TALLER DE TECNOLOGÍAS DE PRODUCCIÓN DE SOFTWARE (Opción A - Java)

Carrera/ Plan: Analista Programador Universitario

Analista Programador Universitario Plan 2021/Plan 2015/Plan 2007

Año 2024

Año: 3ro
Régimen de Cursada: *Semestral*
Carácter (Obligatoria/Optativa): Obligatoria
Correlativas: SI210, SI203, SI204, SI206, SI208, SI302
Profesor/es: Laura A. Fava, Jorge H. Rosso
Hs. semanales: 6 hs. (teoría y práctica)

FUNDAMENTACIÓN

Esta asignatura se dicta en el segundo semestre del último año de la carrera Analista Programador Universitario, recibe estudiantes que ya han adquirido como mínimo, conocimientos sobre algoritmos de programas, diferentes paradigmas de programación, entre ellos el paradigma orientado a objetos y han experimentado con diferentes mecanismos para persistir información, esto es, se han apropiado de conocimientos básicos que nos permiten abordar temas más avanzados e integradores.

Asumiendo esta adquisición de conocimientos previos y advirtiendo la proximidad de su graduación, en este espacio nos proponemos introducir a los alumnos en un esquema de producción de software realista de acuerdo a los estándares del arte, usando JAVA y software libre. Para ello además de enseñarles nuevas tecnologías JAVA para el desarrollo de aplicaciones basadas en web, en la última etapa de la cursada, se les propone resolver alguna problemática relacionada con desarrollo de software. Esta actividad, les permite a los estudiantes no sólo integrar y aplicar todos los conocimientos adquiridos, sino también, ser parte de un grupo de desarrollo de software y contar con un escenario real con necesidades específicas donde desempeñarse.

OBJETIVOS GENERALES

Introducir a los alumnos en un esquema de organización de producción de software, utilizando metodologías, prácticas y herramientas actualizadas y acordes con los estándares actuales.

Fomentar la práctica del alumno en esquemas de trabajo similares a los que se utilizan en las empresas de desarrollo de productos de software.

Ofrecer a los alumnos alternativas tecnológicas, siempre en base a herramientas de utilización actual en el mercado laboral.

COMPETENCIAS

1. Participar en el relevamiento y análisis de los procesos funcionales de una Organización, con la finalidad de que se diseñen los Sistemas de Información asociados, así como los Sistemas de Software que hagan a su funcionamiento.
2. Participar en el diseño, la implementación y mantenimiento de Sistemas de Software y sus Bases de Datos para empresas y organizaciones.
4. Participar como auxiliar en equipos de I/D en Informática.
5. Capacitar al personal técnico de las áreas informáticas de las organizaciones.
6. Evaluar la utilización, eficiencia y confiabilidad del equipamiento, de los sistemas de software y de los datos existentes en empresas y organizaciones.
7. Realizar tareas como auxiliar docente universitario en Informática.

CONTENIDOS MÍNIMOS (de acuerdo al Plan de Estudios)

Introducir un ambiente de desarrollo de software estandarizado (con herramientas integradas que den una visión homogénea y estandarizada de las aplicaciones, su interfaz gráfica, el acceso a las bases de datos y la interconexión entre aplicaciones), enfocado a un organismo o "clase" de empresa usuaria.
Practicar cómo usar el ambiente de desarrollo, aprender las diferencias que tiene con un ambiente de producción, ilustrando la metodología organizacional del pasaje de desarrollo a producción.
Practicar con documentación estandarizada mostrando cómo se pasa de una especificación al código ejecutable.
Ejemplificar la actividad del tester de aplicaciones. Metodología de trabajo y ambiente de prueba (diferencia con los otros ambientes).
Plantear el proceso estandarizado de desarrollo de software en una tecnología de uso en el mercado. Rol de la documentación en cada etapa.
Plantear el desarrollo de una solución a un problema real que ilustra todas las problemáticas antes descriptas.
Describir cuales son las principales características de un proceso de desarrollo de software con calidad (introduciendo los principios básicos de CMM o CMMI).

PROGRAMA ANALÍTICO

UNIDAD DIDÁCTICA I: Aplicaciones J2EE

Motivación: A lo largo del curso se trabajará en la construcción de aplicaciones basadas en web, aplicaciones que involucran el uso de la red, del protocolo HTTP y que ameritan un análisis del nuevo escenario donde ejecutan –hasta ahora los alumnos sólo han trabajado con aplicaciones locales-. ¿Cuáles son las tecnologías, los mecanismos, los aspectos que se deben tener en cuenta al momento de comenzar a desarrollar aplicaciones basadas en web?
Temas: Arquitectura de las aplicaciones web y componentes del estándar JEE (*Java Enterprise Edition*), en especial **Servlets** y **JavaServer Pages (JSP)**. Introducción a las tecnologías del lado del servidor que posibilitan la construcción de aplicaciones web dinámicas y a las tecnologías del lado del cliente: JavaScript y AJAX para implementar interfaces de usuarios gráficas y más eficientes. IDEs (Integration Development Environments) y *Frameworks* para JEE.

