>1.2. Elabora cuadro sinóptico sobre cómo aportan al diseño, actualización y soporte de software las principales técnicas de arquitectura.</td></tr>
</table>

1.3. Clase magistral. Anotación de los puntos más relevantes expuestos de los aspectos afines entre la arquitectura como disciplina y el diseño de software. Entrega resumen.	1.3. Aprendizaje basado en problemas. Mediante investigación y esquemas el alumno identificará el papel y la importancia de la arquitectura como disciplina para desarrollar diseños adecuados de software.	1.3. En equipos de trabajo. Exposición y reporte donde se valora el papel que desempeñan hoy los métodos y estándares arquitectónicos en los desarrollos de buenos diseños de software.
(2 Hrs.)	**(5 Hrs.)**	**(3 Hrs.)**

Escenarios y recursos para el aprendizaje (uso del alumno)	
Escenarios	**Recursos**
Aula	Pintarrón, textos de apoyo, internet, computadora, cañón y diapositivas.

Unidad 2. Arquitectura de software.		
Objetivo: Reconocer, evaluar y determinar el alcance y los límites propios de la arquitectura de software y de los patrones que emplea para el diseño y desarrollo. Analizar y recomendar los patrones de diseño más convenientes para diferentes casos en función de las necesidades de desarrollo, identificando los ámbitos de aplicación.		
Contenidos: **2.1 Diseño de software** **2.2 Tecnología de software** **2.3 Teoría e historia de la arquitectura de software** **2.4 Actividades de un arquitecto de software**		
Métodos, estrategias y recursos educativos		
Discusión y análisis, investigaciones, trabajo en equipos, cuadro comparativo, exposición y clase magistral.		
Actividades de enseñanza y de aprendizaje		
Inicio	**Desarrollo**	**Cierre**
2.1. y 2.2. Clase magistral. Anotación de los puntos más relevantes expuestos de las características de lo que son las tecnologías y el diseño de software. Presenta resumen.	2.1. y 2.2. Investigación de las características generales del diseño y las tecnologías de software y entrega un cuadro sinóptico con los resultados.	2.1. y 2.2. En equipos de trabajo. Exposición y reporte donde presentan ventajas, desventajas y usos del las diversas técnicas para elaborar diseños de software.

2.3.y 2.4. Clase magistral. Anotación de los puntos más relevantes expuestos de los antecedentes y actividades de la arquitectura de software. Entrega resumen.	2.3.y 2.4 Investigación y análisis aportando diferentes puntos de vista sobre las áreas en las que debe o no involucrarse la arquitectura de software. Presenta esquema con las principales características.	2.3.y 2.4 Mediante un cuadro comparativo el alumno conoce e identifica los antecedentes y las diferentes actividades de un arquitecto de software así como sus usos en la práctica.
(2 Hrs.)	**(10 Hrs.)**	**(2 Hrs.)**

Escenarios y recursos para el aprendizaje (uso del alumno)	
Escenarios	**Recursos**
Aula Laboratorio	Pintarrón, textos de apoyo, internet, computadora, cañón y diapositivas.

Unidad 3. Modelos de representación.		
Objetivo: Distinguir los rasgos particulares de cada modelo de representación así como los lenguajes de descripción empleados por la arquitectura, reconociendo y evaluando las ventajas e inconvenientes de cada uno. Identificar y evaluar las condiciones y requisitos establecidos por la industria del software para proponer y aplicar el patrón o diseño más adecuado.		
Contenidos: **3.1 Lenguajes de descripción de arquitecturas** **3.2 Patrones de software** **3.3 Modelos de objetivo, de forma, de función, de desempeño, de datos y administrativos**		
Métodos, estrategias y recursos educativos		
Discusión y análisis, investigaciones, trabajo en equipos, cuadro comparativo, exposición y clase magistral.		
Actividades de enseñanza y de aprendizaje		
Inicio	**Desarrollo**	**Cierre**
3.1. y 3.2. Clase magistral. Anotación de los puntos más relevantes expuestos de las características de los lenguajes de descripción y patrones de software. Presenta resumen.	3.1. y 3.2. Investigación de las características de los diferentes lenguajes de descripción, así como de los patrones de software y entrega un cuadro sinóptico con los resultados.	3.1. y 3.2. En equipos de trabajo. Exposición y reporte donde presentan el alcance, los recursos y usos de los ADL y sus patrones de software correspondientes.

3.3. Clase magistral. Anotación de los puntos más relevantes expuestos de las características de los diferentes modelos arquitectónicos. Entrega resumen.	3.3. Investigación y análisis aportando diferentes puntos de vista sobre las bases y características de cada uno de las modelos. Presenta esquema con los principales atributos.	3.3. Mediante un cuadro comparativo el alumno conoce e identifica los diferentes modelos arquitectónicos así como sus adecuaciones en la práctica.
(2 Hrs.)	**(8 Hrs.)**	**(4 Hrs.)**

Escenarios y recursos para el aprendizaje (uso del alumno)	
Escenarios	**Recursos**
Aula Laboratorio	Pintarrón, textos de apoyo, internet, computadora, cañón y diapositivas.

