Centro de Investigación en Matemáticas, CIMAT AC

Nombre Autorizado de la Institución

Maestría en Ingeniería de Software (MIS)

Nivel y Nombre del Plan de Estudios

Escolar

Modalidad Educativa

Agosto 2021
Vigencia

Estudios de Licenciatura o ingeniería en Ciencias de la Computación, Ingeniería de Software, en Informática o afines.
Antecedente Académico
06 Tecnologías de la información y la comunicación.
Área de Estudio

Clave del Plan de Estudios:	**2021**
Diseño Curricular:	**Flexible**
Total de Ciclos del Plan de Estudios:	**4 semestres**
Duración del Ciclo Escolar:	**16 semanas**
Carga Horaria a la Semana:	**12**

FIN DE APRENDIZAJE O FORMACIÓN

La Maestría en Ingeniería de Software, es un programa con carácter profesionalizante que ha sido creado con el propósito de formar profesionales multidisciplinarios capaces de aplicar conocimientos para la gestión de proyectos de software, entender y aplicar las mejores prácticas de la Ingeniería de Software para el desarrollo de soluciones competitivas a los problemas que se enfrente en su trayectoria profesional. También permitirá desempeñar diversos roles tales como emprendedor de base tecnológica, líder de equipo de proyectos de software, jefe de procesos y calidad de empresas de desarrollo de software, consultor de organizaciones que desarrollan software, como docente que transfiere técnicas y métodos avanzados de Ingeniería de Software a sus alumnos, o aspirante a realizar estudios de doctorado en temas de investigación en común. Además, este programa impulsa y desarrolla en el estudiante la realización de publicaciones y estancias de investigación nacionales e internacionales.

PERFIL DE EGRESO

Al término de sus estudios, el egresado de la Maestría en Ingeniería de Software será capaz de:

1. Analizar y resolver problemas de la industria con la ayuda de estadística, diseño de experimentos, investigación y metodologías para la especificación, desarrollo, prueba, mantenimiento y gestión de sistemas de software;
2. Aplicar principios de Ingeniería al proceso de desarrollo, reforzado con un estudio riguroso y profundo de los problemas de administración, diseño, desarrollo, entrega y mantenimiento de sistemas de software.
3. Diseñar y documentar los proyectos de software para que cumplan con los requisitos especificados, considerando además los requisitos no funcionales (v.g., la experiencia del usuario) y las limitaciones técnicas, de infraestructura o de recursos de las organizaciones.
4. Dirigir proyectos de desarrollo de sistemas asegurando la calidad del producto final.
5. Aplicar conocimientos de Ciencias de la Computación, de matemáticas, de ingeniería y de gestión combinados, en un marco metodológico adecuado, con la finalidad de obtener Productos de Software de alta calidad.

6. Identificar, seleccionar y monitorear tecnologías emergentes en el área de ingeniería de software y sistemas computacionales que permitan obtener una ventaja competitiva en las organizaciones.

PROGRAMA DE INVESTIGACIÓN

Objetivo del programa:

La Maestría en Ingeniería de Software es una maestría profesionalizante. Sin embargo, el programa promueve la investigación aplicada para resolver problemas de organizaciones, gobierno, empresas y sociedad en general a través de proyectos de Software.

Con este fin, el programa busca que el maestro en Ingeniería de Software: (i) tenga conocimientos sólidos que le permitan formular, desarrollar, dirigir y ejecutar experimentos en el área de Ingeniería de Software; (ii) sea capaz de investigar, recopilar, analizar e interpretar datos mediante la selección y aplicación de herramientas matemáticas, estadísticas y computacionales para establecer conclusiones con criterios científico-tecnológicos; (iii) conozca la problemática de las organizaciones y empresas a través del taller con la industria y; (iv) desarrolle soluciones para industria a través del proyecto con la industria.

A continuación se describen las líneas de Generación y Aplicación del Conocimiento de la maestría, sus objetivos particulares, temas de especialización y personal asociado.

Líneas de Generación y Aplicación del Conocimiento:

- **Tendencias y Aplicaciones en Ingeniería de Software.** Tiene por objetivo brindar conocimiento requerido, para la generación y/o aplicación de métodos, técnicas, herramientas y buenas prácticas de seguridad y calidad aplicados al software. De esta manera la Ingeniería de Software es analizada con base a normas internacionales de calidad y en mejores prácticas de la industria del software. En esta línea son contemplados, En esta línea son contemplados, por lo tanto, las diferentes áreas de proceso que se relacionan con el desarrollo de software, metodologías para el desarrollo de software, así como mejoras en entornos de desarrollo y los equipos de trabajo como elementos clave para el desarrollo de software. Por lo tanto, como temas de especialización: gestión de procesos de software, gestión de proyectos, aseguramiento de la información, optimización de

procesos (estocástica), experiencia de usuario, diseño de interfaces, arquitectura de software, sistemas distribuidos y/o concurrentes, dirección de equipos.

La línea de investigación de Tendencias y Aplicaciones en Ingeniería de Software cuenta actualmente con 3 profesores de tiempo completo adscritos a la institución sede CIMAT Zacatecas, 2 profesores de tiempo parcial, 2 profesores asociados de CIMAT unidad Aguascalientes. Entre ellos, seis cuentan con el título de doctorado y son miembros del Sistema Nacional de Investigadores SNI, cuatro en nivel 1.
Los perfiles del personal docente asociados a esta línea incluyen técnicos académicos expertos en ciencias de datos, optimización de procesos industriales, control estadístico de procesos y en pensamiento sistémico.

- **Aplicaciones de Inteligencia Artificial a la Ingeniería de Software.** Tiene el objetivo de estudiar, diseñar, desarrollar, implementar y evaluar iniciativas mixtas del comportamiento humano con la tecnología, y cómo estas pueden resolver desafíos en la investigación en la industria, educación y asistencia médica. Adicionalmente, se estudian las prácticas, técnicas y tecnologías que se emplean en el almacenamiento, procesamiento y análisis de pequeños y grandes volúmenes de datos, así como con el manejo de los flujos de datos derivados de las aplicaciones de internet de las cosas. Los productos de investigación de esta línea están orientados a la generación de conocimiento y a la solución de problemas industriales, gubernamentales y de negocios aplicando ciencia de datos, así como cómputo tradicional, concurrente y paralelo. Para ello, se ofrecen las siguientes asignaturas de especialización: aprendizaje máquina, aprendizaje profundo, aprendizaje por refuerzo, soft computing, sistemas inteligentes, procesamiento de lenguaje natural,cómputo evolutivo, sistemas expertos en ingeniería del software, desarrollo de aplicaciones multiplataforma.

  La línea de investigación Aplicaciones de Inteligencia Artificial a la Ingeniería de Software, cuenta actualmente con 3 profesores de tiempo completo adscritos a la institución sede CIMAT Zacatecas, 1 profesor de tiempo parcial, 1 profesor asociado de CIMAT unidad Aguascalientes. Además de 11 profesores asociados procedentes de otras unidades de CIMAT. De ahí, tres cuentan con el grado de doctor y son miembros del SNI en el nivel 1
  Los perfiles del personal docente asociados a esta línea incluyen técnicos académicos expertos en diseño y análisis de muestreo, diseño y análisis de experimentos, análisis de datos, probabilidad, métodos multivariados

CURSO PROPEDÉUTICO

Se considera la aplicación de un curso propedéutico con duración de una semana como parte de los criterios obligatorios a evaluar en el proceso de admisión. Serán invitados a participar del curso propedéutico aquellos candidatos a la maestría que a criterio del Comité Académico del Posgrado (CAP) hayan aprobado de manera satisfactoria el examen de admisión y cumplan con el perfil y demás requisitos necesarios para ingresar al posgrado. Este consta de tres módulos, a proponer por los miembros del núcleo académico, bajo aprobación del CAP. Se contempla un módulo por cada línea de investigación, más un módulo de fundamentos matemáticos o de programación. Dicho curso propedéutico no pretende ser un curso remedial, sino una herramienta para evaluar la capacidad de aprendizaje de los estudiantes

PERFIL DE INGRESO

Para ingresar a la Maestría en Ingeniería de Software es necesario: haber cursado una Licenciatura o ingeniería en Ciencias de la Computación, Ingeniería de Software, en Informática o afines; tener conocimientos de programación, estadística básica y matemáticas discretas; contar con experiencia demostrable en el desarrollo de software y tener capacidad de lectura y compresión de material técnico en inglés.

ADMINISTRACIÓN Y OPERATIVIDAD DEL PLAN DE ESTUDIOS

Se considera un plan de estudios flexible, para obtener el grado se requiere obtener un mínimo de 96 créditos que incluyen cuatro asignaturas obligatorias, dos materias electivas de Ingeniería de Software, 2 materias electivas de Ciencias de la Computación, un Taller con la Industria, un Proyecto con la Industria y al menos un Seminario de Titulación I.

La duración mínima de la maestría es de 1.5 años (que incluye un curso de verano) y un máximo de 2 años que incluye la acreditación del Seminario de Titulación II. Cada semestre a partir del tercero, el alumno debe acreditar un curso de Seminario de Titulación. Para ambas modalidades, el alumno tendrá una carga máxima de 36 créditos por semestre y de 42 créditos en el semestre que incluye un verano.

Las asignaturas **obligatorias** son:

1. Ingeniería de Software I
2. Programación y Algoritmos
3. Estadística y Diseño Experimental
4. Ingeniería de Software II
5. Taller con la Industria
6. Seminario de Titulación I
7. Proyecto con la Industria
8. Seminario de Titulación II

La selección de cursos electivos la efectuará el estudiante bajo la supervisión del tutor o asesor, a partir de la lista de cursos electivos avalados por el Comité Académico del Posgrado (CAP) y notificados cada semestre. El CAP de la maestría determinará cada semestre los cursos electivos que se ofrecerán, conciliando la calidad educativa, las necesidades del programa y la carga académica de los profesores. El CAP dará aviso oportuno al núcleo académico sobre las materias electivas que se vayan a ofrecer cada semestre (para su publicación y programación en los sistemas correspondientes). Las materias electivas podrán validarse en un periodo distinto al contemplado originalmente en el mapa curricular bajo la supervisión del asesor o tutor, con el visto bueno del CAP y siguiendo el procedimiento establecido por la Coordinación de Formación Académica, para exámenes de equivalencia.

Para lograr los objetivos antes mencionados, el programa ofrece un amplio abanico de materias electivas, tanto teóricas como prácticas con un enfoque multidisciplinario, donde el primer semestre el alumno llevará en su mayoría cursos de tronco común para sentar bases sólidas en Ingeniería de Software. Después, con ayuda de un tutor o asesor, se definirá el área de conocimiento específica dentro de la cual enmarcará su trabajo para titulación, en una de las dos líneas de aplicación y generación del conocimiento ofrecidas: (a) Tendencias y Aplicaciones en Ingeniería de Software y (b) Aplicaciones de Inteligencia Artificial a la Ingeniería de Software.

Opciones de Titulación

Se puede obtener el grado de Maestro en Ingeniería de Software con alguna de las siguientes opciones de titulación:

- Tesis. Es un trabajo de investigación inédito, que tendrá como objetivo presentar nuevos conocimientos, métodos, o interpretaciones sobre cualquier aspecto de la ingeniería de software.
- Informe de actividad profesional. En este informe hace referencia a un proyecto con la industria, en donde el candidato describe el proyecto que ha realizado, describe el problema, su contexto y la solución implementada para resolverlo.

- Trabajo Terminal: En este informe, el candidato describe el trabajo profesional realizado en colaboración con una organización, empresa, gobierno o sociedad. El enfoque aceptado para un proyecto terminal comprende el diseño y desarrollo de una propuesta de intervención y mejora de algún aspecto o proceso relacionado con la Ingeniería de Software. Se pueden contemplar como trabajo terminal: Tesina, Estudios de Caso, Memorias (publicaciones)..

SUSTENTO TEÓRICO DEL MODELO CURRICULAR

El modelo curricular sigue las guías del modelo curricular presentado en el documento "MSIS 2016: Global Competency Model for Graduate Degree Programs in Information Systems", presentado como un esfuerzo conjunto por ACM (Association for Computing Machinery) y la AIS (Association for Information Systems). En específico, se consideraron las competencias del perfil de **Desarrollo e Implementación de Sistemas.** Este perfil cubre el diseño de sistemas y servicios de información, proponiendo diseños que consideren cómo los humanos interactúan y experimentan con artefactos de TI. También incluye competencias relacionadas con la implementación y despliegue de sistemas para uso organizacional.

JUSTIFICACIÓN DE LA PROPUESTA CURRICULAR

No aplica

PROPUESTA DE EVALUACIÓN PERIÓDICA DEL PLAN DE ESTUDIOS

La evaluación y actualización del plan de estudios de la MIS tiene como objetivo la mejora continua y el aseguramiento de la calidad de tal manera que se garantice que el plan de estudios:

1. Responda a las tendencias actuales en el área de Ingeniería de Software.
2. Facilite la generación y aplicación del conocimiento de las LGACs.
3. Genere soluciones a diferentes problemas de los sectores organizaciones, gobierno, empresas y sociedad en general con un enfoque colaborativo, de retribución social e inter-, multi- y transdisciplinario .
4. Garantice la competencia y el desempeño profesional del egresado.

Justificación. Las modificaciones propuestas al plan de estudios de la MIS deberán ser justificadas en un documento y con fundamentos en resultados cualitativos y cuantitativos de la autoevaluación de la MIS, recomendaciones de organismos acreditadores (como aquellas emanadas de las evaluación PNPC), y el análisis de las tendencias en organizaciones, gobierno, empresas y sociedad en general.