UNIDAD DIDÁCTICA II: Componentes JEE

Motivación: Aquí comienza el aprendizaje de la plataforma empresarial de JAVA o JEE. Se analizará en profundidad el estándar JEE 5 -*Java Enterprise Edition* 5-. Nos preguntaremos: ¿cuáles son las componentes del estándar JEE?, ¿para qué sirve cada una de ellas?, ¿por qué los Servlets son una parte vital del estándar Java EE?, ¿cómo integrar las componentes para desarrollar aplicaciones con arquitecturas modulares, extensibles y reusables?
Temas: ***Servlets***. Características generales. Servicios que proveen los Contenedores Web. Ciclo de vida de los Servlets: init(), service() y destroy(). El archivo descriptor de la aplicación: web.xml, *deploy* de Servlets. Parámetros de inicialización de servlets y de contexto. Redireccionamiento del requerimiento: sendRedirect() y delegación del requerimiento y de la respuesta: forward() e include().

Filtros. Características generales. Ciclo de vida de los Filtros: init(), doFilter() y destroy(). Configuración. El objeto FilterChain. Las clases wrappers. Programando requerimientos y respuestas customizados.

JavaServer Pages (JSP). ¿Qué son las páginas JSP? Ciclo de vida de una JSP: fase de traducción, fase de compilación y fase de ejecución. Elementos para construcción de JavaServer Pages: Elementos de scripting, directivas, acciones estándares, el lenguaje de expresiones (JSP EL) y Java Standard Tag Library (JSTL).

Sesiones. Métodos tradicionales y el objeto HTTPSession. ¿Cómo mantener el estado con el protocolo HTTP? Mecanismos de intercambio de ID para manejar sesiones: cookies y URL *rewriting.* Lectura y escritura de datos de las sesiones. Sesiones en ambientes multi-servidores.

UNIDAD DIDÁCTICA III: Persistencia de datos

Motivación: Por muchos años, la persistencia ha sido un tema de debate en la comunidad de Java. ¿Es la persistencia un problema que ha sido resuelto? En esta unidad se comienzan a analizar las alternativas de JAVA para persistencia

de datos. Se muestra inicialmente la manera más primitiva CRUD (create, read, update, delete) con SQL y JDBC, luego se presenta el uso de ORM (*Object-Relational Mapping*) para alcanzar mayor abstracción.
Temas: ¿Qué es la persistencia de datos? Mecanismos para persistir: Serialización, Conexión a Base de Datos usando JDBC (Java DataBase Connectivity) y ORM. Uso del patrón DAO (Data Access Object) para encapsular el acceso a datos.
Serialización. Su uso, ventajas y desventajas.
Java DataBase Connectivity. Tipos de Drivers JDBC. La API (Application Programming Interface) JDBC. Establecimiento de una Conexión. Ejecución de Sentencias SQL. Las clases Statement, PreparedStatement y CallableStatement. Pool de Conexiones. La interfaz DataSource.
Object-Relational Mapping. JPA Java Persistence API. Comprendiendo los Estándares. Hibernate y EJB 3.0.

UNIDAD DIDÁCTICA IV: Spring
Motivación: Si bien es posible escribir aplicaciones web codificando directamente con las APIs de JAVA estándares, es una buena práctica usar *frameworks* para hacerlo. El uso de *frameworks* no sólo acelera el proceso de desarrollo ofreciendo una capa de abstracción a las APIs, sino también, ayudan a desarrollar aplicaciones modulares, extensibles y reusables, basadas en su gran mayoría en algún patrón de diseño aprobado por la comunidad de desarrolladores. En esta etapa final del curso se enseña **Spring Core y Spring MVC,** *frameworks open source* de amplio uso en todos los ámbitos de desarrollo, maduros y con una comunidad de usuarios muy activa. Demostraremos que estos *frameworks* promueven el uso de buenas prácticas de desarrollo y facilitan la integración con los frameworks para persistencia vistos en la unidad anterior. Asimismo se enseñará la solución provista por Spring para persistencia, llamada Spring Data.
Temas: Evolución del framework Spring Core. Arquitectura del Framework. Inversión de Control (IoC) e Inyección de Dependencias (DI). Configuración de la aplicación por XML y por anotaciones.
Introducción a Spring MVC. Procesamiento de un requerimiento. Configuración y contextos de la aplicación. Internacionalización. Validación.
Implementación de web services mediante REST (Representational State Transfer). Principios de RESTful: identificación de recursos mediante URIs, uso de una interfaz uniforme y acotada. Implementando REST con Spring MVC. Spring Data.

