Universidad Tecnológica de Pereira

UNIVERSIDAD TECNOLÓGICA DE PEREIRA
FACULTAD DE INGENIERÍAS: ELÉCTRICA, ELECTRÓNICA, FÍSICA Y CIENCIAS DE LA COMPUTACIÓN
PROGRAMA INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

ASIGNATURA:	**ARQUITECTURA CLIENTE/SERVIDOR**
CODIGO:	**IS924**
CREDITOS:	**4**
INTENSIDAD:	**6 horas semanales para 96 horas totales**
REQUISITOS:	**IS823 Comunicaciones II**

JUSTIFICACION

En un mundo donde la computación está cada vez más distribuida y donde Internet es el principal recurso de información y capacidad de procesamiento, se requiere conocer las diferentes técnicas existentes para permitir la interconexión de programas de computador para intercambiar información y compartir cargas de procesamiento utilizando ambientes seguros.

OBJETIVOS

OBJETIVO GENERAL

Desarrollar la capacidad de implementación de sistemas C/S de alto rendimiento, y conocer sus características y posibilidades de aplicación en diferentes áreas de los Sistemas Distribuidos.

OBJETIVOS ESPECIFICOS

1. Identificar la filosofía de las Arquitecturas Cliente Servidor.

2. Realizar un reconocimiento de las técnicas más modernas de programación de sistemas distribuidos robustos, extensos y de alto rendimiento, y demás temas afines.

3. Reconocer los conceptos básicos de seguridad en sistemas distribuidos.

CONTENIDO

1. Talleres instalación LINUX
2. Presentación de la problemática Cliente-Servidor (CS) y su aplicación en Sistemas distribuidos (SD). Arquitectura CS por capas

3. Sockets, comunicaciones TCP/IP, connections, streams
4. Programación IPC (share memory, message queues, pipes)
5. Protocolos (SMTP, HTTP, XML, FTP, SOAP)
6. Servidores iterativos y concurrentes (Procesos, threads y eventos)
7. RPC
8. Sockets, mensajes asincrónicos y prioritarios
9. Agentes
10. Servidores de alto rendimiento (demonios)
11. Sockets, UPD/IP, connectionless
12. Evaluación individual (esquema CS en seudo-C)
13. Encripción y seguridad VS CS
14. CS aplicada a la Web
15. Bases de alto rendimiento (high performance (HP)), conceptos de middleware y grid computing, conceptos de computación intensiva de datos (HSM)
16. Arquitecturas orientadas a SERVICIO, objetos distribuidos (CORBA, y otros)

METODOLOGIA

Explicación de los diferentes conceptos básicos y elaboración de prácticas, las cuales serán desarrolladas en equipos de trabajo.

Las prácticas serán realizadas en C/C++ o Python sobre sistema operacional Linux.
Conocimientos previos de IPC son requeridos.

COMPETENCIAS

COMPETENCIAS TRANSVERSALES / GENÉRICAS:
- Aprendizaje autónomo
- Capacidad de análisis y síntesis
- Capacidad de aplicar los conocimientos a la práctica
- Resolución de problemas
- Trabajo en Equipo
- Comunicación oral

COMPETENCIAS ESPECÍFICAS:
- Cognitivas (Saber):
  - Aprendizaje de Lenguajes de Programación.
  - Habilidades de Programación.
  - Respuesta a tiempos de entrega.
  - Levantamiento de Requerimientos.
  - Análisis y diseño de software
  - Idioma
  - Estimación y programación del trabajo
  - Planificación, organización y estrategia.

- Comunicativas (Expresión oral).

- Actitudinales (Ser):
  - Trabajo en Equipo
  - Calidad
  - Autoestima (Seguridad en el trabajo desarrollado)

TÉCNICAS DOCENTES

Las técnicas docentes que se van a utilizar son:

- Clases de teoría
- Exposiciones sobre trabajos de casos prácticos.
- Tutorías colectivas de teoría
- Clases prácticas
- Corrección de las prácticas
- Tutorías colectivas de prácticas
- Tutorías individualizadas

DESARROLLO Y JUSTIFICACIÓN:

Clases de teoría:

- Se hará una reseña inicial del contenido de cada tema y se indicará su relación con los otros temas.
- Al comenzar la explicación de una sección de un tema, se indicarán las relaciones que posee con otras secciones del mismo tema o de temas diferentes.
- Se explicará detenidamente cada sección de cada tema teórico.

Acerca de las prácticas:

- Las prácticas y tutorías colectivas de prácticas, están sujetas a la disponibilidad de salas con computadores.