Unidad 4. Atributos de calidad en el diseño de software.		
Objetivo: Analizar la importancia de emplear métricas de calidad, así como conocer las técnicas, mediciones y estimaciones pertinentes aplicadas al diseño, determinando el impacto que tendrá e las siguientes etapas del desarrollo. Explicar la importancia de la evaluación de los patrones y del diseño de un sistema antes de iniciar su codificación.		
Contenidos: **4.1 Concepto de calidad de software** **4.2 Tiempo de vida y tiempo de ejecución** **4.3 Atributos funcionales y no funcionales de software** **4.4 Evaluación de atributos** **4.5 Principios básicos y tareas esenciales**		
Métodos, estrategias y recursos educativos		
Discusión y análisis, lluvia de ideas, cuadro sinóptico, método de caso, ABP, Debate, trabajo en equipos, exposición y clase magistral.		
Actividades de enseñanza y de aprendizaje		
Inicio	**Desarrollo**	**Cierre**
4.1. y 4.2. Lluvia de ideas e investigación personal. Elabora un diagrama con las principales razones por los que es importante evaluar la calidad del diseño de software en sus diferentes tiempos.	4.1. y 4.2. Clase magistral. Anotación de los puntos más relevantes del efecto de evaluar bien o no en sus diversos tiempos el diseño arquitectónico antes de proceder a construir el software. Entrega resumen.	4.1. y 4.2. En equipos de trabajo. Elabora mapa conceptual con los principales criterios de evaluación del diseño en sus diferentes tiempos para lograr el posterior desarrollo de software de calidad.

4.3. y 4.4. Método de caso. Discusión en grupos pequeños sobre las principales formas de medir y estimar la calidad en la etapa de diseño del software. Entrega reporte.	4.3. y 4.4. Clase magistral. Anotación de los puntos expuestos de las principales técnicas, mediciones y estimaciones aplicables a la etapa de diseño del software. Entrega resumen.	4.3. y 4.4. Elabora cuadro sinóptico con los criterios que ayudan a evaluar y medir adecuadamente la calidad del diseño arquitectónico antes de continuar con el desarrollo.
4.5. Debate donde se analizan y proponen diferentes puntos de vista sobre los principios y tareas involucrados para implementar o no pruebas de calidad en el diseño de software. Entrega conclusiones.	4.5. Aprendizaje basado en problemas. Mediante investigación y un esquema el alumno identifica las tareas y los parámetros de calidad aplicables al diseño arquitectónico software.	4.5. En equipos de trabajo. Exposición y reporte donde se indican los beneficios de aplicar los principios y las tareas esenciales para el aseguramiento de calidad antes de liberar el diseño arquitectónico de software.
(2 Hrs.)	**(5 Hrs.)**	**(3 Hrs.)**

Escenarios y recursos para el aprendizaje (uso del alumno)	
Escenarios	**Recursos**
Aula Laboratorio	Pintarrón, textos de apoyo, internet, computadora, cañón y diapositivas.

Unidad 5. Ciclo de producción en arquitectura de software.
Objetivo: Aplicar las técnicas, estándares y criterios más convenientes para diseñar software, tomando en cuenta los requerimientos propios de la industria, así como del ciclo de producción de arquitectura de software. Seleccionar los patrones arquitectónicos más útiles para un caso de desarrollo de software y, de acuerdo a sus características propias proponer la solución más adecuada.
Contenidos: **5.1 Interés de la producción de software** **5.2 El ciclo de producción** **5.3 Desarrollo de software basado en Arquitectura**
Métodos, estrategias y recursos educativos
Taller, investigaciones, cuadro sinóptico, resolución de problemas y clase magistral.
Actividades de enseñanza y de aprendizaje

Inicio	Desarrollo	Cierre
5.1., y 5.2. En equipos de trabajo investiga las técnicas y criterios que se emplean por la industria para crear un diseño y su representación en particular para el ciclo de producción de un software específico. Entrega conclusiones.	5.1., y 5.2. Clase magistral. Anotación de los puntos más relevantes expuestos de los criterios e interés de producción de casos específicos para incluir en el desarrollo un diseño arquitectónico, de acuerdo a sus características propias. Entrega resumen.	5.1., y 5.2. Elabora cuadro sinóptico con los principales criterios de producción que ayudan a definir el diseño y su adecuada representación en casos de desarrollo con características específicas.
5.3. En equipos de trabajo seleccionan y proponen el caso y software para el que plantearán un diseño arquitectónico. Entrega propuesta de desarrollo.	5.3. Taller supervisado para resolver el problema planteado en equipos de trabajo de acuerdo a los criterios indicados. Entrega avances.	5.3. Exposición y entrega de trabajo final con la propuesta completa del diseño arquitectónico de software específico.
(2 Hrs.)	**(10 Hrs.)**	**(4 Hrs.)**

Escenarios y recursos para el aprendizaje (uso del alumno)	
Escenarios	**Recursos**
Aula Laboratorio	Pintarrón, textos de apoyo, internet, computadora, cañón y diapositivas.

VII. Acervo bibliográfico

Básico:

1. Taylor, R. N., Medvidovic, N., Dashofy, E.M. (2009), *Software Architecure: Foundations, Theory, and Practice*: John Wiley & Sons.
2. Bass, L., Clements, P., and Kazman, R. (1998) *Software Architecture in Practice*: Addison-Wesley. Reading Massachusets, USA.
3. Rechtin, E. and Maier M. (1997) *The Art of Systems Architecting*: CRC Press.

Complementario:

1. Coplien, J. O., Bjørnvig, G. (2010) *Learn Architecture for Agile Software Development*: John Wiley & Sons.
2. Buschmann, F., Meunier, R., Rohnert, H., Sommerland, P., and Stal, M. (1996) *Pattern-Oriented Software Architecture. A System of Patterns*: John Wiley & Sons.
3. Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994) *Design Patterns: Elements of Reusable Object-Oriented Systems*: Addison-Wesley.
4. Bennett, D. (1997) *Designing Hard Software. The Essential Tasks*: Manning Publication Co. Greenwich, Connecticut, USA.

VIII. Mapa curricular