UNCAUS
UNIVERSIDAD
NACIONAL DEL
CHACO AUSTRAL

Presidencia Roque Sáenz Peña, 22 de agosto de 2019

RESOLUCIÓN N° 141/19 - C.D.C.B. y A.

VISTO:

El Expediente **01-2019-02080**, iniciado por el Docente de la Carrera Ing. LEGUIZAMON Elvio Fabián; medio por el cual eleva la Propuesta de Programa Diseño de Sistemas de la Carrera de Ingeniería en Sistemas de Información de la Universidad Nacional del Chaco Austral, para su aprobación; y

CONSIDERANDO:

Que la mencionada Propuesta de Programa corresponde a la asignatura dictada en el tercer año de la carrera;

Que dicha propuesta es acorde con contenidos mínimos, carga horaria, objetivos planteados, forma de evaluación y bibliografía pertinente;

Que analizadas las actuaciones, el Consejo Departamental opina que lo solicitado se encuadra con lo establecido por el Reglamento Académico de Alumnos;

Lo aprobado en sesión de la fecha;

POR ELLO:

EL CONSEJO DEPARTAMENTAL
DEL DEPARTAMENTO DE CIENCIAS BÁSICAS Y APLICADAS
DE LA UNIVERSIDAD NACIONAL DEL CHACO AUSTRAL
RESUELVE:

ARTÍCULO 1°: Aprobar la Propuesta de Programa de la Asignatura Diseño de Sistemas de la Carrera de Ingeniería en Sistemas de Información del Departamento de Ciencias Básicas y Aplicadas de la Universidad Nacional del Chaco Austral, y que se adjunta Anexo que forma parte de la presente resolución.

ARTÍCULO 2°: Regístrese, comuníquese al Ing. LEGUIZAMON, Elvio Fabian y a las Áreas correspondientes. Cumplido, archívese.-

Mg. Ing. Enzo Gabriel JUDIS
Director de Departamento
Ciencias Básicas y Aplicadas

<table>
<tr><td colspan="2">UNCAUS
UNIVERSIDAD
NACIONAL DEL
CHACO AUSTRAL</td><td colspan="3">DISEÑO DE SISTEMAS</td></tr>
<tr><td colspan="2">Carga Horaria: 180 horas
Horas Teóricas: 72
Horas Prácticas: 108</td><td colspan="3">Programa vigente desde: 2019</td></tr>
<tr><td colspan="2">Carrera</td><td>Año</td><td colspan="2">Cuatrimestre</td></tr>
<tr><td colspan="2">Ingeniería en Sistemas de Información</td><td>Tercero</td><td colspan="2">Anual</td></tr>
<tr><td colspan="3">CORRELATIVA PRECEDENTE</td><td colspan="2">CORRELATIVA SUBSIGUIENTE</td></tr>
<tr><td colspan="3">Asignaturas</td><td colspan="2">Asignaturas</td></tr>
<tr><td colspan="2">Para cursar</td><td>Para rendir</td><td colspan="2" rowspan="2">Administración de recursos</td></tr>
<tr><td>Regularizada</td><td>Aprobada</td><td>Aprobada</td></tr>
<tr><td>Análisis de Sistemas</td><td>Sistemas y Organizaciones</td><td>Análisis de Sistemas</td><td colspan="2"></td></tr>
<tr><td>Paradigmas de Programación</td><td>Algoritmos y Estructuras de Datos</td><td>Paradigmas de Programación</td><td colspan="2">Simulación</td></tr>
<tr><td colspan="2">DOCENTES:</td><td colspan="3">Lic. Paszco Gustavo Ariel
Ing. Leguizamón Elvio Fabián</td></tr>
<tr><td colspan="2">FUNDAMENTACIÓN:</td><td colspan="3">El diseño y la construcción de sistemas están soportados por varios principios fundamentales: diseño de datos, diseño de la arquitectura de software, diseño de interfaces, diseño de componentes, diseño de procedimientos. Estos principios favorecen que los objetivos de calidad del software se alcancen con mayor facilidad. El ciclo de vida de desarrollo de sistemas facilita la creación de sistemas de calidad, partiendo desde el análisis de información de lo que se quiere desarrollar hasta llegar a la prueba, puesta a punto e implementación del software y su posterior mantenimiento.
En el diseño de sistemas se estudian los principios y técnicas que permiten construir arquitecturas de software correctas.
En primer lugar, se introducirá la fase de diseño y proceso de diseño, para posteriormente centrarse en los principios y conceptos fundamentales del diseño y programación de software, haciendo hincapié en técnicas que permitan alcanzar un diseño eficaz, basado en módulos altamente cohesionados, con bajo</td></tr>
</table>