Periodicidad. El plan de estudios de MIS, se evaluará periódicamente, con una frecuencia de tres años. Por lo que se deberá evaluar finalizar al menos una vez al año.

Para la evaluación del plan de estudio se propone la siguiente metodología:

1. Se obtiene la información de todas las partes interesadas: evaluaciones obtenidas por los estudiantes en cada asignatura, medición del desempeño del egresado, recopilación de necesidades de un comité integrado por un representante de la industria, gobierno y academia.
2. El núcleo académico básico se reúne para analizar el plan de estudios frente a los resultados de la información recopilada en el paso 1, con la finalidad de analizar la pertinencia del contenido temático de cada materia.
3. Se presentan las actualizaciones identificadas para el plan de estudio al Cuerpo Académico de Posgrado (CAP), quien será encargado de avalar o rechazar la actualización sugerida considerando los objetivos del plan de la MIS. En el caso de proceder, se actualiza el plan de estudios.
4. En caso de que procedan las modificaciones del plan de estudios, estas serán debidamente publicadas y notificadas a los estudiantes, personal e instancias

correspondientes, entrando en vigor para las generaciones subsecuentes a la notificación de las mismas.

Dr. Víctor Manuel Rivero Mercado
Director General

NOMBRE Y CARGO DE LA PERSONA FACULTADA PARA AUTORIZAR EL PLAN DE ESTUDIOS

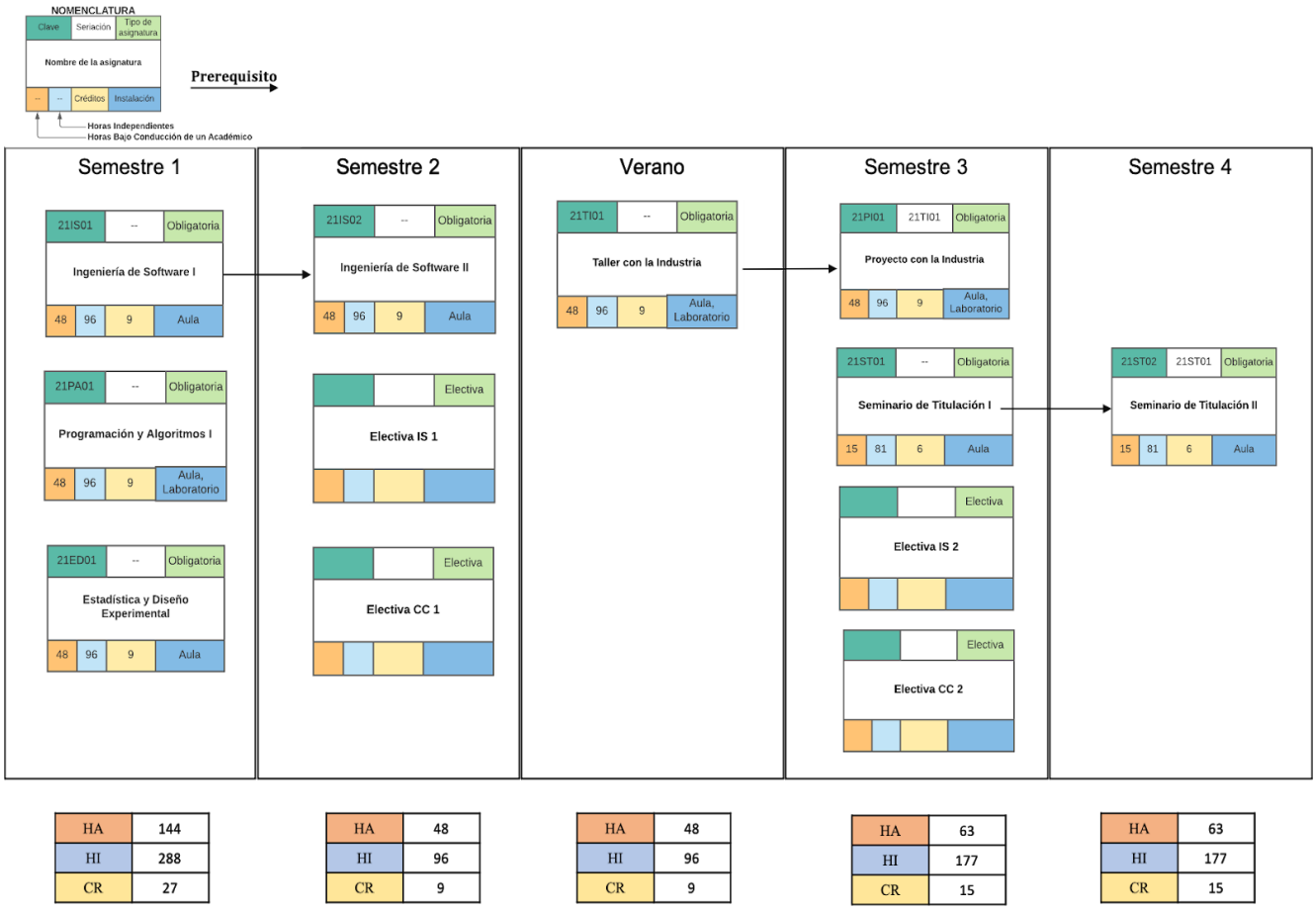
MAPA CURRICULAR

Centro de Investigación en Matemáticas A.C.- Unidad Zacatecas

Plan de estudios modalidad escolarizada

Maestría en Ingeniería de Software

PARA 102 CRÉDITOS



HA	144
HI	288
CR	27

HA	48
HI	96
CR	9

HA	48
HI	96
CR	9

HA	63
HI	177
CR	15

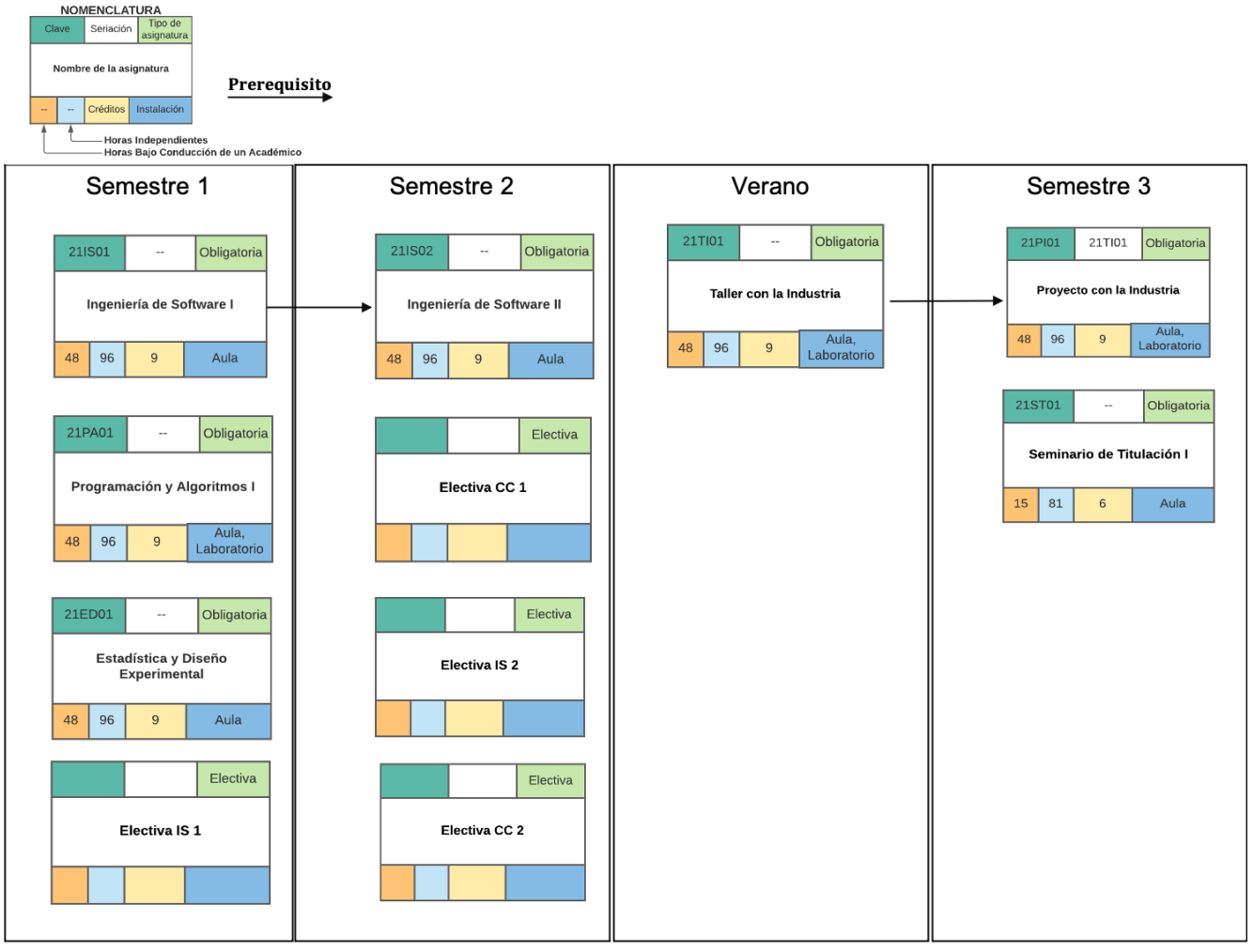
HA	63
HI	177
CR	15

MATERIAS ELECTIVAS							
CICLO	ASIGNATURA O UNIDAD DE APRENDIZAJE	CLAVE	SERIACIÓN	CON ACADEMICO	INDEPENDIENTES	CRÉDITOS	INSTALACIONES
2 o 3	Tópicos selectos de Ingeniería de Software	21TIS01	21IS02	48	96	9	Aula
2	Gestión de Proyectos de Software	21GP01		48	96	9	Aula
2 o 3	Gestión del Proceso de Software	21GS01		48	96	9	Aula
2 o 3	Dirección de Equipos de Desarrollo de Software	21DE01		48	96	9	Aula
2 o 3	Arquitectura de Software	21AS01	21IS01	48	96	9	Aula
2 o 3	Optimización de procesos (optimización estocástica)	21OP01	21ED01	48	96	9	Aula
2 o 3	Diseño Interfaces y Experiencia del Usuario	21DU01		48	96	9	Aula, Laboratorio
2 o 3	Cómputo Paralelo	21CP01	21PA01	48	96	9	Aula
2 o 3	Aprendizaje Máquina I	21AM01	21PA01	48	96	9	Aula, Laboratorio
2 o 3	Programación y Algoritmos II	21PA02	21PA01	48	96	9	Aula, Laboratorio
2 o 3	Cómputo Evolutivo	21CE01	21PA01	48	96	9	Aula, Laboratorio
2 o 3	Procesamiento de Lenguaje Natural	21PL01	21PA01	48	96	9	Aula, Laboratorio
2 o 3	Optimización	21OP02		48	96	9	Aula
2 o 3	Tópicos Selectos de Ciencias en Computación	21TC01		48	96	9	Aula
2 o 3	Ciencia de Datos	21CD01		48	96	9	Aula

REQUERIMIENTOS MÍNIMOS PARA LAS ASIGNATURAS ELECTIVAS	
HORAS BAJO CONDUCCIÓN DE UN ACADÉMICO	192
CRÉDITOS	36

TOTAL DE ASIGNATURAS QUE INTEGRAN EL PLAN DE ESTUDIOS			
TIPO DE ASIGNATURA	HORAS BAJO CONDUCCIÓN DE UN ACADÉMICO	HORAS INDEPENDIENTES	CRÉDITOS
OBLIGATORIAS	318	738	66
ELECTIVAS	192	384	36
SUMAS TOTALES	510	1122	102

PARA 96 CRÉDITOS



HA	144
HI	288
CR	27

HA	48
HI	96
CR	9

HA	48
HI	96
CR	9

HA	63
HI	177
CR	15

MATERIAS ELECTIVAS							
CICLO	ASIGNATURA O UNIDAD DE APRENDIZAJE	CLAVE	SERIACIÓN	CON ACADEMICO	INDEPENDIENTES	CRÉDITOS	INSTALACIONES
2 o 3	Tópicos selectos de Ingeniería de Software	21TIS01	21IS02	48	96	9	Aula
2	Gestión de Proyectos de Software	21GP01		48	96	9	Aula
2 o 3	Gestión del Proceso de Software	21GS01		48	96	9	Aula
2 o 3	Dirección de Equipos de Desarrollo de Software	21DE01		48	96	9	Aula
2 o 3	Arquitectura de Software	21AS01	21IS01	48	96	9	Aula
2 o 3	Optimización de procesos (optimización estocástica)	21OP01	21ED01	48	96	9	Aula
2 o 3	Diseño Interfaces y Experiencia del Usuario	21DU01		48	96	9	Aula, Laboratorio
2 o 3	Cómputo Paralelo	21CP01	21PA01	48	96	9	Aula
2 o 3	Aprendizaje Máquina I	21AM01	21PA01	48	96	9	Aula, Laboratorio
2 o 3	Programación y Algoritmos II	21PA02	21PA01	48	96	9	Aula, Laboratorio
2 o 3	Cómputo Evolutivo	21CE01	21PA01	48	96	9	Aula, Laboratorio
2 o 3	Procesamiento de Lenguaje Natural	21PL01	21PA01	48	96	9	Aula, Laboratorio
2 o 3	Optimización	21OP02		48	96	9	Aula
2 o 3	Tópicos Selectos de Ciencias en Computación	21TC01		48	96	9	Aula
2 o 3	Ciencia de Datos	21CD01		48	96	9	Aula

REQUERIMIENTOS MÍNIMOS PARA LAS ASIGNATURAS ELECTIVAS	
HORAS BAJO CONDUCCIÓN DE UN ACADÉMICO	192
CRÉDITOS	36

TOTAL DE ASIGNATURAS QUE INTEGRAN EL PLAN DE ESTUDIOS			
TIPO DE ASIGNATURA	HORAS BAJO CONDUCCIÓN DE UN ACADÉMICO	HORAS INDEPENDIENTES	CRÉDITOS
OBLIGATORIAS	303	657	60
ELECTIVAS	192	384	36
SUMAS TOTALES	492		96

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Ingeniería de Software I

SEMESTRE 1
CICLO ESCOLAR
21IS01
CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

Al finalizar este curso el estudiante será capaz de describir los conceptos básicos de Ingeniería de Software, así como las prácticas más reconocidas de desarrollo de software; analizar técnicas e identificar diferencias de modelos y metodologías de desarrollo de software actuales, para elegir y combinar las prácticas más eficientes según el contexto y naturaleza de un proyecto e implementar una arquitectura con la estructuración de un sistema para satisfacer los requisitos de los clientes y otros involucrados, en especial los requisitos de atributos de calidad.

