Arquitectura de Sistemas

Curso Académico: (2020 / 2021) **Fecha de revisión: 08-07-2020**

Departamento asignado a la asignatura: Departamento de Ingeniería Telemática

Coordinador/a: IBAÑEZ ESPIGA, MARIA BLANCA

Tipo: Obligatoria Créditos ECTS : 6.0

Curso : 2 Cuatrimestre : 1

REQUISITOS (ASIGNATURAS O MATERIAS CUYO CONOCIMIENTO SE PRESUPONE)

Programación y Programación de sistemas

OBJETIVOS

El estudiante debe ser capaz de diseñar un sistema software utilizando el lenguaje de programación C. El sistema debe contener estructuras de datos no triviales, gestión dinámica de memoria y utilizar técnicas de ingeniería para traducir un conjunto de restricciones de alto nivel, derivadas de un hipotético escenario industrial, en una aplicación robusta.

El estudiante debe ser capaz de utilizar con soltura las siguientes herramientas utilizadas en entornos industriales: un compilador con las opciones para generar diferentes versiones para depuración y analizar los mensajes que se obtienen mientras se desarrolla una aplicación, un entorno de desarrollo integrado para implementar el sistema software, un compilador cruzado para crear versiones de una aplicación, y herramientas de análisis del comportamiento de memoria en una aplicación.

El estudiante debe ser capaz de: trabajar de forma efectiva en un equipo en la ejecución de un proyecto consistente en el diseño de una aplicación software, generar ideas de forma colaborativa en un equipo y optimizar su rendimiento para cumplir con los requisitos del proyecto, y dividir las tareas de forma efectiva entre los miembros del equipo.

El estudiante debe ser capaz de: aprender de forma autónoma, manipular las diferentes fuentes de información, generar información concisa sobre las tareas conseguidas, manejar el tiempo de trabajo personal, y presentar de forma efectiva los resultados derivados de su trabajo.

DESCRIPCIÓN DE CONTENIDOS: PROGRAMA

El programa está dividido en los siguientes bloques:

1. El lenguaje de programación C
1.1. Estructuras de datos básicas y construcciones de flujo
1.2. Estructura de una aplicación C.
1.3. El pre-procesador, división de ficheros, y creación de ejecutables
1.4. Manipulación de punteros.

2. Gestión de memoria dinámica en C
2.1. Estructuras de datos dinámicas
2.2. Fugas de memoria
2.3. Herramientas para detectar fugas de memoria

3. Arquitectura de la plataforma Linux
3.1. Kernel, procesos y sistema de ficheros.
3.2. Principales bibliotecas
3.3. Concurrencia

4. Diseño del proyecto mediante trabajo en equipo
4.1. Resolución de conflictos
4.2. Desarrollo del proyecto

ACTIVIDADES FORMATIVAS, METODOLOGÍA A UTILIZAR Y RÉGIMEN DE TUTORÍAS

Las actividades utilizadas para verificar las competencias y destrezas en el curso son (seguidas por la referencia a los objetivos del programa):

- Ejercicios sobre los siguientes temas: diseño de las estructuras de datos más apropiadas para una funcionalidad en una aplicación, escribir fragmentos de código para manipular estructuras de datos, lectura/escritura de campos de las estructuras, datos sobre los procesos, etc; calculo de la cantidad de memoria ocupada por diferentes estructuras de datos (OP: a).

- Durante las sesiones de laboratorio se escriben, compilan, enlazan y ejecutan fragmentos de código con las diferentes opciones del compilador para incluir información de depuración, definir constantes, etc. Se analiza la corrección de estos fragmentos de código con el depurador (OP: b).

- Durante las sesiones de laboratorio se escriben fragmentos de código para crear, destruir y manipular estructuras de datos utilizando memoria dinámica. Los estudiantes también dividen la funcionalidad del programa requerido entre múltiples funciones de las que escriben su código (OP: c).

- Durante un período de ocho semanas, los estudiantes se dividen en equipos que deben ejecutar un proyecto consistente en el diseño de una aplicación software con múltiples hitos, entregables y objetivos (OP: d).

- Búsqueda de documentos auxiliares para completar la información que se estudia en un tema. En su informe final, deben mencionar las fuentes de información utilizadas (OP: i).

- Uso de las siguientes herramientas: compilador, entorno de desarrollo integrado, controlador de versiones y emulador en las sesiones de laboratorio (OP: k).

Durante estas actividades la plantilla docente revisa el trabajo de los estudiantes en la clase, supervisa las sesiones de laboratorio, responde a las preguntas en el foro del curso, mantiene una hora de tutoría semanal en el despacho y convoca tutorías con múltiples alumnos cuando lo considera oportuno.

SISTEMA DE EVALUACIÓN

Exámenes (EC) durante el curso (60 % del total por evaluación continua):
- 1 Examen parcial de teoría (20%)
- Proyecto realizado durante el curso (40%)

Examen final (EF)(40%)
- Hay que sacar un mínimo de un 40% de nota final del examen final

Examen Convocatoria extraordinaria
Máxima nota entre opción 1 y opción 2.
Opción 1: 100% examen de la convocatoria extraordinaria
Opción 2: 60% evaluación continua y 40% examen convocatoria extraordinaria. Hay que sacar un mínimo de un 40% de nota final del examen de la convocatoria extraordinaria.

Peso porcentual del Examen Final:	40
Peso porcentual del resto de la evaluación:	60

BIBLIOGRAFÍA BÁSICA

- Steve Oualline: Practical C Programming, Proquest, 1991
- Félix García Carballeira El lenguaje de programación C : diseño e implementación de programas, Prentice Hall.
- Félix García Carballeira Problemas resueltos de programación en lenguaje C, Tomson Paraninfo.

BIBLIOGRAFÍA COMPLEMENTARIA

- Varios autores Maemo Diablo Training Material (http://maemo.org/development/training/), .