	acoplamiento y construidos sobre la base de metodologías orientadas a objetos. Dentro del proceso de diseño de sistemas se tendrá en cuenta los efectos que pueda producir la introducción del nuevo sistema sobre el entorno en el que deba funcionar, adecuando los criterios de diseño a las características del mismo. En este contexto debemos tener en cuenta la adaptación de todo del software a las capacidades de las personas que van a utilizarlo, de forma que su operación sea sencilla, cómoda, efectiva y eficiente.
OBJETIVOS:	**GENERAL:** **Que los alumnos logren:** Comprender el desarrollo de la arquitectura de un software y del entorno tecnológico que van a dar soporte, junto con su documentación detallada teniendo en cuenta el ciclo de vida de desarrollo de sistemas. **ESPECÍFICOS:** **Que los alumnos logren:** • Identificar métodos y modelos orientados a objetos. • Aplicar el proceso unificado de desarrollo de software. • Operacionalizar la metodología orientada a objetos, actividades y ciclo de vida, UML. • Realizar un estudio específico contrastando patrones de creación, estructurales y de comportamiento. • Sintetizar la arquitectura de Diseño de Software, su importancia y contenidos. • Identificar variedades de Interfaces de usuario y sus usos apropiados. • Diseñar Métodos de captura eficiente y efectiva de datos. • Interpretar el uso de metodologías ágiles. • Reconocer el objetivo del prototipado y diferenciar los diferentes tipos de prototipos. • Analizar la importancia de verificación, validación y documentación de diseño en todo el Ciclo de Vida de Desarrollo de Sistemas.
CONTENIDOS MÍNIMOS:	Actividades de Diseño. Patrones de Diseño. Diseño de Arquitectura. Verificación y Validación del Diseño. Documentación de las Diferentes Etapas del Diseño. Diseño de Interfaces. Diseño de Procedimientos. Estrategias de Prototipado y de Ensamblaje de

	Componentes.
MÉTODOS PEDAGÓGICOS:	A través de los métodos aplicados se tiende al desarrollo del pensamiento crítico, reflexivo, práctico y autónomo sobre la práctica educativa; y al desarrollo de habilidades y competencias propias del modelado de sistemas de información orientados a objetos para su aplicación en problemas computacionales reales de negocios. • El aula se entiende como un espacio de taller para la construcción, en el que se trabaja interactuando permanentemente. • Exposición del docente: para orientar al alumno en el marco teórico de la asignatura. • Discusión dirigida: para debatir acerca de las propuestas de solución encontradas a los problemas. • Estudio dirigido mediante investigación. • Estudio de casos para retroalimentar desde un enfoque analítico mediante observaciones realizadas por cada estudiante. • Planteo, análisis, resolución y comentarios de trabajos prácticos en forma individual y grupal. • Planteo de un problema crítico real de sistemas y su resolución mediante programación orientada a objetos en base a laboratorios grupales.
MÉTODOS DE EVALUACIÓN: **MÉTODOS DE EVALUACIÓN:**	**EVALUACIÓN PARCIAL:** Las evaluaciones parciales consisten en contenidos teóricos, prácticos y de laboratorio sobre el programa de la materia. **EVALUACION DE TRABAJOS PRÁCTICOS Y LABORATORIOS:** Para evaluar los contenidos prácticos de la materia el alumno deberá presentar 7 (siete) Trabajos Prácticos y 4 (cuatro) Laboratorios, cuyas fechas de entrega estarán establecidas en las caratulas de los mismos. Si el alumno desaprueba o no entrega a término un total de 2(dos) prácticas (incluyendo Trabajos Prácticos y Laboratorio) se considera alumno LIBRE. **ESCALA DE VALORACIÓN:** La escala de valoración de los exámenes parciales será cuantitativa (Escala de 1 a 10) de un total de 4 (cuatro) exámenes parciales.