CONTENIDO TEMÁTICO

I. Introducción a la Ingeniería de Software
- A. Importancia del Software
- B. Características y tipos de software
- C. La crisis del software
- D. Definición e Importancia de la Ingeniería de Software

II. Ingeniería de requisitos de software
- A. Lo esencial de los requisitos de software
- B. Ingeniería de requisitos
- C. Licitación de requisitos

III. Diseño y modelado de los requisitos
- A. Requisitos en métodos ágiles y tradicionales
- B. Requisitos de negocio, de usuario y funcionales
- C. Especificación de requisitos de software (estándar IEEE 830-1998)
- D. Modelado de requisitos en Lenguaje Unificado de Modelado
- E. Requisitos no funcionales

IV. Introducción a la Arquitectura de Software

A. Definición de Arquitectura de Software
B. Objetivos de Negocio
C. Proceso de Desarrollo de Arquitectura

V. Modelos de desarrollo de software

A. Modelo de cascada
B. Modelo Iterativo
C. Modelo en espiral
D. Modelo de desarrollo de componentes
E. Modelos emergentes

BIBLIOGRAFÍA

1. WIEGERS, KARL E. Software Requirements Business Modeling with UML: Business Patterns at Work Microsoft Press 2003
2. McConnell, Rapid Development, S Microsoft Press 1999
3. Mario Barbacci, Mark H. Klein, Thomas A. Longstaff. Charles B. Weinstock.Quality Attributtes CMU/SEI-95-TR-021. ESC-TR-95-021 Microsoft Press 1995
4. IEEE, IEEE Recommended Practice for Software Requirements Specifications. ISBN 0-7381-0332-2 IEEE 1998
5. Dean Leffingwell. Agile software requirements: Lean requirements practices for teams, programs, and enterprise Addison-Wesley 2011
6. Roger S. Pressman. Ingeniería del software: un enfoque práctico. 6ta Edición Traducción de Víctor Campos Olguín, Javier Villegas Quezada. (7th. Edition). McGraw Hill. 2010
7. Sommerville, I. Software engineering (7ª ed.) Pearson 2004
8. Booch G. El lenguaje Unificado de Modelado, UML 2.0, Guia de Usuario. 1ª.
9. Edición Pearson ADDISON WESLEY 2006
10. Hans Van Vliet Wiley. Software Engineering. Principles and Practice (3a ed) 2007
11. Len Bass, Paul Clements, and Rick Kazman, Software Architecture in Practice Addison Wesley 2012
12. Anthony J. Lattanze. Architecting Software Intensive Systems: A Practitioner's Guide Taylor and Francis/Auerbach 2008
13. Richard N. Taylor, Nenad Medvidovic, and Eric MSoftware Architecture: Foundations, Theory and Practice Addison Wesley 2007
14. Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little and Robert Nord. Documenting Software Architectures: Views and Beyond, Addison Wesley 2011
15. Tavish Armstrong. The Performance of Open Lulu.com, 2013
16. Amy Brown and Greg WilsonThe Architecture Of Open Source Applications Lulu.com Vol I, 2011 Vol II, 2012
17. Miles, R., & Hamilton, K. (2006). Learning UML 2.0. " O'Reilly Media, Inc.".
18. Fowler, M. (2004). UML distilled: a brief guide to the standard object modeling language. Addison-Wesley Professional.

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña.

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Exposiciones por parte del profesor
Exposiciones por parte del alumno
Realización de tareas por parte del alumno
Realización de lecturas por parte del alumno
Desarrollo de prácticas
Desarrollo de un proyecto integrador
Exámenes.

CRITERIOS DE EVALUACIÓN

Tareas	10%
Lecturas	10%
Dos exámenes parciales	20%
Proyecto final	60%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Programación y Algoritmos I

SEMESTRE 1

CICLO ESCOLAR

21PA01

CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

El curso tiene como objetivo dar a los estudiantes sólidos fundamentos en programación, y en particular utilizando el lenguaje C y C++.

CONTENIDO TEMÁTICO

I. Introducción

A. Estructura y sintaxis de un programa en C; archivos de cabecera; palabras reservadas; tipos de datos; variables, declaración; alcance de las variables, locales y globales; tipos de expresiones, lvalue, rvalue; operadores aritméticos, lógicos, relacionales y misceláneos; printf y scanf; compilación; redirección de la entrada y salida estándar; errores comunes; tiempo de compilación y tiempo de ejecución; argv y argc.

B. Condiciones; if, else; for; do while; break y continue; while.

II. Funciones

A. Declaración y definición de funciones; partes de una función; paso de argumentos por copia; archivos cabecera propios; compilación con varios archivos de declaración y definición de funciones.

III. Memoria y arreglos:

A. Heap, stack, code, globals; notas sobre eficiencia; arreglos; direcciones de memoria; arreglos en funciones; arreglos en ciclos; arreglos multidimensionales.

IV. Lectura y escritura de archivos:

A. Apertura y modos de apertura; lectura; escritura; archivos binarios.

V. Apuntadores y memoria dinámica:

A. Apuntadores; operadores *, & y []; Apuntador NULL; memoria dinámica; malloc y free; arreglos con memoria dinámica; aritmética de apuntadores; apuntadores y archivos de texto.

VI. Cadenas de caracteres:

A. Caracteres especiales y secuencias de escape; manejo de strings; string.h.

VII. Estructuras, uniones:

A. Tipos enum; typedef struct; memoria en la estructura; funciones que reciben y devuelven estructuras; typedef; estructuras y apuntadores; aritmética de apuntadores en estructuras; unión; estructuras y uniones; arreglos de estructuras.

VIII. Clases de almacenamiento:

A. auto, register, static, extern.

IX. Apuntadores a funciones:

A. Sintaxis; funciones que reciben y devuelven apuntadores a funciones; typedef.

X. Funciones con número variable de argumentos:

A. va_list; va_start,va_end.

XI. Operaciones con bits:

A. Operadores; generadores de pseudo-aleatorios**.**

XII. Preprocesador:

A. Macros; el preprocesador de C; directivas en gcc; macros predefinidas; concatenación; definición condicional de macros; compilación condicional.

XIII. Algoritmos y complejidad:

A. Invariante de ciclo; orden; merge sort (recursión); insertion sort.

XIV. Estructuras de datos y recursividad:

A. Listas ligadas; listas doblemente ligadas; algoritmos recursivos para (manejo de memoria en) listas ligadas; algoritmo de Shunting-Yard; árboles binarios.

XV. Introducción a la programación orientada a objetos:

A. Conceptos OOP; objeto; abstracción; encapsulamiento; herencia; polimorfismo; sobrecarga de funciones y operadores; compilación con g++; tipos de datos; namespace.

XVI. Clases:

A. class; datos y funciones miembro; constructor; modificadores de acceso; inicializaciones; sobrecarga del constructor; constructor de copia; destructor; funciones friend; funciones inline; apuntador this; miembros static.

XVII. Herencia:

A. Clases base y heredada.

XVIII. Sobrecarga de funciones y operadores:

A. Sobrecarga de funciones; Sobrecarga de operadores; Operadores sobrecargables.

XIX. Polimorfismo, Abstracción, Encapsulamiento, Interfaces:

A. Polimorfismo de funciones en clases derivadas; funciones virtuales.
B. Ejemplos de abstracción mediante etiquetas de acceso.
C. Ejemplos de encapsulamiento.
D. Clases abstractas; funciones puramente virtuales.

XX. Temas misceláneos:

A. Makefiles; multithreading.

BIBLIOGRAFÍA

1. R. Sedgewick. Algorithms in C++. Addison Wesley.
2. B.Preiss. Data Structures and Algorithms with Object Oriented Design Patterns in (C++, Java). http://www.brpreiss.com/
3. C.Cormen, C.Leiserson, R.Rivest y C.Stein. Introduction to Algorithms. MIT Press.
4. J.Kleinberg y E.Tardos. Algorithm Design. Addison Wesley.
5. D.Knuth. The Art of Computer Programming. Vol.1 Fundamental Algorithms, Vol.3 Sorting and Searching. Addison-Wesley.
6. Stallman, Richard M., and Zachary Weinberg. "The C preprocessor." Free Software Foundation (1987).
7. Kernighan BW, Ritchie DM. The C programming language. Englewood Cliffs: Prentice-Hall; 1988 Mar 22
8. Goldberg, David. "What every computer scientist should know about floating-point arithmetic." ACM Computing Surveys (CSUR) 23.1 (1991): 5-48.
9. Drepper, Ulrich. "What every programmer should know about memory." Red Hat, Inc 11 (2007): 2007.
10. Mark Allen Weis, Efficient C Programming A practical approach, Prentice Hall
11. Pitt-Francis, Joe, and Jonathan Whiteley. Guide to scientific computing in C++. Springer Science & Business Media, 2012.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Cursos presenciales
Resolución de ejercicios
Exámenes

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Desarrollo de proyectos
Desarrollo de software
Preparación de presentaciones
Desarrollo de informes
Lectura de publicaciones especializadas

CRITERIOS DE EVALUACIÓN

Prácticas y Tareas	40%
Proyectos (10% el primero, 20% el final).	30%

Exámenes	30%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Equipo de cómputo con herramientas y compiladores para C y C++.
Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Estadística y Diseño Experimental

SEMESTRE 1

CICLO ESCOLAR

21ED01

CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

Al finalizar este curso el estudiante será capaz de desarrollar su estado del arte, diseñar experimentos, de comprender y aplicar las herramientas de desarrollo de software (e.g. lenguaje R) y metodologías básicas de la estadística y la probabilidad.

En específico los estudiantes podrán: desarrollar revisiones, diseñar experimentos que involucren sujetos de estudio, presentar resultados descriptivos en situaciones diversas según el comportamiento de los datos, programar ecuaciones para análisis de conjuntos de datos, realizar validaciones de hipótesis con pruebas paramétricas y no paramétricas según sea el caso, realizar distintos tipos de análisis de correlaciones, realizar representaciones gráficas de resultados, y particionar datos en grupos.

CONTENIDO TEMÁTICO

I. Metodología de investigación
   A. Lectura y comprensión de artículos
   B. Definición de un problema
   C. Definición de un tema de investigación
   D. Revisión literaria
   E. Revisión sistemática

II. Estrategias de investigación
   A. Experimentos
   B. Cuasi-experimentos
   C. Encuestas
   D. Casos de estudio
   E. Observación

III. Método científico

A. Identificar problema
B. Formular hipótesis
C. Conducción de un estudio piloto
D. Método
E. Participantes
F. Procedimiento
G. Resultados
H. Discusiones

IV. Estadística descriptiva básica

V. Programación de ecuaciones para conjuntos de datos

A. Programación de ecuaciones
B. Almacenamiento, recuperación y cambios de valores de datos

VI. Pruebas de hipótesis

A. Pruebas paramétricas
B. Pruebas no paramétricas

VII. Exploración y representación visual de datos

A. Estructuras de datos para graficar
B. Gráficos de barras, líneas y dispersión
C. Distribuciones de datos con histogramas, curvas de densidad, cajas, y otros
D. Interpretación de gráficos

VIII. Análisis estadístico

A. Análisis de correlaciones
B. Análisis de grupos de datos
C. Análisis de series de tiempo

ACTIVIDADES DE APRENDIZAJE
BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases presenciales donde se considera la asistencia y ejercicios.

ACTIVIDADES DE APRENDIZAJE
INDEPENDIENTES

Lecturas, preparación de presentaciones y proyecto final con integración de temas principales vistos en el temario.

CRITERIOS DE EVALUACIÓN

Asistencia	5%
Presentaciones	25%
Ejercicios	30%
Proyecto final	40%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Para el proceso de enseñanza-aprendizaje de la asignatura, se utilizarán espacios de tipo aula, software para programación con el lenguaje estadístico R, y en caso necesario plataforma para programación en el lenguaje Python.