- El objetivo principal de las prácticas es establecer un espacio en el que los estudiantes se enfrenten a los diferentes problemas de implementación, al tiempo que presentan los avances realizados en los trabajos propuestos.

- Se ofrecerán las herramientas necesarias de desarrollo, utilizando los lenguajes de programación C/C++ como ejemplo por su simplicidad y potencia.

- Se trabajará la habilidad de los estudiantes para levantar requerimientos al cumplir el papel de cliente en las diferentes prácticas teniendo el estudiante la obligación de indagar el alcance de los proyectos. Se brindarán los espacios necesarios para que este análisis se pueda realizar.

- Se entregarán ejemplos de las funcionalidades básicas de cada proyecto.

- Se podrá corregir y evaluar en presencia del estudiante los trabajos de prácticas que haya realizado.

- Se le indicarán al estudiante los fallos y posibles soluciones alternativas.

Tutorías Grupales:

· Los grupos de estudiantes con el fin de poder organizar y garantizar que la atención sea individual, deberá solicitar con anticipación cita con el profesor.

· Los estudiantes deben utilizar estas tutorías a lo largo de todo el curso y no sólo antes de la fecha del examen o de la entrega de los proyectos.

· El profesor intentará resolver las dudas particulares que pueda tener cada estudiante en relación con los temas de teoría, los trabajos de las exposiciones, las prácticas, etc.

· Aunque las dudas más simples puedan plantearse mediante correo electrónico, es preferible que haya una reunión del profesor y el estudiante para resolver las dudas más complejas.

MECANISMOS DE CONTROL Y SEGUIMIENTO

El profesor podrá comprobar el grado de seguimiento de la asignatura mediante:
· La asistencia a las clases de teoría y prácticas
· Las exposiciones de temas de teoría.
· La corrección de las prácticas.
· Las tutorías personales
· El parcial teórico
· La revisión de entregas parciales de los proyectos.

ORGANIZACIÓN SEMANAL – ARQUITECTURA CLIENTE – SERVIDOR

Semana	Temas	Clases teoricas (Horas)	Tutorías Profesor (Horas)	Tutorías Monitor (Horas)	Práctica (Horas)	Examen (Horas)
1	Presentación del curso, objetivos, contenido, metodología, evaluación. Conceptos básicos: cliente, servidor. Jornada de instalación de Linux.	6	2			
2	Problemática cliente servidor y su aplicación en sistemas distribuidos. Arquitectura c/s por capas.	6	2			
3	Sockets, comunicaciones, TCP/IP. connections, streams	6	2			

4	Programación IPC (share memory, message queues, pipes).	6	2			
5	Protocolos SMTP, HTTP, XML, FTP, SOAP	6	2			
6	Servidores iterativos y concurrentes (procesos, threads y eventos)	4	2			
7	RPC	6	2			
8	Sockets, mensajes asincrónicos y prioritarios.	6	2			
9	Agentes	6	2			
10	* Lunes festivo Servidores de alto rendimiento (demonios)	6	2			
11	Sockets UPD/IP, connectionless	6	2			
12	Evaluación.	4	2			2
13	* Lunes festivo Encripción y segurdad VS CS	6	2			
14	* Lunes festivo CS aplicada a la web	6	2			
15	Bases se alto rendimiento (high performance (HP)), conceptos de Middleware y gris computing, conceptos de computación intensiva de datos (HSM))	6	2			
16	Arquitectura orientada a servicio, objetos distribuidos (CORBA y otros)	6	2			

EVALUACIÓN

Semana	Porcentaje de evaluación	Tipo	Tema cubierto
6 (seis)	30 %	Examen teórico practico	Temas vistos hasta: **Protocolos**
12 (doce)	30%	Examen teórico practico	Temas vistos hasta : **Sockets, UPD/IP, connectionless**
Finales	40%	Proyecto	Proyecto que involucre los temas vistos en la asignatura.

Referencias Bibliográficas y links de interés:

- Computación distribuida : fundamentos y aplicaciones. M. L. Liu. Addison-Wesley, 2004. ISBN 84-7829-066-4
- Thinking in C++, Bruce Eckel
  http://www.mindview.net/Books/TICPP/ThinkingInCPP2e.html
- the C Programming Language, Brian W. Kernighan
- Advanced Programming in the UNIX Environment, W richard stevens
  http://www.kohala.com/start/apue.html

- Client/Server Architecture (McGraw-Hill Computer Communications Series), Alex Berson

- Web services : concept, architectures and applications. Gustavo Alonso et al. Springer, 2004. ISBN 3-540-44008-9

- GNU C
  http://www.gnu.org/software/libc/manual/html_node/