	Los trabajos prácticos y laboratorios tendrán carácter evaluativo, la escala de valoración de los mismos será cualitativo (Excelente – Muy Bueno – Aprobado - Desaprobado) siendo necesario la aprobación del 80% de ellos para la regularidad de la asignatura. **EXAMEN FINAL:** Consiste en la defensa de Proyecto Integrador y preguntas de tipo teórico, práctico y laboratorio sobre los contenidos de la materia. La asignatura no es promocional. Se aplica la normativa vigente. Res. 080/12.-C.S.-
PROGRAMA ANALÍTICO DE CONTENIDOS:	**UNIDAD N° 1. Introducción al Diseño de Sistemas.** Ciclo de Vida de Desarrollo de Sistemas, etapas. Métodos y Modelos: Conceptos. Principios de modelado de sistemas, modelado orientado a objetos. Tipos de Métodos Orientados a Objetos. Métodos de Análisis Orientados a Objetos. Métodos de Diseño Orientados a Objetos. Diseño de software y Diseño de sistemas. Diferencias y similitudes. El Proceso Unificado de Desarrollo de Software. Visión General de un Proceso Unificado. Proceso Conducido por Casos de Uso. Proceso Iterativo e Incremental. Estudio De Un Lenguaje OO: Entornos de Programación. Lenguajes. Sintaxis y Semántica. Especificación de Clases. Especificación de Herencia y Polimorfismo. Reescritura o sobrecarga. Encadenamiento tardío. Control de Acceso y Herencia. Grado de acoplamiento y grado de cohesión. **UNIDAD N° 2. Diseño de Sistemas Orientado a Objetos.** Metodología Orientada a Objetos. UML (Unified Modeling Languaje): Aspectos Dinámicos y Estáticos de un Sistema en UML: Diagramas de Paquetes, Diagramas de Clases, Diagramas de Componentes, Diagramas de Despliegue, Diagrama de Comportamiento, Casos de Uso, Diagramas de Actividades, Máquina de Estados. Diagramas de Interacción: Diagrama de Secuencia y Diagrama de Comunicación. Herramientas CASE (Computer Aided Software Engineering) en UML. Comparación con otras Metodologías. **UNIDAD N° 3. Diseño de Prototipos.** Prototipos: Concepto. Modelos de Prototipos. Estrategias de Prototipado: Prototipos para Pantallas,

PROGRAMA ANALÍTICO DE CONTENIDOS:	Procedimientos y Funciones. Errores en la Construcción. Uso de prototipos en el Ciclo de Vida de Desarrollo de Sistemas (CVDS). Herramientas para Construcción de Prototipos: de Consulta, Pantalla, Reportes, Recuperación de Datos y Diccionario de Datos. Herramientas para Aplicaciones Web: Axure RP, Herramientas CASE, Otros. Ensamblaje de Componentes. Ejemplos de Implementación. **UNIDAD N° 4. Introducción al Diseño de Datos.** Bases de datos: Tipos de bases de datos, relacional, Orientada a objetos. Diagramas de entidad relación. Modelo relacional. Concepto de persistir. Arquitectura de persistencia. Archivos XML. CSV. JSON. Diagramas de Clases orientados al Diseño Arquitectónico de Software. Diagramas de Clases Orientadas a Bases de Datos, Diferencia entre Jerarquía de Clases. Herramientas visuales de Base de Datos, MySQL WorkBench. Diagramas de Entidad Relación extendido, variables y constantes. Métodos. Trigger. Envío de mensajes. Creación y Destrucción de Objetos. Ejemplos de Implementación. **UNIDAD N° 5. Patrones de Diseño.** Introducción a los Patrones de Diseño. Conceptos. Diseño reutilizable. Patrón. Patrón de Diseño. Anti-patrón. Convenciones de Descripción de Patrones. Catálogo de Patrones de Diseño: Estudio Específico de los Patrones de Creación, Estructurales y de Comportamiento. Implementación de los Patrones en Lenguajes Orientados a Objetos. Ejemplos de Implementación. Diseño Reutilizable Utilizando Frameworks Orientados a Objetos: Conceptos y Propiedades de los Frameworks. Beneficios. Técnicas de Diseño. Uso y Evolución de Frameworks Orientados a Objetos. Framework para Aplicaciones Web. Patrón de Arquitectura Model-View-Controler (MVC). **UNIDAD N° 6. Diseño y Programación.** Diseño web, escritorio y aplicaciones. Lenguajes de Programación, diferencias. Diseño híbrido. Diseño y programación de plantillas web, templates con PHP y HTML5. FrameWork. Tecnología Boostrap. Técnica de desarrollo web para aplicaciones interactivas