BIBLIOGRAFÍA

1. Grolemund, G. (2014). Hands-on programming with r: write your own functions and simulations. " O'Reilly Media, Inc.".
2. Robert, I. (2016). Kabacoff R in Action: Data Analysis and Graphics with R.
3. Chang, W. (2018). R graphics cookbook: practical recipes for visualizing data. O'Reilly Media.
4. Kirk, R. E. (2013). Experimental design: Procedures for the behavioral sciences. Thousand Oaks, CA: SAGE Publications, Inc. doi: 10.4135/9781483384733
5. Vercruyssen, M., & Hendrick, H. W. (2011). Behavioral research and analysis: an introduction to statistics within the context of experimental design. CRC Press.
6. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). Experimentation in software engineering. Springer Science & Business Media.
7. Boehm, B., Rombach, H. D., & Zelkowitz, M. V. (Eds.). (2005). Foundations of empirical software engineering: the legacy of Victor R. Basili. Springer Science & Business Media.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Ingeniería de Software II

SEMESTRE 2
CICLO ESCOLAR
21IS02
CLAVE DE LA ASIGNATURA
21IS01
PRERREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

El alumno conocerá y aplicará las reglas generales de los modelos prescriptivos de desarrollo de software, los marcos de referencia de calidad y las técnicas de medición de software más utilizados. Además, de conocer y analizar las metodologías de desarrollo de software aplicadas en la industria, siendo capaz de aplicar los conocimientos en diversos proyectos de software que se puedan presentar.

CONTENIDO TEMÁTICO

I. **Las 4 P's de la Ingeniería de software**
   A. Procesos
   B. Proyectos
   C. Productos
   D. Personas

II. **Fundamentos de metodologías de desarrollo**
   A. Producto de Software ventajas y problemas.
   B. Sistema de Información ventajas y problemas

II. **Metodologías de desarrollo de software**
   A. Introducción a las Metodologías
   B. Metodologías tradicionales
   C. Metodologías ágiles
   D. Análisis de semejanzas y diferencias entre metodologías tradicionales y ágiles

E. Metodologías más empleadas

III. Pruebas del software

A. Introducción a la validación del software
B. Pruebas del software
C. Plan de validación del software
D. Integración de verificación y validación al proceso de desarrollo de software
E. El papel de los estándares en el proceso de verificación y validación del software

IV. Herramientas utilizadas en la calidad del software

A. Herramientas fundamentales de la calidad
B. Herramientas de pruebas unitarias y de aceptación
C. Herramientas para integración continua
D. Herramientas para entrega continua

BIBLIOGRAFÍA

1. Roger S. Pressman. Ingeniería del software: un enfoque práctico. 6ta Edición Traducción de Víctor Campos Olguín, Javier Villegas Quezada. (7th. Edition). McGraw Hill. 2010
2. Sommerville, I. Software engineering (7ª ed.) Pearson 2004.
3. Shari Lawrence Pfleeger. Ingeniería de Software Teoría y Práctica Prentice Hall 2002
4. Mary Beth Chrissis, Mike Konrad, Sandra Shrum. CMMI for Development: Guidelines for Process Integration and Product Improvement (3rd Edition) (SEI Series in Software Engineering) Addison-Wesley Professional; 2011
5. Humphrey W. A Discipline for Software Engineering. AddisonWesley 1995
6. Gonzalo Cuevas Agustín, Gestión del Proceso Software Editorial Universitaria, Ramón Areces. 2002
7. Viega John & McGraw GaryBuilding Secure Software: How to Avoid Security Problems the Right Way (paperback). Addison-Wesley Professional Computing Series, 2011
8. Blankenship J., Bussa M. & Millett S. Pro Agile .NET development with Scrum. Apress. ISBN 978-1-4302-3534-7 (eBook) pps. 372, 2011
9. Chow and D.-B. Cao. A survey study of critical success factors in agile software projects, 2008
10. D. Bustard, G. Wilkie, and D. Greer. Towards optimal software engineering: learning from agile practice. Innovations in Systems and Software Engineering, 9(3):191–200, 2013.
11. ISO/IEC. ISO/IEC TR 29110-5-1-2:2011 - Software engineering – Lifecycle profiles for very small entities (VSEs) – Part 5-1-2: Management and engineering guide – Generic profile group: Basic profile. Freely available from ISO at: http://standards.iso.org/ittf/PubliclyAvailableStandards/c060389_ISO_IEC_TR_29110-5-1-1_2012(E).zip, 2011.
12. Certificación NMX-I-059 Moprosoft - NYCE Mexico. NYCE. https://www.nyce.org.mx/certificacion-nmx-i-059-moprosoft/. 2008
13. Humble J, Farley D. Continuous Delivery, Reliable Software Releases through Build, Test, and Deployment Automation. Pearson Education, 2010.

14. Duvall PM, Matyas & S, Glover A. Continuous Integration, Improving Software Quality and Reducing Risk. Pearson Education, 2007

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña.

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Exposiciones por parte del profesor
Exposiciones por parte del alumno
Realización de tareas por parte del alumno
Realización de lecturas por parte del alumno
Desarrollo de prácticas
Desarrollo de un proyecto integrador

CRITERIOS DE EVALUACIÓN

Tareas	10%
Lecturas	10%
Exposiciones	20%
Proyecto final	60%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Taller con la Industria

Verano
CICLO ESCOLAR
21TI01
CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

Al finalizar este curso el estudiante será capaz de identificar la problemática industrial para aplicar conocimientos, habilidades, destrezas y técnicas avanzadas de Ingeniería de Software para resolver algún problema en específico o una necesidad de la industria del software o de la investigación que esté realizando el estudiante. Durante el curso, el alumno aprenderá a definir un proyecto aplicado a la industria, planificar una serie de actividades interrelacionadas, plantear objetivo de acuerdo al tipo de proyecto y los recursos disponibles. Se espera que al finalizar el curso, los alumnos hayan diseñado un producto, servicio, proceso o modelo que agregue valor tecnológico o científico, además se espera que hayan incorporado las técnicas aprendidas durante su estancia en el posgrado.

CONTENIDO TEMÁTICO

I. **Taller inicial con la industria (problemas)**
II. **Planeación del proyecto**
III. **Diseño del proyecto**
IV. **Reporte del proyecto**

BIBLIOGRAFÍA

1. Roel J. Wieringa. Design Science Methodology for Information Systems and Software Engineering. Springer-Verlag Berlin Heidelberg. XV, 332. DOI. 10.1007/978-3-662-43839-8. 2014.
2. Hevner, A., March, S., Park, J., & Ram, S. Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75–105. https://doi.org/10.2307/25148625. 2004.

3. David J Anderson. Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press 2010
4. Herki Kniberg. Scrum from the Trenches InfoQ 2007
5. Project Management Institute. Project Managment Body Of Knowledge. Fifth edition. Project Management Institute 2013.

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Trabajos periódicos
Reporte de proyecto final
Presentación del proyecto final

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Lecturas de artículos y capítulos de libros.
Resolución de problemas.
Proyecto final.

CRITERIOS DE EVALUACIÓN

Trabajos periódicos	40%
Reporte de proyecto final	50%
Presentación del proyecto final	10%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Seminario de Titulación I

SEMESTRE 3
CICLO ESCOLAR
21ST01
CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

Al finalizar este curso el alumno será capaz de abordar su trabajos de titulación ya sea por tesis, informe de proyecto con la industria o trabajo terminal (proyecto terminal, tesina, estudios de caso, memorias (publicaciones)) de manera satisfactoria incrementando su eficiencia y productividad. Además de cubrir aspectos relacionados con la forma de llevar a cabo su trabajo para titulación de modo que revierta en bien de la sociedad y de la comunidad científica, de ser posible con buenos resultados ya sea por publicaciones y/o productos de propiedad intelectual.

CONTENIDO TEMÁTICO

V. Introducción y motivación

A. Reflexiones sobre la investigación
B. Investigación en ingeniería del software: La vida más allá de la programación
C. Áreas y tópicos de investigación en la ingeniería del software

VI. Cómo centrar un tema de investigación

A. Consideraciones y recomendaciones generales para centrar un tema de investigación
B. Guía práctica para definir un tema de investigación
C. Compromisos de tutores, directores y alumnos una vez que se define un tema de investigación

VII. Búsqueda de información y revisión crítica de trabajos científicos

A. Bases de datos y motores de búsqueda especializados
B. Guía práctica para la revisión crítica de trabajos científicos
C. Proceso de revisión literaria (Mapeo sistemático y revisión sistemática de literatura).

VIII. Herramientas tecnológicas de apoyo a la investigación

IX. Propiedad Intelectual

X. Técnicas para redactar y estructurar un informe técnico y una tesis

A. Técnicas y estructura para redactar un informe técnico

B. Método general para estructurar y escribir una tesis

BIBLIOGRAFÍA

1. Ranjit Kumar. Research Methodology: A Step-by-Step Guide for Beginners, 4th edition. SAGE Publications. 2014
2. John M. Swales & Christine Feak. Academic Writing for Graduate Students, 3rd. Edition: Essential Tasks and Skills. University of Michigan Press. 2012
3. Kate L. Turabian. A Manual for Writers of Research Papers,Theses, and Dissertations, Eighth Edition: Chicago Style for Students and Researchers (Chicago Guides to Writing, Editing, and Publishing). University Of Chicago Press.2013
4. Kitchenham, B. et Charters, S. (2007). Guidelines for performing Systematic Literature Reviews in Software Engineering (no EBSE 2007-001). Keele University and Durham University Joint Report. Repéré à http://www.dur.ac.uk/ebse/resources/Systematic-reviews-5-8.pdf
5. Petersen, K., Feldt R., Mujtaba, S., Mattsson, M. Systematic Mapping Studies in Software Engineering. In: 12th International Conference on Evaluation and Assessment in Software Engineering. (2008).

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor.
Exposiciones.
Resolución de problemas.

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Trabajos.
Revisiones de artículos.
Tema para titulación formalizado.

CRITERIOS DE EVALUACIÓN

Asistencia	10%
Trabajo sobre definición de tema para titulación	20%
Revisión y análisis crítico de artículo científico	10%
Elaboración de anteproyecto para titulación	60%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Proyecto con la Industria

SEMESTRE 4
CICLO ESCOLAR
21PI01
CLAVE DE LA ASIGNATURA
21TI01
PRERREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

Tiene por objetivo la aplicación de conocimientos, habilidades, destrezas y técnicas avanzadas de Ingeniería de Software para resolver algún problema en específico o una necesidad de la industria del software o de la investigación que esté realizando el estudiante. Durante el curso, el alumno aprenderá a definir un proyecto aplicado a la industria, planificar una serie de actividades interrelacionadas, plantear objetivo de acuerdo al tipo de proyecto y los recursos disponibles y llevar a cabo el proyecto durante un periodo definido. Se espera que al finalizar el curso, los alumnos hayan creado un producto, servicio, proceso o modelo que agregue valor tecnológico o científico, además se espera que hayan incorporado las técnicas aprendidas durante su estancia en el programa.

CONTENIDO TEMÁTICO

I. **Desarrollo y ejecución del proyecto**
II. **Pruebas y reporte del proyecto**
III. **Presentación y/o divulgación**

BIBLIOGRAFÍA

1. Roel J. Wieringa. Design Science Methodology for Information Systems and Software Engineering. Springer-Verlag Berlin Heidelberg. XV, 332. DOI. 10.1007/978-3-662-43839-8. 2014.
2. Hevner, A., March, S., Park, J., & Ram, S. Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75–105. https://doi.org/10.2307/25148625. 2004.
3. David J Anderson. Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press 2010
4. Herki Kniberg. Scrum from the Trenches InfoQ 2007
5. Project Management Institute. Project Managment Body Of Knowledge. Fifth edition. Project Management Institute 2013.

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Trabajos periódicos
Reporte de proyecto final
Presentación del proyecto final

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Lecturas de artículos y capítulos de libros.
Resolución de problemas.
Proyecto final.

CRITERIOS DE EVALUACIÓN

Trabajos periódicos	40%
Reporte de proyecto final	50%
Presentación del proyecto final	10%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.

Asesorías y aula virtual. Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Seminario de Titulación II

SEMESTRE 4
CICLO ESCOLAR
21ST02
CLAVE DE LA ASIGNATURA
21ST01
PRERREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

Al finalizar este curso el estudiante será capaz de abordar sus trabajos de titulación ya sea por tesis o trabajo terminal (proyecto terminal, informe de actividad profesional o tesinas) de manera satisfactoria incrementando su eficiencia y productividad, Además de cubrir aspectos relacionados con la forma de llevar a cabo su trabajo para titulación de modo que revierta en bien de la sociedad y de la comunidad científica, de ser posible con buenos resultados ya sea por publicaciones y/o productos de propiedad intelectual.