UNIDAD DIDÁCTICA V: Angular
Motivación: En los últimos años la arquitectura está teniendo un cambio desde aplicaciones multipage a aplicaciones single-page (SPA). Angular es un framework *open source*, que usa una arquitectura MVC (Model-View- Controller) para crear aplicaciones SPA. Permite utilizar los mismos documentos HTML como templates para las vistas de la aplicación y además maneja el *databinding* del modelo de datos con la vista, dando dinamismo.
Temas: Características de Angular. Arquitectura de Angular. Componentes principales: Módulos, Componentes. Inyección de dependencias y Scopes. Data Binding. Servicios, Filtros, Directivas. Autenticación mediante tokens.

BIBLIOGRAFÏA

- Servlets and JavaServer Pages – The J2EE Technology Web Tier, Jayson Falkner, Kevin Jones. *Addison-Wesley*.
- Head First Servlets & JSP, Bryan Basham, Kathy Sierra, Bert Bates. O'Reilly
- Java Persistence with Hibernate, Christian Bauer, Gavin King. *Manning*, 2007.
- Documentación oficial de Angular, https://angular.io/docs

- *J2EE Tutorial*, Stephanie Bodoff, Dale Green, Kim Haase, Eric Jendrock, Mónica Pawlan, Beth Stearns. *Addison-Wesley*.
- *Database Programming with JDBC* and Java 2nd edition, George Reese, O'Reilly, 2002.
- *ng-book: The Complete Guide to Angular 11*, Felipe Coury, Ari Lerner, Carlos Taborda, Nic Raboy, Burke Holland. Fullstack.io.

METODOLOGÍA DE ENSEÑANZA

La materia consta de un encuentro semanal con una duración de 6 horas donde se integra el desarrollo conceptual que se utilizará en la producción práctica. Se organiza en una instancia inicial de explicación teórica donde se articulan los contenidos previos y se introducen y trabajan nuevos temas del programa que luego serán aplicados en la resolución de problemas prácticos.

La ***instancia teórica*** comienza con un repaso del tema previo, donde se indaga a los alumnos sobre las dificultades que han tenido en la aplicación de los conceptos teóricos en la práctica. Después del cierre del tema anterior se plantea el nuevo tema, se lo desarrolla y se intenta motivar a los estudiantes planteándose los desafíos de la instancia práctica que la sucede.
La ***instancia práctica*** comienza con una breve explicación de la propuesta o trabajo práctico del día, donde se selecciona uno de los ejercicios para que sea entregado en forma individual por cada estudiante. Si bien sólo se pide la entrega de uno de los ejercicios, la disponibilidad de máquinas y la buena relación docente/alumno permite realizar un seguimiento continuo de todos los ejercicios planteados en los trabajos prácticos.
Además de las producciones individuales semanales, se propone también una actividad grupal integradora en la etapa final de la cursada. Este trabajo final es un prototipo avanzado de un Sistema Informático que es resuelto por todos los grupos –de no más de tres alumnos- que componen la clase. Esta actividad tiene como objetivo, además de aplicar el conocimiento adquirido, funcionar como estímulo para implicar a los alumnos en sus aprendizajes.

Materiales Didácticos

Nuestra facultad nos provee soporte para la creación de cursos a través de la plataforma MOODLE (http://moodle.org/). Esta plataforma nos permite disponer de un lugar centralizado para compartir el material utilizado para todas las actividades, en especial, las clases teóricas con antelación a su exposición y los trabajos prácticos antes del comienzo de los mismos.
Para el desarrollo de las clases teóricas se utilizan diapositivas digitales previamente subidas a MOODLE. De manera similar, los enunciados de los trabajos prácticos también están disponibles en dicha plataforma antes de la práctica.
Para la instancia práctica los estudiantes deberán contar con una computadora para instalarse un entorno de desarrollo y una base de datos para realizar el proyecto propuesto por la cátedra.