PROGRAMA ANALÍTICO DE CONTENIDOS:	(RIA), tecnología AJAX. Funciones dinámicas con JQuery y AJAX. Vistas HTML y Boostrap, registros paginados, uso de DataTables. Diseño de formularios HTML. Diseño CSS. Librerías y funciones JavaScript, uso de BotBox, JSBarcode, JQueyPrint, otros. Seguridad, encriptación de datos, técnica HASH. Permisos de usuario. Vistas, consultas y estadísticas, gráficos DashBoard (Excel, pdf, otros). Reportes, librerías FPDF, HTML, otros. Implementación Local y red LAN. Servicios de Hosting. **UNIDAD N° 7. Arquitecturas de Diseño.** Problemas del Diseño de Software. Niveles de Diseño. Reutilización de software. Importancia. Mecanismos de reutilización. Tecnologías de reutilización. Reutilización de Diseño. Reutilización en Orientación a Objetos. Arquitectura de Software, Modelos. Niveles de Diseño de Software. Estilos Arquitectónicos. Lenguajes de Especificación de Arquitecturas. Validación de Arquitecturas. Arquitectura Orientada a Eventos y a Objetos, de Flujo de Datos, SOA (Service Oriented Architecture). Software basado en Componentes: Definición. Objetivos y Principios. Componentes y Arquitecturas de Sistemas. Arquitecturas de Componentes. Contratos. Proceso de Desarrollo. Ejemplos de implementación. **UNIDAD N° 8. Diseño de Interfaces y Procedimientos.** Interfaces y Procedimientos: Conceptos. Diseño de Interfaces de Usuario de Entrada/Salida. Interfaces Gráficas de Usuario(GUI). Tipos: Interfaces de Lenguaje Natural, de Pregunta y Respuesta, de Menús, de Formularios y Lenguaje de Comandos. Interfaces Web. Interfaces de Consulta. Diseño de Procedimientos. Procedimientos Precisos de Entrada de Datos. Captura de Datos. Métodos de Entrada de Datos. Validación. Ejemplos de Implementación. **UNIDAD N° 9. Documentación e Implementación.** Documentación de Sistemas. Documentación de Programas y Módulos. Definición de Normas y Procedimientos. Manuales de la Organización. Documentación Operativa. Manuales del

	Usuario, Manuales de Procedimiento de la Organización incluyendo el Sistema de Software. Sistemas de Documentación Online con acceso en la Web. Implementación de un Sistema. Planificación de la Implementación. Capacitación de los Usuarios. Operación en Paralelo. Asignación de recursos. Procedimiento de Actualización y mantenimiento. Auditoria. Ejemplos de Implementación.
PROGRAMA ANALÍTICO DE TRABAJOS PRÁCTICOS:	• **TRABAJO PRÁCTICO N° 1:** DISEÑO DE PROTOTIPOS. El trabajo consiste en diseñar, mediante un software de prototipado la representación limitada de un producto de software, cuya necesidad permite a las partes (cliente-desarrollador) probarlo en situaciones reales o explorar su uso. La práctica corresponde a la unidad N° 3 del programa analítico de contenidos. • **TRABAJO PRÁCTICO N° 2:** DIAGRAMAS DE ENTIDAD RELACIÓN EXTENDIDO. El trabajo práctico consiste en resolver sobre varios escenarios de complejidad media, entidades relevantes de un sistema de información, así como sus interrelaciones y propiedades. La práctica corresponde a la Unidad N° 4 y la unidad N° 6 del programa analítico de contenidos. • **TRABAJO PRÁCTICO N° 3:** DIAGRAMAS DE CLASES. El trabajo práctico consiste en desarrollar, sobre ejercicios de escenarios reales, modelos que permitan visualizar las relaciones entre las clases, ya que una clase es una descripción de conjunto de objetos que comparten los mismos atributos, operaciones, métodos, relaciones y semántica. La práctica corresponde a la Unidad N° 1 y la Unidad N° 2 del programa analítico de contenidos. • **TRABAJO PRÁCTICO N° 4:** UML (DIAGRAMAS DE ACTIVIDADES-DIAGRAMAS DE SECUENCIA- MÁQUINAS DE ESTADO). La práctica consiste el resolver, sobre varios escenarios reales, la especificación,