CONTENIDO TEMÁTICO

I. **Qué hace buena a una investigación en Ingeniería de Software**
II. **Métodos empíricos de Ingeniería de Software**
    A. Experimentos controlados
    B. Estudios de caso
    C. Encuesta de investigación
    D. Investigación en acción
III. **Recomendaciones generales para escribir artículos científicos y reportar resultados**
    A. Consejos prácticos para la escritura de artículos científicos, reportes técnicos y bitácoras de trabajo
    B. Estructura general de un trabajo científico
    C. Principales guías de estilo para referencias y trabajos

D. Tipos de publicaciones científicas

BIBLIOGRAFÍA

1. Ranjit Kumar. Research Methodology: A Step-by-Step Guide for Beginners, 4th edition. SAGE Publications. 2014
2. John M. Swales & Christine Feak. Academic Writing for Graduate Students, 3rd. Edition: Essential Tasks and Skills. University of Michigan Press. 2012
3. Kate L. Turabian. A Manual for Writers of Research Papers,Theses, and Dissertations, Eighth Edition: Chicago Style for Students and Researchers (Chicago Guides to Writing, Editing, and Publishing). University Of Chicago Press.2013
4. Kitchenham, B. et Charters, S. (2007). Guidelines for performing Systematic Literature Reviews in Software Engineering (no EBSE 2007-001). Keele University and Durham University Joint Report. Repéré à http://www.dur.ac.uk/ebse/resources/Systematic-reviews-5-8.pdf
5. Petersen, K., Feldt R., Mujtaba, S., Mattsson, M. Systematic Mapping Studies in Software Engineering. In: 12th International Conference on Evaluation and Assessment in Software Engineering. (2008).
6. Wohlin C., Höst M, and Henningsson K., Empirical Research Methods in Software Engineering. R. Conradi and A.I. Wang (Eds.): ESERNET 2001-2003, LNCS 2765, pp. 7–23, 2003. © Springer-Verlag Berlin Heidelberg 2003
7. Wieringa R. Design Science Methodology for Information Systems and Software Engineering. Springer-Verlag Berlin Heidelberg. XV, 32. 10.1007/978-3-662-43839-8. 2014
8. Michael Felderer & Guilherme Horta Travassos, Contemporary Empirical Methods in Software Engineering. Springer-Verlag Berlin Heidelberg. X, 525. 10.1007/978-3-030-32489-6

   NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor.
Exposiciones.
Resolución de problemas.

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Trabajos.

Revisiones de artículos.
Tema para titulación formalizado.

CRITERIOS DE EVALUACIÓN

Asistencia	10%
Trabajo sobre definición de tema para titulación	20%
Revisión y análisis crítico de artículo científico	10%
Tesis o Trabajo terminal	60%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Tópicos selectos de Ingeniería de Software

SEMESTRE 2, 3 ELECTIVA
CICLO ESCOLAR
21TIS01
CLAVE DE LA ASIGNATURA
21IS02
CORREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

El software forma una parte esencial en las organizaciones y está presente en prácticamente en todas las áreas de la actividad humana (negocios, finanzas, energía, transporte, educación, comunicaciones y gobierno). Debido a que las consecuencias de una falla en el software pueden ser graves, se hace esencial la funcionalidad y la seguridad confiables. Como resultado, el aseguramiento del software está emergiendo como una disciplina importante para el desarrollo, adquisición y operación de sistemas y servicios de software que brindan los niveles necesarios no solo de confiabilidad sino además de seguridad.

CONTENIDO TEMÁTICO

I. **Introducción**
   A. Historia
   B. Errores típicos en la calidad y seguridad.
II. **Fundamentos de la calidad y seguridad en el software.**
III. **Gestión de la calidad y seguridad**
   A. Modelos y estándares de Calidad
   B. Modelos y estándares de seguridad
   C. Metodologías de calidad y seguridad
IV. **Métricas y medidas en la calidad y seguridad**
   A. Goal Question Metrics
V. **Herramientas para la calidad y seguridad en el software**
VI. **Establecimiento de plan de calidad y seguridad en el desarrollo de software**

BIBLIOGRAFÍA

1. Claude Y. Laporte & Alain. Software Quality Assurance. Wiley-IEEE Computer Society PR; Edición Revised ed. 2018. 978-1118501825
2. Mary Beth Chrissis, Mike Konrad, Sandra Shrum. CMMI for Development: Guidelines for Process Integration and Product Improvement (3rd Edition) (SEI Series in Software Engineering) Addison-Wesley Professional; 2011
3. Sommerville, I. Software engineering (7ª ed.) Pearson 2004.
4. Roger S. Pressman. Ingeniería del software: un enfoque práctico. 6ta Edición Traducción de Víctor Campos Olguín, Javier Villegas Quezada. (7th. Edition). McGraw Hill. 2010
5. Humphrey W. A Discipline for Software Engineering. AddisonWesley 1995
6. Victor Basili, Gianluigi Caldiera, Dieter Rombach. The Goal Question Metric Approach. Addison-Wesley Professional; 1994.
7. Viega John & McGraw. GaryBuilding Secure Software: How to Avoid Security Problems the Right Way (paperback). Addison-Wesley Professional Computing Series, 2011.
8. Calero, Coral, and Mario G. Piattini Velthuis. Calidad del producto y proceso software. Editorial Ra-Ma, 2010.

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor.
Exposiciones.
Trabajos
Proyecto Final

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Trabajos
Exposiciones
Proyecto final
Desarrollo de proyectos
Revisión literaria

CRITERIOS DE EVALUACIÓN

Trabajos y tareas	20%
Exposiciones	20%
Proyecto Final (3 etapas del 20%)	60%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Gestión de Proyectos de Software

SEMESTRE 2 ELECTIVA
CICLO ESCOLAR
21GP01
CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

La carencia de una buena gestión de proyectos continúa siendo la causa principal del fracaso de los proyectos, repitiéndose constantemente en las organizaciones el incumplimiento en tiempo y en coste aceptable de los proyectos, en productos liberados libres de defectos, por lo que, en esta materia el alumno adquiere el conocimiento para gestionar los proyectos de manera exitosa. Al terminar la materia, el alumno será capaz de desarrollar un plan de control integral de un proyecto de software, que considere las actividades de gestión del proyecto, el alcance, el tiempo de duración, los costos, los recursos, los riesgos y la comunicación.

CONTENIDO TEMÁTICO

I. Introducción a la Gestión de Proyectos
- A. Conceptos básicos de la Gestión de Proyectos
- B. Importancia de la Gestión de Proyectos
- C. Problemática de la gestión de proyectos
- D. Beneficios de la gestión de Proyectos
- E. Implicados en la gestión de Proyectos

II. Tareas de la gestión de proyectos
- A. Selección de recursos
- B. Estimación de costos y productividad
- C. Calendarización del proyecto
- D. Simulación de alternativas
- E. Seguimiento del proyecto
- F. Indicadores financieros

III. Procesos relacionados con la Gestión de Proyectos

A. Gestión de Proyectos
B. Seguimiento y Control de Proyectos
C. Gestión de Riesgos
D. Gestión de la Calidad
E. Gestión de la Configuración

IV. Metodologías para la gestión de proyectos

V. Tendencias y herramientas en la Gestión de Proyectos

BIBLIOGRAFÍA

1. Claude Y. Laporte & Alain. Software Quality Assurance. Wiley-IEEE Computer Society PR; Edición Revised ed. 2018. 978-1118501825
2. Mary Beth Chrissis, Mike Konrad, Sandra Shrum. CMMI for Development: Guidelines for Process Integration and Product Improvement (3rd Edition) (SEI Series in Software Engineering) Addison-Wesley Professional; 2011
3. Humphrey W. A Discipline for Software Engineering. Addison Wesley 2000
4. Roger S. Pressman. Ingeniería del software: un enfoque práctico. 6ta Edición Traducción de Víctor Campos Olguín, Javier Villegas Quezada. (7th. Edition). McGraw Hill. 2010
5. Sommerville, I. Software engineering (7ª ed.) Pearson 2004
6. Gonzalo Cuevas Agustín, Gestión del Proceso Software Editorial Universitaria. Ramón Areces 2000.
7. Chow and D.-B. Cao. A survey study of critical success factors in agile software projects, 2008
8. D. Bustard, G. Wilkie, and D. Greer. Towards optimal software engineering: learning from agile practice. Innovations in Systems and Software Engineering, 9(3):191–200, 2013.
9. ISO/IEC. (2011). ISO/IEC TR 29110-5-1-2:2011 - Software engineering – Lifecycle profiles for very small entities (VSEs) – Part 5-1-2: Management and engineering guide – Generic profile group: Basic profile. Freely available from ISO at: http://standards.iso.org/ittf/PubliclyAvailableStandards/c060389_ISO_IEC_TR_29110-5-1-1_2012(E).zip
10. Project Management Institute. A Guide to the Project Management Body of Knowledge. 2005.

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor.
Exposiciones.
Tareas
Proyecto Final

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Revisión de artículos.
Exposiciones.
Tareas
Proyectos Final
Revisión literaria

CRITERIOS DE EVALUACIÓN

Tareas	20%
Exposiciones	10%
Reporte de lecturas	10%
Proyecto Final (3 etapas 20% c/u)	60%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Gestión del Proceso de Software

SEMESTRE 2, 3 ELECTIVA

CICLO ESCOLAR

21GS01

CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

En materia proporciona al alumno una visión integral de los procesos y su importancia para la madurez y capacidad de las organizaciones, además de brindar las bases para el modelado de procesos a través de la selección y uso de técnicas y herramientas existentes.

CONTENIDO TEMÁTICO

I. **Establecimiento de la cultura de procesos en las organizaciones.**
II. **Conceptos Generales de la mejora de procesos.**
    A. Introducción a la mejora de procesos.
    B. Modelos de mejora de procesos software.
    C. Uso de entornos multimodelo.
    D. Evaluación y mejora de procesos de software.
    E. Herramientas para implementar la mejora de procesos.
III. **La relación entre la calidad de procesos y la calidad de productos y servicios de software.**
IV. **Modelos y estándares de calidad.**
V. **Uso de Goal Question Metric en la definición de procesos.**
VI. **Entendimiento del entorno empresarial para la definición de procesos.**
VII. **Herramientas para el modelado de procesos.**
VIII. **Definición de procesos de software**
    A. Extracción del Conocimiento Tácito Organizacional.
IX. **Evaluación de procesos de software**
    A. Trazabilidad entre objetivos de negocio y procesos organizacionales.
    B. Métodos para evaluación de procesos
X. **Aplicación de estudio de caso.**
    A. Establecimiento de equipos.

B. Identificación de necesidades.
C. Propuesta de Objetivos de negocio y métricas.
D. Extracción de conocimiento tácito.
E. Propuesta de modelado de procesos.
F. Trazabilidad entre objetivos de negocio y procesos.
G. Propuesta de mejora para la organización

BIBLIOGRAFÍA

1. Mary Beth Chrissis, Mike Konrad, Sandra Shrum. CMMI for Development: Guidelines for Process Integration and Product Improvement (3rd Edition) (SEI Series in Software Engineering) Addison-Wesley Professional; 2011
2. ISO/IEC. (2011). ISO/IEC TR 29110-5-1-2:2011 - Software engineering – Lifecycle profiles for very small entities (VSEs) – Part 5-1-2: Management and engineering guide – Generic profile group: Basic profile. Freely available from ISO at: http://standards.iso.org/ittf/PubliclyAvailableStandards/c060389_ISO_IEC_TR_29110-5-1-1_2012(E).zip.
3. Gonzalo Cuevas Agustín, Gestión del Proceso Software Editorial Universitaria. Ramón Areces 2000.
4. P. Bourque and R.E. Fairley, eds., *Guide to the Software Engineering Body of Knowledge, Version 3.0*, IEEE Computer Society, 2014; www.swebok.org.
5. Kim Caputo. CMM Implementation Guide: Choreographing Software Process Improvement. Addison-Wesley Professional; 1998.
6. Mirna Muñoz, Gonzalo Cuevas, Tomás San Feliu. Metodología multimodelo para implementar mejoras de procesos de software. Editorial Académica Española. 2012.
7. Victor Basili. Goal Question Metric A methodology for collecting valid software engineering data. Basili, V.R., D.M. Weiss 1984
8. SEI. SCAMPI Standard CMMI Appraisal Method for Process Improvement SEI/CMU 2004
9. Han van Loon. Process Assessment and ISO/IEC 15504: A Reference Book Springer 2nd ed. 2007 edition(November 16, 2014)

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor.
Exposiciones.
Resolución de problemas.

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Tareas
Exposiciones
Reportes
Casos de estudio.

CRITERIOS DE EVALUACIÓN

Trabajos	20%
Exposiciones	20%
Proyecto Final (3 etapas 20% c/u)	60%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Dirección de Equipos de Desarrollo de Software

SEMESTRE 2, 3 ELECTIVA

CICLO ESCOLAR

21DE01

CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

Esta materia proporcionará al alumno una visión integral de la importancia de la conformación de equipos de desarrollo de software mediante la impartición de conocimientos y habilidades necesarias para integrar, liderar y reforzar la colaboración de manera efectiva de equipos de desarrollo de software, de tal manera que el individuo identifique el sistema del que forman parte, se conozca y conozca cómo funcionan los equipos dentro del macrosistema.

CONTENIDO TEMÁTICO

I. El pensamiento sistémico ¿qué es y para qué sirve?
- A. Elementos clave del pensamiento sistémico.
- B. Teoría general de los sistemas.
- C. Tipos de Sistemas (abiertos y cerrados)
- D. Niveles de Complejidad (subsistemas, sistemas, suprasistemas)
- E. Puntos de apalancamiento.
- F. Aplicando sistemas en mi contexto: Genograma / Organigrama

II. Genograma, Sistema de Creencias y Pautas generacionales
- A. Aplicando sistemas en mi contexto: Genograma / Organigrama
- B. Sistema de Creencias.
- C. Pautas Generacionales.