EVALUACIÓN

Todas las actividades de clase son, potencialmente, actividades de evaluación, a partir de las cuales se puede revelar información útil para cumplir con las funciones básicas de la evaluación: retroalimentar la acción didáctica y acreditar los aprendizajes. Por este motivo, se realizan diferentes evaluaciones a saber a lo largo de la secuencia de enseñanza:
***Evaluación diagnóstica inicial*:** El objetivo de esta evaluación es obtener las características de los estudiantes en el momento de partida, de manera de poder planificar efectivamente el proceso de enseñanza-aprendizaje.
***Evaluación diagnóstica continua o formativa*:** Tiene como propósito facilitar el aprendizaje de los alumnos, no simplemente medir cuánto han aprendido. Para ello, al finalizar algunas unidades didácticas claves se les pide a los estudiantes que entreguen un trabajo individual relacionado con los temas de la unidad y con una complejidad levemente superior a los ejercicios desempeñados en las clases prácticas.
Estos trabajos motivan el aprendizaje de los estudiantes y promueven la auto evaluación, mientras que a los docentes nos permiten diagnosticar y remediar dificultades en el proceso de aprendizaje.

Evaluación sumativa: tiene como objetivo hacer un balance de lo aprendido a lo largo del proceso y es realizada con dos propósitos:

a. **acreditar si el estudiante alcanzó o no los objetivos mínimos**, esto es la aprobación de la cursada. Para aprobar la cursada un estudiante debe haber aprobado al menos un 80% de las evaluaciones continuas o ejercicios especiales.
b. **calificar al estudiante**, esto es la definición de una nota. Los alumnos tienen dos alternativas: promocionar con el desarrollo mínimo de un proyecto con nota 6 o continuar el desarrollo del proyecto para obtener una nota superior a 6.

CRONOGRAMA DE CLASES Y EVALUACIONES

Clase/Encuentro	Actividades teóricas y prácticas Contenidos	Evaluaciones previstas
Aplicaciones Web	Evolución de las aplicaciones web. Tecnologías Java: La plataforma empresarial Java (J2EE), Servidores J2EE, IDEs para desarrollo de aplicaciones Java. *Frameworks* para desarrollo de aplicaciones Java. Práctica sobre Servidores HTTP, protocolo HTTP, aplicaciones Web.	
Servlets	Características generales de los Servlets. Ciclo de vida de los Servlets. Servicios que proveen los Contenedores Web. Manejo de requerimientos y respuestas HTTP. Práctica de Servlets.	
Filtros	Características generales de los Filtros. Ciclo de vida de los Filtros. El objeto FilterChain Clases Wrappers. Requerimientos y respuestas customizados. Práctica sobre Filtros	
JavaServer Pages	Definición de páginas JavaServer Pages (JSP). Ciclo de vida de una JSP. Elementos para construcción de JavaServer Pages: elementos de scripting, directivas, acciones estándares. El lenguaje de expresiones (JSP EL). Práctica sobre JSP.	
MAVEN	Conceptos básicos. Gestión de aplicaciones Java SE/EE usando MAVEN. Archetypes. Creación de proyectos con Maven. Funcionamiento. Ciclo de Vida, Fases, Plugins Dependencias, repositorios.	**ENTREGABLE 1** Páginas Servlet, JSP y Sesiones **11/10/2024**
JavaServer Pages y Sesiones	¿Cómo mantener el estado con el protocolo HTTP? Mecanismos de intercambio de ID para manejar sesiones: cookies y URL *rewriting.* El objeto **HTTPSession:** ligar y eliminar elementos. Invalidar sesión. Soporte de sesiones en servlets y JSP: Sesiones en ambientes multi-servidores. Práctica sobre JSP y Sesiones.	**ENTREGABLE 2** Prototipo **18/10/2024**
Tecnologías del lado del cliente	JavaScript, Librería JQuery, JSON Bootstrap	
JDBC	Persistencia. Tipos de persistencia con JAVA: Serialización, JDBC&SQL, Mapeo desde el modelo de objetos al modelo relacional. El estándar EJB 3.0 & Hibernate. Definición de capas de acceso a datos (DAO)	**ENTREGABLE 3** Modelo de Datos **25/10/2024**
Hibernate	Hibernate. Arquitectura, Módulos. Ciclo de vida de objetos persistentes Mapeo nativo (usando XML) Mapeo con Anotaciones	
JPA	Estándares: EJB 3.0/JPA Motores JPA Mapeos JPA-QL Estructura de una aplicación usando JPA	
Spring Core	Evolución del framework Spring Core. Arquitectura del Framework. Inversión de Control (IoC) e Inyección de	**ENTREGABLE 4** Hibernate y JPA

	Dependencias (DI). Configuración de la aplicación por XML y por anotaciones.	**01/11/2024**
Spring y servicios REST	Implementación de web services mediante REST(Representational State Transfer). Principios de RESTful: identificación de recursos mediante URIs, uso de una interfaz uniforme y acotada. Implementando REST con Spring MVC.	

Contacto de la cátedra (mail, sitio WEB, plataforma virtual de gestión de cursos):

Laura Fava: lfava@info.unlp.edu.ar
Jorge Rosso: jrosso@info.unlp.edu.ar

Firma del/los profesor/es