PROGRAMA ANALÍTICO DE TRABAJOS PRÁCTICOS:	visualización y documentación relativa al desarrollo de software utilizando Lenguaje Unificado de Modelado para resolver los aspectos estáticos y dinámicos de un sistema. La práctica corresponde a la Unidad N° 1 y la Unidad N° 2 del programa analítico de contenidos. • **TRABAJO PRÁCTICO N° 5:** PATRONES DE DISEÑO. La práctica consiste el resolver, sobre varios escenarios reales, problemas comunes en el desarrollo de software utilizando patrones creacionales, estructurales y de comportamiento. La práctica corresponde a la Unidad N° 5 del programa analítico de contenidos. • **TRABAJO PRÁCTICO N° 6:** ESTILOS ARQUITECTÓNICOS DE SOFTWARE. El trabajo práctico consiste en sintetizar, sobre varios ejercicios, estructuras de soluciones para la comunicación, coordinación y cooperación entre componentes de software y hardware. La práctica corresponde a la Unidad N° 7 del programa analítico de contenidos. • **TRABAJO PRÁCTICO N° 7:** DISEÑO DE INTERFACES. El trabajo consiste en diseñar, mediante un escenario real el entorno de operación persona-computadora, para que el usuario pueda controlar en forma efectiva las acciones de artefactos de hardware (computadoras, dispositivos móviles) La práctica corresponde a la Unidad N° 8 del programa analítico de contenidos. **LABORATORIOS** • **LABORATORIO N°1:** PROGRAMACIÓN PHP LINEAL Consiste en programación sobre el lenguaje PHP utilizando técnicas de programación lineal. Los ejercicios consisten en adquirir conocimiento de sentencias básicas, definición de variables, arrays, estructuras de control, funciones, entre otros. El laboratorio corresponde a la Unidad N° 6 del programa analítico de contenidos. • **LABORATORIO N°2:** PROGRAMACIÓN

	PHP ORIENTADO A OBJETOS. Consiste en programación sobre el lenguaje PHP utilizando técnicas de programación orientadas a objetos. Los ejercicios consisten en adquirir conocimiento de técnicas de programación orientadas a objetos de media complejidad. El laboratorio corresponde a la Unidad N° 6 y temas de la Unidad N° 2 del programa analítico de contenidos. • **LABORATORIO N°3:** PROGRAMACIÓN PHP ORIENTADO A OBJETOS. Consiste en programación sobre el lenguaje PHP utilizando técnicas de programación orientadas a objetos. Los ejercicios consisten en adquirir conocimiento de técnicas de programación orientadas a objetos de media complejidad agregando módulos de patrones de diseño. El laboratorio corresponde a la Unidad N° 6, temas de la Unidad N° 2 y temas de la Unidad N° 5 del programa analítico de contenidos. • **LABORATORIO EXTRA:** DESARROLLO DE CONTENIDOS DE LABORATORIO FINAL El trabajo consiste en desarrollar un software completo utilizando el lenguaje PHP integrando todas las unidades de la materia. El laboratorio corresponde a la integración de todas las unidades de la materia.
BIBLIOGRAFÍA:	• Bennett Simon, Farmer Ray , Mcrobb Steve, "Análisis y Diseño Orientado a Objetos", Editorial Mcgraw-Hill, 2007 • Arnedo Moreno Joan, Brinquez Jimenez Jordi , Garcia Barriocanal Elena , Piera I Eroles Miquel, Ramos Gonzalez Juan Jose , Riera I Terren Da, "Programación orientada a Objetos", Editorial Marcombo, 2007. • Jacobson Ivar, Booch Grady , Rumbaugh James, " El proceso unificado de desarrollo de software" , Editorial Pearson Educación, 2000 • Larman Craig. "Uml Y Patrones", Editorial Pearson Alhambra, 2000 • Booch Grady, "Análisis Y Diseño Orientado A Objetos. Aplicaciones", Editorial Pearson Educacion, 1996 • Fontela Carlos M. "Orientación A Objetos", Editorial Nueva Librería, 2008

	• Weitzenfeld Alfredo, "Ingeniería De Software Orientada A Objetos Con Uml, Java E Internet", Editorial Cengage Learning / Thomson Internacional, 2003 • Fontela Carlos, "Java y UML", Editorial Nueva Librería, 2004. • Booch, Grady, Jacobson Ivar, Rumbaugh James "El Lenguaje Unificado de Modelado Manual de Referencia", Editorial Adisoon Wesley, 2007. • Carlos Blé Jurado "Diseño Ágil con TDD" primera edición, Creative Commons, 2010. • Gamma Erich, Helm Richard, Johnson Ralph, Vlissides John "Patrones de Diseño Elementos de Software Orientado a Objetos Reutilizables", Editorial Addison Wesley, 2003.