III. El pensamiento sistémico en la vida cotidiana y las organizaciones
- A. Axiomas de la comunicación humana
- B. Del pensamiento lineal al pensamiento circular
- C. Causas lineales vs. causas circulares ejemplos
- D. Pautas y secuencias
- E. Cambios de primer y segundo orden

IV. Pensar en equipos es pensar en sistemas
- A. Hipótesis sistémicas
- B. Soluciones intentadas fracasadas, soluciones acertadas
- C. Revisión de un caso, trabajar con equipos en las organizaciones

V. Fundamento para la Formación de Equipos
- A. Perfiles de personalidad
- B. Roles necesarios para el cambio
- C. Equipos vs grupos
- D. Razones de fracaso de un equipo
- E. Características comunes a los equipos eficaces
- F. Diagnóstico de equipos de trabajo y proyección de los mismos

VI. Técnicas para facilitar reuniones de equipos
- A. Dirigir reuniones productivas
- B. Proceso de reuniones
- C. Técnicas de decisión de equipos
- D. Habilidades para interactuar en equipo

VII. Modelos de crecimiento del equipo
- A. Actitudes y comportamiento
- B. Etapas del modelo de crecimiento

VIII. Proceso de desarrollo de software en equipo

IX. Desarrollo de Proyecto

BIBLIOGRAFÍA

1. Raúl Medina, Esteban Laso y Eduardo Hernández Pensamiento sistémico: Nuevas perspectivas y contextos de intervención Litteris psicología 2014
2. Peter Senge. La quinta disciplina. Garnica 2004
3. Ludwig Von Bertalanffy Teoría general de los sistemas Fondo de Cultura Económica. 1989
4. Watts S. Humphrey Addison-Wesley Introduction to the Team Software. Process Professional. 1999
5. Watts S. Humphrey. TSP: Coaching Development Teams (The SEI Series in Software Engineering) Addison-Wesley Professional 2006
6. Bill Curtis and William E. Hefley The People CMM: A Framework for Human Capital Management (2nd Edition) Professional and Addison-Wesley 2009
7. Jan Beave The Agile Team Handbook r CreateSpace Independent Publishing Platform 2013
8. Lyssa Adkins. Coaching Agile Teams: A Companion for ScrumMasters, Agile Coaches, and Project Managers in Transition (Addison-Wesley Signature Series (Cohn)) 2010

NOTA: La materia se apoyará con el uso de artículo científicos relacionados con los temas, por lo tanto, el docente y el alumno pueden hacer uso de la biblioteca digital http://www.cimat.mx/es/Catalogos_Servicios_en_Linea pueden acceder utilizando correo institucional, utilizando su cuenta y contraseña

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Participación
Ejercicios
Tareas
reportes
Proyecto

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Participación
Tareas
Reportes
Proyecto
Revisión de literatura

CRITERIOS DE EVALUACIÓN

Exposiciones	20%
Trabajos	30%
Reportes de lecturas	10%
Proyecto final	40%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Arquitectura de Software

SEMESTRE 2, 3 ELECTIVA
CICLO ESCOLAR
21AS01
CLAVE DE LA ASIGNATURA
21IS01
CORREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

En el contexto de la ingeniería de software el desarrollo de la arquitectura tiene que ver con la estructuración de un sistema para satisfacer los requerimientos de clientes y otros involucrados, en especial los requerimientos de atributos de calidad. En esta materia se abordan aspectos relacionados al desarrollo de la arquitectura con un énfasis importante hacia las bases teóricas, pero también realizando ejercicios prácticos que permiten relacionar la teoría con la realidad. Al término de este curso el alumno deberá comprender el concepto de arquitectura de software en el contexto de desarrollo de sistemas.

CONTENIDO TEMÁTICO

I. Identificación de requisitos arquitectónicos
- A. Requisitos arquitectónicos
- B. Métodos de identificación de requisitos arquitectónicos

II. Diseño de la arquitectura
- A. Principios de Diseño de Sistemas
- B. Conceptos de Diseño de Arquitectura: Patrones, Tácticas y Tecnologías.
- C. Métodos de Diseño de Arquitectura

III. Documentación de la arquitectura
- A. Vistas Arquitectónicas
- B. Notaciones
- C. Métodos de Documentación de la Arquitectura

IV. Evaluación de la arquitectura

A. Tipos de Evaluación
B. Métodos de Evaluación de Arquitectura

V. Tópicos Avanzados

BIBLIOGRAFÍA

1. Booch G. El lenguaje Unificado de Modelado, UML 2.0, Guia de Usuario. 1ª. Edición Pearson ADDISONWESLEY 2006
2. Len Bass, Paul Clements, and Rick Kazman, Software Architecture in Practice Addison Wesley 2012
3. Anthony J. Lattanze. Architecting Software Intensive Systems: A Practitioner's Guide Taylor and Francis/Auerbach 2008
4. Richard N. Taylor, Nenad Medvidovic, and Eric MSoftware Architecture: Foundations, Theory and Practice Addison Wesley 2007
5. Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little and Robert Nord. Documenting Software Architectures: Views and Beyond, Addison Wesley 2011
6. Tavish Armstrong. The Performance of Open Lulu.com, 2013
7. Amy Brown and Greg Wilson The Architecture Of Open Source Applications Lulu.com Vol I, 2011 Vol II, 2012
8. Roger S. Pressman. Ingeniería del software: un enfoque práctico. 6ta Edición Traducción de Víctor Campos Olguín, Javier Villegas Quezada. (7th. Edition). McGraw Hill. 2010
9. Sommerville, I. Software engineering (7ª ed.) Pearson 2004
10. Shari Lawrence Pfleeger. Ingeniería de Software. Teoría y Práctica. Prentice Hall. 2002

ACTIVIDADES DE APRENDIZAJE
BAJO CONDUCCIÓN DE UN ACADÉMICO

Exposiciones por parte del profesor
Exposiciones por parte del alumno
Desarrollo de prácticas
Desarrollo de un proyecto integrador

ACTIVIDADES DE APRENDIZAJE
INDEPENDIENTES

Tareas.
Revisiones de literatura
Reportes

CRITERIOS DE EVALUACIÓN

Proyecto final (3 etapas 20% c/u)	60%
Tareas	10%
Exámenes	30%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Optimización de Procesos (Optimización Estadística)

SEMESTRE 2,3 ELECTIVA
CICLO ESCOLAR
21OP01
CLAVE DE LA ASIGNATURA
21ED01
PRERREQUISITOS

FINES DE APRENDIZAJE O FORMACIÓN

Comprender el razonamiento de los métodos estadísticos de diseño y análisis de experimentos, regresión lineal y otros conceptos útiles para la optimización estadística de procesos, con el fin de buscar la mejora continua en problemas industriales, científicos y tecnológicos.

CONTENIDO TEMÁTICO

I. Introducción al Diseño de Experimentos

A. El diseño de experimentos en la industria
B. Diseño de experimentos en la investigación
C. Definiciones básicas en el diseño de experimentos
D. Etapas en el diseño de experimentos
E. Consideraciones prácticas sobre el uso de estadísticos
F. Clasificación y selección de los diseños experimentales
G. Conceptos estadísticos

II. Diseños con un solo factor: Análisis de Varianza

A. Diseño completamente aleatorizado
B. Análisis de Varianza (ANOVA)
C. Análisis del modelo con efectos fijos
D. Verificación de la adecuación y supuestos del modelo
E. Interpretación de los resultados

III. Modelos de regresión lineal

A. Regresión lineal simple
B. Análisis de varianza
C. Coeficiente de correlación y de determinación
D. Inferencia sobre los parámetros del modelo
E. Inferencia sobre la predicción y la media de la predicción
F. Análisis de residuales
G. Regresión lineal múltiple
H. Variables Dummy

IV. Diseños en bloques

A. Diseño en bloques completamente aleatorizado
B. Cuadro latino
C. Cuadro grecolatino
D. Diseño de bloques incompletos balanceados

V. Diseños Factoriales

A. Ventajas de los diseños factoriales
B. Diseño factorial con dos y tres factores
C. Diseño factorial general
D. Ajuste de curvas y superficies de respuesta
E. Modelos de efectos aleatorios
F. Diseño factoriales 2k
G. Diseños factoriales 22 y 23
H. Diseño factorial 2k no replicado
I. Diseño factorial 2k con puntos centrales
J. Diseño factorial 2k con bloques
K. Diseño factorial 3k y factoriales

VI. Diseños Factoriales Fraccionados

A. Diseños factoriales fraccionados 2k-p
B. Concepto de resolución
C. Resolución y estructura de alias en los diseños 2k-p
D. Fracción un medio y un cuarto del diseño 2k
E. Experimentos factoriales fraccionados 3k

VII. Diseños de mezclas

A. Diseños de Látice simplex
B. Diseños de Centroides simplex
C. Diseños con restricciones

VIII. Optimización de procesos con metodología de superficie de respuesta

A. Concepto de optimización
B. Introducción a la metodología de superficie de respuesta (MRS)
C. Modelos de superficie de respuesta
D. Diseños de primer orden
E. Diseños de segundo orden
F. Pérdida de ajuste
G. Técnicas de optimización
H. Diseños experimentales para ajustar superficies de respuesta

I. Escalamiento ascendente
J. Análisis canónico, y análisis de cordillera
K. Optimización simultánea de varias respuestas: Función de deseabilidad

BIBLIOGRAFÍA

1. Box, G. E. P., Hunter, J. S., and Hunter, W. G. *Statistics for experimenters: design, innovation, and discovery*. Wiley-Interscience, 2005.
2. Castillo, E. D. *Process Optimization*, vol. 105 of *International Series in Operations Research & Management Science*. Springer US, Boston, MA, 2007.
3. Domínguez y Domínguez, J., and Castaño Tostado, E. *Diseño de experimentos: estrategias y análisis en ciencias e ingenierías*. Alfaomega, 2018.
4. Gutiérrez, H. y R. de la Vara. *Análisis y diseño de experimentos*. 2ª edición McGraw Hill/Interamericana Editores, S.A. de C.V. 2008.
5. Khuri, A. I. *Response Surface Methodology and Related Topics*. WORLD SCIENTIFIC, jan 2006.
6. Kuehl, R. O., and Kuehl, R. O. *Design of experiments: statistical principles of research design and analysis*. Duxbury/Thomson Learning, 2000.
7. Montgomery, D. C. *Design and analysis of experiments*. Wiley, 2009.
8. Myers, R. H. A.-C. *Response Surface Methodology: Process and Product Optimization Using Designed Experiments.* Wiley, 2016.
9. Wu, C.-F., and Hamada, M. *Experiments: planning, analysis, and optimization*. Wiley, 2009.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor.
Exposiciones.
Resolución de problemas.
Exámenes

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Trabajos.
Revisiones de artículos, estudios de caso y reportes de investigación.

CRITERIOS DE EVALUACIÓN

Asistencia	10%
Tareas	40%
Exámenes (2)	50%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Diseño Interfaces y Experiencia del Usuario

SEMESTRE 2, 3 ELECTIVA

CICLO ESCOLAR

21DU01

CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

El alumno conocerá las metodologías de análisis y diseño centradas en el usuario y su experiencia que le permitan el desarrollar y evaluar interfaces y entornos interactivos.
El alumno conocerá metodologías de diseño centradas en el usuario, además de principios, guías y estándares que le permitan desarrollar sistemas interactivos usables.
El alumno conocerá principios del factor humano y sistemas computacionales que le permitan proponer interfaces óptimas en el desarrollo de sus sistemas interactivos.
El alumno conocerá las tecnologías para el desarrollo de entornos interactivos que pueden ser aprovechadas en diversos ámbitos como el entretenimiento, la educación y la medicina, entre otros.
El alumno desarrollará entornos interactivos utilizando el motores de juegos, además conocerá las capacidades que ofrecen para la integración de dispositivos y herramientas interactivas.

CONTENIDO TEMÁTICO

I. El factor humano y los sistema computacionales
- A. Disciplinas participantes
- B. Los aspectos del humano
- C. Los aspectos del sistema
- D. Los aspectos de la interacción

II. Experiencia del Usuario (EU)
- A. Componentes de la experiencia de usuario
- B. Modelos de evaluación de EU
- C. Evaluación fisiológica

III. Procesos para la Experiencia de Usuario

A. Eligiendo los métodos, técnicas y procesos
B. Proceso ágil

IV. Introducción al diseño de entornos interactivos

A. Desarrollo de un sistema interactivo
    1. Ciclo de desarrollo del software
    2. Proceso del diseño de la interfaz.
B. Diseño centrado en el usuario
C. Análisis y modelado de usuarios, tareas y entornos
    1. Métodos de análisis de tareas
D. Reglas para el diseño de interfaces de usuario
E. Principios, guías, estándares
F. Prototipado
G. Desarrollo de interfaces de usuario

V. Introducción a la Usabilidad en los entornos interactivos

A. Objetivos de la usabilidad en los entornos interactivos
B. Importancia de la usabilidad
C. Evaluación de la Usabilidad en los entornos interactivos
D. Métodos de evaluación de la usabilidad en los entornos interactivos
    1. Métodos de inspección
    2. Métodos de indagación
    3. Métodos de test
    4. Laboratorios de Usabilidad

VI. Tecnologías para el desarrollo de entornos interactivos en realidad virtual

A. Conceptos básicos y fundamentos de Realidad aumentada (AR), Realidad Virtual (VR) y Realidad extendida (XR)
B. Áreas de conocimiento
C. Percepción sensorial en el ser humano
D. Arquitectura de un sistema de realidad virtual
E. Niveles de interacción e inmersión
F. Interfaces de usuarios espaciales
G. Interfaces de usuarios diegéticas
H. Interacción con elementos de la interfaz de usuario

VII. Desarrollo de aplicaciones en realidad virtual y aumentada

A. Los dispositivos para realidad virtual
    1. Herramientas para el desarrollo
    2. Ejemplos prácticos en unity
B. Introducción a la realidad aumentada
    1. Herramientas para el desarrollo
    2. Desarrollo de aplicaciones para dispositivos móviles
    3. Ejemplos prácticos en unity

BIBLIOGRAFÍA

1. Dix, A., Dix, A. J., Finlay, J., Abowd, G. D., & Beale, R. (2003). Human-computer interaction. Pearson Education.
2. Nielsen, J. (1994). Usability engineering. Morgan Kaufmann.
3. Jerald, J. (2015). The VR book: Human-centered design for virtual reality. Morgan & Claypool.
4. Steven, M. L. (2016). Virtual Reality. Cambridge University Press.
5. Fictum, C. (2016). VR UX: learn VR UX, storytelling & design. Creater Space Independent Publishing Platform.
6. Buttfield-Addison, P., Manning, J., & Nugent, T. (2019). Unity Game Development Cookbook: Essentials for Every Game. O'Reilly Media.
7. Greengard, S. (2019). Virtual reality. Mit Press.
8. Hartson, R., & Pyla, P. S. (2012). The UX Book: Process and guidelines for ensuring a quality user experience. Elsevier.
9. Hartson, R., & Pyla, P. S. (2018). The UX book: Agile UX design for a quality user experience. Morgan Kaufmann.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor.
Prácticas en clase.
Presentación de trabajos.

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Trabajos de diseño y/o desarrollo.
Revisiones de artículos.
Desarrollo de presentaciones.

CRITERIOS DE EVALUACIÓN

Asistencia y participación	5%
Prácticas	35 %
Proyecto integrador	35%
Presentaciones	15%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Equipo tecnológico en laboratorio.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Cómputo Paralelo

SEMESTRE 2, 3 ELECTIVA
CICLO ESCOLAR
21CP01
CLAVE DE LA ASIGNATURA
21PA01
CORREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

La asignatura presenta un recorrido por las técnicas y herramientas más utilizadas para el análisis, diseño, implementación y comparación de algoritmos paralelos sobre diversas plataformas. Se revisan diferentes modelos de programación, haciendo especial énfasis en el uso de herramientas estándar como OpenMP, MPI y CUDA. Se analiza la utilización de estas herramientas en aplicaciones de varios ámbitos diferentes, incluyendo la optimización matemática.

CONTENIDO TEMÁTICO

I. Introducción

A. Arquitectura Von Neumann
B. Desempeño del procesador
C. Arquitecturas UMA y NUMA
D. Sistema multiprocesamiento
E. Caché: coherencia, caché hits, cache miss
F. Prefetchers
G. Relación de los conceptos anteriores con OpenMP
H. Análisis de rendimiento: Valgrind (Cachegrind, cg_annotate)

II. Programación en C

A. Organización de la memoria (Heap, Stacks, Globals)
B. Programación para uso eficiente del caché

III. Conceptos básicos en cómputo paralelo

A. Representación de algoritmos en un Grafo Dirigido Acíclico (DAG)

B. Medidas de complejidad
C. Aceleración teórica y eficiencia
D. Métricas para entornos heterogéneos
E. La ley de Amdahl
F. Escalabilidad
G. Funciones de overhead
H. Extracción de paralelismo: modelo de datos paralelos, modelo de granjas, grafos de tareas, pipeline
I. Fuentes de pérdida de rendimiento

IV. OpenMP

A. Introducción
B. Deadlocks, sincronía y conformidad con el estándar
C. Directivas
D. Cláusulas
E. Modelo de memoria
F. Librerías
G. Estructura de un programa C con OpenMP
H. Reducciones
I. Sincronización
J. Tareas
K. Anidaciones
L. Locks
M. Aplicaciones

V. MPI

A. Introducción
B. Modelo de programación
C. Compilación y ejecución
D. Estructura de programas MPI en C
E. SSH
F. Tipos de datos simples
G. Comunicaciones: punto a punto o en grupo, sincronía, bloqueos
H. Implementación interna de operaciones de comunicación grupales
I. Administración del ambiente
J. Buffering
K. Estructuras y datos derivados
L. Topologías
M. Empaquetamiento
N. Debuggeo con MPI.
O. Aplicaciones

VI. Optimización paralela

A. Introducción a metaheurísticas
B. Paralelización de técnicas de optimización exactas
C. Paralelización de técnicas de optimización aproximadas
D. Análisis de rendimiento

VII. CUDA

A. Introducción
B. Arquitectura de las Unidades de Procesamiento Gráficas (GPU)
C. Modelo de programación
D. Compilación y ejecución
E. Bloques y threads

F. Sincronización
G. Operaciones atómicas
H. Compartición de datos
I. Uso eficiente de GPUs
J. Algoritmos paralelos
K. Streams
L. Análisis de rendimiento
M. Librerías
N. Aplicaciones.

BIBLIOGRAFÍA

1. A. Grama, A. Gupa, G. Karypis, V. Kumar. Introduction to Parallel Computing. Addison-Weley, 2003
2. D.P. Bertsekas and J.N. Tsitsiklis. Parallel and Distributed Computation: Numerical Methods. Athena Scientific, 1997.
3. Quinn, M. J. Parallel Computing: Theory and Practice. McGraw-Hill, New York, 1994.
4. J. Jájá: An Introduction to Parallel Algorithms. Addison Wesley, Massachusetts, 1992.
5. Valgrind documentation: http://valgrind.org/docs/manual/Rogue
6. Wave Software. 8 Steps to Optimizing Cache Memory Access and Application Performance. A recommended approach for cache memory optimization, 2011
7. Markus Kowarschik, Christian Weib. Chapter "An Overview of Cache Optimization Techniques and Cache-Aware Numerical Algorithms" in Algorithms for Memory Hierarchies, 2003
8. Brian Dougherty, Jules White, Russell Kegley, Jonathan Preston, Douglas C. Schmidt, and Aniruddha Gokhale, Optimizing Integrated Application Performance with Cache-aware Metascheduling, Proceedings of the 1st International Symposium on Secure Virtual Infrastructure, 2011
9. Stefano Cozzini. Optimization techniques: an overview. http://http://www.democritos.it/events/computational_physics/lecture_stefano3.pdf, 2005.
10. OpenMP Specification: http://www.openmp.org
11. R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald, R. Menon. Parallel Programming in OpenMP. Morgan Kaufmann Publishers, 2000.
12. B. Chapman, G. Jost, R. V. Der Pas. Using OpenMP. Portable Shared Memory Parallel Programming. The MIT Press, 2008
13. M. J. Quinn. Parallel Programming in C with MPI and OpenMP. McGraw Hill, 2003.
14. P. S. Pacheco. Parallel Programming with MPI. Morgan Kaufmann, 1997.
15. Jason Sanders, Edward Kandrot. CUDA by Example: An Introduction to General-Purpose GPU Programming. Addison-Wesley, 2011.
16. Curso udacity: Intro to Parallel Programming: https://www.udacity.com/course/cs344
17. CUDA C Programming Guide: http://docs.nvidia.com/cuda/cuda-c-programming-guide/

18. S. Mohanty, A. K. Mohanty, F. Carminati. Efficient pseudo-random number generation for Monte-Carlo simulation using graphic processors. Journal of Physics, 2012.
19. John Cheng, Max Grossman, Ty McKercher. Professional CUDA C Programming. Wiley, 2014.
20. S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi. Optimization by Simulated Annealing. Science 220 (13), 1983.
21. Enrique Alba. Parallel Metaheuristics: A New Class of Algorithms. Wiley, 2005.
22. A. Corana, M. Marchesi, C. Martini, S. Ridella. Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm. ACM Transactions on Mathematical Software 13 (3), 1987.
23. J. S. Higginson, R. R. Neptune, F. C. Anderson. Simulated parallel annealing within a neighborhood for optimization of biomechanical systems. Journal of Biomechanics 38 (9), 2005.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Cursos presenciales
Resolución de ejercicios
Programación de software

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Trabajos.
Revisiones de artículos.
Resolución de ejercicios
Programación de software

CRITERIOS DE EVALUACIÓN

Tareas	60%
Primer proyecto	20%
Segundo proyecto	20%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.

Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Aprendizaje de Máquina 1

SEMESTRE 2, 3 ELECTIVA
CICLO ESCOLAR
21AM01
CLAVE DE LA ASIGNATURA
21PA01
CORREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

Introducir al estudiante a los conceptos teóricos de optimización, aprendizaje máquina y aprendizaje profundo, así como al desarrollo práctico de soluciones basadas en aprendizaje de máquina profundo, principalmente con el uso de redes neuronales profundas de convolución y recurrentes.

CONTENIDO TEMÁTICO

I. Introducción a Aprendizaje de Automático
- A. Fundamentos de Optimización
  1. Introducción a la optimización
  2. Optimalidad en problemas sin restricciones
  3. Optimalidad en Problemas con restricciones
- B. Fundamentos de Aprendizaje Automático
  1. Análisis de componentes principales (PCA)
  2. Mínimos cuadrados
  3. Regularización: Ridge (L2), Lasso (L1) y Elastic Net (L2+L1)]
  4. Regresión Logística

II. Introducción al Aprendizaje Profundo
- A. Aprendizaje Profundo

1. Revisión a la regresión logística y el perceptrón
2. Redes multicapa
3. Backpropagation

B. Redes Profundas de Convolución (ConvNN)
1. Capas de convolución, Pooling, Dropout, Normalización por lotes,
2. Aumentación de datos
3. Redes pre-entrenadas para problemas con base de datos pequeñas
4. Variaciones en arquitecturas de NN

C. Redes Profundas Recurrentes
1. Incrustación de datos (Embedding)
2. Redes recurrentes profundas (RNNs)
3. Redes de memoria larga para términos cortos (LSTM)
4. Problema del gradiente evanescente

D. Visualización de activación en redes convolucionales
1. Mapas de saliencia
2. Mapas de activación

III. Arquitecturas Modernas para el Aprendizaje Profundo

A. Avances en ConvNN
1. Redes Residuales
2. Capas especiales: Concatenación, Convolución transpuesta, Lambda,
3. Interpolación

B. Redes Modernas
1. Autocodificador Variacional
2. UNet para segmentación de imágenes
3. Redes Generadoras Antagónicas (GANs)
4. Redes Generadoras Antagónicas Convolucionales

BIBLIOGRAFÍA

1. Trevor Hastie, Robert Tibshirani and Jerome Friedman, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer. 2 Ed. (2013). https://web.stanford.edu/~hastie/Papers/ESLII.pdf
2. Christopher Bishop, "Pattern Recognition and Machine Learning; Springer, (2016). https://www.microsoft.com/en-us/research/people/cmbishop/prml-book/
3. Ian Goodfellow, Yoshua Bengio and Aaron Courville, "Deep Learning", MIT Press. (2016). http://www.deeplearningbook.org
4. Francois Chollet, "Deep Learning with Python", Manning Pubs. Packt Publishing Ltd, (2018)
5. Mariano Rivera, "Tópicos de Aprendizaje Automático: Notas de Clase", disponible electrónicamente(2018). http://personal.cimat.mx:8181/~mrivera/cursos/temas_aprendizaje.html

Otro Material

- https://www.continuum.io/downloads http://scikit-learn.org/stable/#
- https://keras.io
- http://www.sympy.org/en/index.html
- http://scikit-learn.org/stable/

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor
Ejercicios
Exámenes

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Tareas
Lectura de publicaciones especializadas

CRITERIOS DE EVALUACIÓN

Tareas	50%
3 Exámenes	50%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC
Aula virtual
Asesorías y aula virtual
Bibliotecas digitales

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Programación y Algoritmos II

SEMESTRE 2, 3 ELECTIVA
CICLO ESCOLAR
21PA02
CLAVE DE LA ASIGNATURA
21PA01
PRERREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

Esta materia trata problemas de algoritmia avanzada para alcanzar altos niveles de eficiencia en programación. Toma como soporte educativo principal la base de datos de problemas de tipo ACM, con el fin de que el estudiante pueda rápidamente identificar un tipo de problema computacional así como programar una solución eficiente en tiempo y memoria

CONTENIDO TEMÁTICO

I. **Introducción: tips generales en concursos de tipo ACM**
II. **Estructuras de datos avanzadas**
    a. Repaso: estructuras de datos lineares. Arreglos estáticos, dinámicos; listas ligadas; pilas; colas; arreglos de bits; std::vector, std::deque, std::list, std::stack; std::queue; std::bitset
    b. Estructuras de datos no-lineares:
        1. Árboles binarios balanceados: árboles AVLs, árboles RB; std::map y std::set
        2. B-trees
        3. Colas de prioridad y montículos. std::priority_queue
        4. Montículos binomiales; montículos de Fibonacci
        5. Tablas de hash. std::unordered_map
    c. Estructuras de datos para grafos
    d. Estructuras de datos para conjuntos disjuntos
    e. Árbol de segmento

f. Quadtrees y Octrees
g. Árbol de Fenwick
h. Estructuras dedicadas a cadenas de caracteres: tries
i. Estructuras de datos persistentes

III. Paradigmas de resolución de problemas.

a. Búsqueda exhaustiva.
b. Divide y vencerás
c. Programación dinámica

IV. Algoritmos sobre grafos

a. Recorridos de grafos
b. Árboles generadores mínimos
c. Caminos más cortos, nodo fuente único
d. Caminos más cortos, nodos fuentes múltiples
e. Flujos máximos
f. Casos en grafos particulares

V. Resolución de problemas matemáticos

a. Manejo de los enteros grandes
b. Combinatoria
c. Problemas relacionados a la teoría de números
d. Búsqueda de ciclos
e. Teoría de juegos
f. Potencias de matrices

VI. Procesamiento de cadenas

a. Procesamiento clásicos
b. Emparejamiento de cadenas
c. Programación dinámica para el procesamiento de cadenas
d. Manejo de los tries (ver II.H)

VII. Geometría computacional

a. Representación de objetos geométricos
b. Representación de polígonos
c. Algoritmos involucrando polígonos

BIBLIOGRAFÍA

1. S. Halim and F. Halim. Competitive Programming. http://cpbook.net/
2. T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein. Introduction to Algorithms (3rd ed.). MIT Press and McGraw-Hill, 2009.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Cursos presenciales
Diseño de algoritmos para la resolución de problemas
Programación de esquemas para la resolución problemas
Uso de bases de datos de problemas

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Trabajos.
Revisiones de artículos.
Tema para titulación formalizado.

CRITERIOS DE EVALUACIÓN

Exámenes parciales	40%
Tareas	60 %

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Cómputo Evolutivo

SEMESTRE 2,3 ELECTIVA
CICLO ESCOLAR
21CE01
CLAVE DE LA ASIGNATURA
21PA01
CORREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

En esta asignatura se introducen y analizan diversos métodos de optimización aproximados basados en cómputo evolutivo para los ámbitos de optimización con restricciones y optimización multi-objetivo. Además, se cubre la forma de diseñar e implementar algoritmos evolutivos paralelos. Para cada propuesta se analizan sus fundamentos y se llevan a la práctica a través de la resolución de problemas complejos.

CONTENIDO TEMÁTICO

I. **Optimización con restricciones**
   A. Funciones de penalización
   B. Decodificadores
   C. Operadores especiales
   D. Ordenamiento estocástico
   E. El método de la restricción épsilon
   F. Aplicación de esquemas y conceptos multi-objetivo
   G. Análisis de rendimiento
II. **Optimización multi-objetivo**
   A. Dominancia de Pareto
   B. Análisis de Rendimiento y Métricas multi-objetivo
      1. Hipervolumen
      2. Indicador Épsilon
      3. Contribución

4. Superficies de cubrimiento
5. Test Estadísticos
6. Otras métricas

C. Funciones de Escalarización
D. Variantes iniciales de algoritmos evolutivos multi-objetivo
E. Algoritmos evolutivos basados en la dominancia de Pareto
F. Algoritmos evolutivos basados en descomposición
G. Algoritmos evolutivos basados en indicadores
H. Funciones de Benchmark Multi-objetivo
I. Mecanismos de Preservación de diversidad
1. Espacio de las variables
2. Espacio objetivo

III. Algoritmos evolutivos paralelos

A. Paralelización de la función de evaluación
B. Modelo Maestro-Trabajadores
C. Esquemas basados en islas
D. Modelos celulares
E. Métricas de rendimiento

BIBLIOGRAFÍA

1. E. Mezura-Montes. Constraint-Handling in Evolutionary Optimization, Springer, 2009.
2. C. Coello, G. B. Lamont, D. Van Veldhuizen. Evolutionary Algorithms for Solving Multi-Objective Problems, Springer, 2007.
3. K. Deb. Multi-objective Optimization using Evolutionary Algorithms, Wiley, 2001.
4. E. Alba. Parallel Metaheuristics: A New Class of Algorithms. Wiley, 2005.
5. El-Ghazali Talbi. Metaheuristics: From Design to Implementation. Wiley, 2009.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Cursos presenciales
Resolución de ejercicios
Desarrollo de software de cómputo matemático

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Lectura de publicaciones recientes
Desarrollo de un proyecto
Preparación de presentaciones

CRITERIOS DE EVALUACIÓN

Tareas Semanales	35%
Tareas Mensuales	35%
Proyecto	25%
Presentación de proyecto	5%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC
Aula virtual
Asesorías y aula virtual
Bibliotecas digitales

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Procesamiento de Lenguaje Natural

SEMESTRE 2, 3 ELECTIVA
CICLO ESCOLAR
21PL01
CLAVE DE LA ASIGNATURA
21PA01
CORREQUISITO

FINES DE APRENDIZAJE O FORMACIÓN

El lenguaje humano es un medio de comunicación muy eficaz, y a su vez extremadamente complejo. Uno de los retos a resolver en esta era del conocimiento y la información es el tratamiento del lenguaje por medios automáticos. El objetivo general de este curso es proporcionar a los estudiantes conocimientos fundamentales y avanzados del Procesamiento del Lenguaje Natural y a las herramientas disponibles actualmente, que representan un nicho de oportunidad para el desarrollo de investigación y trabajos de alto impacto en ciencias de la computación con aplicación a una amplia variedad de áreas incluyendo, ciencias sociales, humanidades, lingüística, etc.

CONTENIDO TEMÁTICO

I. **Fundamentos**
   A. Procesamiento de Texto, Normalización y Segmentación
   B. Recursos Léxicos y Ontologías
   C. Etiquetado y Extracción de Información en Texto
   D. Información Sintáctica y Semántica

II. **Minería de Textos**
   A. Clasificación Supervisada y Patrones Lingüísticos
      1. Bolsas de Términos
      2. Esquemas de Pesado Automáticos
      3. Análisis de Sentimientos y Estilo
   B. Representaciones Basadas en Conceptos
      1. Análisis Semántico Latente (LSA) [PCA para Texto]

2. Asignación Latente de Dirichlet (LDA)
3. Representaciones Distribucionales de Términos

III. Modelos de Lenguaje

A. Modelos de Lenguaje Estadísticos
B. Modelos de Lenguaje Neuronales
C. Incrustaciones de Términos (Embeddings)
    1. Word2Vec
    2. Glove
    3. FastText

IV. Aprendizaje Profundo para Procesamiento de Lenguaje Natural

A. Redes Neuronales Convolucionales
B. Redes Neuronales Recurrentes
    1. Gated Recurrent Units (GRUs)
    2. Long Short Term Memories (LSTMs)
C. Modelos Secuencia a Secuencia (Sequence 2 Sequence)
D. Mecanismos de Atención en Redes Neuronales
E. Transformadores (Transformers)
F. Codificadores Bidireccionales de Transformadores (BERT)
G. Tópicos Tendencia en Procesamiento de Lenguaje Natural
    1. Generación Adversaria de Texto
    2. Ataques Adversarios
    3. Transferencia de Conocimiento

BIBLIOGRAFÍA

1. Speech and Language Processing, Daniel Jurafsky, James H Martin. Pearson.
2. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit, Steven Bird, Ewan Klein, Edward Loper. O'Relly.
3. Deep Learning. Ian Goodfellow, Yoshua Bengio, Aaron Courville. MIT Press.
4. Deep Learning with Python. Francois Chollet. Maning.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Cursos presenciales
Desarrollo de proyectos

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Lectura de publicaciones especializadas

CRITERIOS DE EVALUACIÓN

Tareas	40%
Exámenes (2)	30%
Presentación final	30%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC
Aula virtual
Asesorías y aula virtual
Bibliotecas digitales

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Optimización

SEMESTRE 2, 3 ELECTIVA

CICLO ESCOLAR

21OP02

CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

Este curso ofrece a los alumnos fundamentos sólidos de optimización sin restricciones, con un panorama completo de los diferentes algoritmos existentes para este tipo de problemas.

CONTENIDO TEMÁTICO

I. Introducción
- A. Formulación matemática
- B. Ejemplo: Un problema de transporte
- C. Tipos de problemas de optimización
- D. Algoritmos de optimización
- E. Convexidad

II. Fundamentos de optimización sin restricciones
- A. ¿Qué es una solución?
- B. Algoritmos (una visión preliminar)
  1. Búsqueda en línea
  2. Métodos de región de confianza

III. Métodos de búsqueda en línea
- A. Tamaño de paso
- B. Algoritmos para selección del tamaño de paso

IV. Métodos de región de confianza
- A. Punto de Cauchy

V. Métodos de gradiente conjugado
- A. Método de gradiente conjugado lineal
- B. Gradiente conjugado no lineal

C. Gradiente bi-conjugado

VI. Introducción al cálculo variacional

A. Problema sin restricciones

VII. Cálculo numérico de derivadas

A. Aproximación por diferencias finitas

VIII. Métodos de Newton prácticos

A. Newton con pasos inexactos
B. Métodos de Newton con búsqueda en línea
C. Técnicas de región de confianza
D. Técnicas de modificación del Hessiano
E. Métodos de Newton de región de confianza

IX. Métodos Quasi-Newton

A. El método Broyden–Fletcher–Goldfarb–Shanno (BFGS)

X. Mínimos cuadrados no lineales

A. Método Gauss-Newton
B. Método Levenberg-Marquardt

XI. Métodos de penalización para problemas no lineales con restricciones

A. Penalización cuadrática

XII. Algoritmos sin derivadas

A. Descenso de simplejo (método de Nelder-Mead)
B. Recocido simulado
C. Algoritmos bio-inspirados

BIBLIOGRAFÍA

1. J. Nocedal and S. J. Wright. Numerical Optimization, Springer Series in Operation Research, 2000.
2. C. T. Kelley. Iterative Methods for Optimization, SIAM Frontiers in Applied Mathematics, no 18.
3. http://www.siam.org/books/textbooks/download.php
4. M. Rivera. Notas del Curso, en http://www.cimat.mx/~mrivera/optimizacion.html

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Cursos presenciales
Exámenes

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Tareas
Resolución de ejercicios

CRITERIOS DE EVALUACIÓN

Prácticas y Tareas	30%
Proyecto Final	10%
Exámenes	60%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC.
Aula virtual.
Asesorías y aula virtual.
Bibliotecas digitales.
Laboratorio de cómputo, herramientas y compiladores para programación de algoritmos y resolución de ejercicios.

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Tópicos Selectos de Ciencias de la Computación

SEMESTRE 2,3 ELECTIVA

CICLO ESCOLAR

21TC01

CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

Proporcionar al estudiante las herramientas teórico-prácticas a nivel de investigación de posgrado para el tema selecto de ciencias de la computación y no está considerado en los temas que forman parte del Plan de Estudios.

CONTENIDO TEMÁTICO

El temario de esta materia se elige por el profesor de acuerdo a los intereses de los alumnos.

BIBLIOGRAFÍA

La bibliografía de esta materia se elige por el profesor de acuerdo a los intereses de los alumnos.

ACTIVIDADES DE APRENDIZAJE
BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases con el profesor

Exposiciones
Resolución de problemas
Exámenes

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Trabajos
Revisiones de artículos
Tema para titulación formalizado

CRITERIOS DE EVALUACIÓN

Esto se determina por el profesor, dependiendo de la materia

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC
Aula virtual
Asesorías y aula virtual
Bibliotecas digitales

DENOMINACIÓN DE LA ASIGNATURA O UNIDAD DE APRENDIZAJE

Ciencia de Datos

SEMESTRE 2, 3 ELECTIVA

CICLO ESCOLAR

21CD01

CLAVE DE LA ASIGNATURA

FINES DE APRENDIZAJE O FORMACIÓN

Mostrar los métodos básicos de aprendizaje supervisado, no supervisado, y métodos de visualización para datos en alta dimensión. Se hará especial énfasis en el uso computacional y aplicaciones en ciencia de datos.

CONTENIDO TEMÁTICO

I. Métodos de visualización y reducción de dimensión
- A. Técnicas básicas de visualización
- B. Métodos de proyección y reducción de dimensión
- C. Métodos basados en componentes principales

II. Métodos de aprendizaje no supervisado
- A. El concepto de disimilaridad
- B. Clustering
  1. clustering jerárquico
  2. clustering basado en algoritmos combinatorios (K-medias y métodos relacionados)
- C. Métodos de Kernel y aplicaciones
  1. Kernel PCA
  2. Clustering espectral
  3. Representación de datos no estructurados

III. Métodos de aprendizaje supervisado
- A. Teoría de decisión estad´ ística´
- B. Clasificación lineal
  1. Análisis discriminante lineal y cuadrático
  2. LDA de rango reducido

C. Regresión logıstica
D. Hiperplanos separadores y el algoritmo perceptron
E. Redes neuronales Feedforward
F. Máquinas de soporte vectorial
G. Regularización y selección de modelos
H. Modelos aditivos y métodos relacionados
    1. Árboles de decisión
    2. Boosting
    3. Random Forest

BIBLIOGRAFÍA

1. Aurélien Gerón. Hands-on Machine Learning with Sci-Learn, Keras and Tensorflw, 2nd edition. O'Reilly Media Inc. 2019
2. Andreas C. Müller and Sarah Guido. Introduction to Machine Learning with Python. O'Reilly Media Inc. 2017.
3. Peter Bruce, Andrew Bruce, and Peter Gedeck. Practical Statistics for Data Scientists. O'Reilly Media Inc. 2020.
4. Graham J. Williams. The Essentials of Data Science, Kowledge Discovery Using R. CRC Press. 2017.

ACTIVIDADES DE APRENDIZAJE BAJO CONDUCCIÓN DE UN ACADÉMICO

Clases Sesiones de ayudantías Laboratorios de cómputo

ACTIVIDADES DE APRENDIZAJE INDEPENDIENTES

Tareas Estudio

CRITERIOS DE EVALUACIÓN

Prácticas y Tareas	30%
Proyecto Final	30%
Exámenes	40%

MODALIDADES TECNOLÓGICAS E INFORMÁTICAS

Presencial con TIC
Aula virtual
Asesorías y aula virtual
Bibliotecas digitales