Robot de telemetría

Internet: Arquitectura y Protocolos

2025-2

Introducción

En este primer proyecto de curso se pondrán a prueba sus habilidades para la programación de aplicaciones de red que soporten concurrencia, así como sus competencias para diseñar e implementar protocolos de la capa de aplicación.

En este curso, un protocolo es simplemente el conjunto de reglas para que dos programas se comuniquen. Dichas reglas incluyen los formatos de mensajes y procedimientos que se generan que controlan el intercambio coordinado de información y de servicios entre entidades que se conectan a una red de datos.

Se usará la API de Sockets Berkeley para crear un servidor que reciba peticiones de uno o varios clientes. Tenga en cuenta que este es un proyecto con alta demanda de tiempo, por lo que se recomienda empezar cuanto antes y consultar con el profesor las dudas o dificultades que se presenten en el desarrollo del proyecto.

Conceptos

Sockets y Abstracción

Los desarrolladores de aplicaciones utilizan abstracciones, que son herramientas que ocultan los detalles complejos de un sistema, para simplificar su trabajo. Un **socket** es precisamente una de estas abstracciones. Se puede entender como un punto de conexión o un mecanismo que permite a las aplicaciones "adherirse" a la arquitectura de red y comunicarse con otras aplicaciones a través de ella. De esta manera, una aplicación escribe datos utilizando su socket y otra aplicación, en otra máquina, los lee desde el suyo.

Tipos de Sockets en el Proyecto

Existen diferentes tipos de sockets, pero para este proyecto, se utilizarán dos de la API de Sockets Berkeley:

- **Sockets de flujo (SOCK_STREAM):** Proporcionan una conexión fiable, ordenada y orientada a la conexión. Son ideales para aplicaciones que no pueden permitirse la pérdida o el desorden de datos.
- **Sockets de datagrama (SOCK_DGRAM):** Ofrecen un servicio más rápido y sin conexión. No garantizan la fiabilidad ni el orden de los datos, lo que los hace adecuados para aplicaciones donde la velocidad es más importante que la precisión.

Para este proyecto, debe decidir y justificar, basándose en los requerimientos de su aplicación y en el diseño de su protocolo, cuál de estos tipos de sockets utilizará y en qué escenarios.

Contexto

Un equipo de ingenieros electrónicos ha construido y programado un robot que mide variables atmosféricas, con la intención de facilitar la telemetría de diferentes variables de un entorno. El equipo de desarrollo está interesado en entregar esas variables de forma remota a diferentes usuarios.

Usted ha sido asignado para desarrollar e implementar el protocolo de comunicaciones para que varios usuarios puedan obtener las mediciones en "tiempo real" de las variables que mide el robot y, adicionalmente, enviar comandos al mismo (por ejemplo, movimiento a la izquierda, a la derecha, adelante o atrás).

Requerimientos

1) El robot debe recibir peticiones de los usuarios que quieren recibir información sobre las variables medidas. Mientras que la conexión esté abierta, el robot debe enviar información cada 15 segundos a todos los usuarios conectados. En este sentido, la aplicación en el robot debe mantener una lista actualizada, añadiendo o removiendo usuarios según el estado de conexión.
2) El robot puede recibir comandos para que se mueva en 4 direcciones diferentes (izquierda, derecha, frente, y atrás). En caso de que encuentre un obstáculo, el robot debe responder que no puede moverse en esa dirección.
3) Solamente un usuario "administrador" puede enviar comandos al robot. Este usuario debe ser correctamente identificado en la aplicación del robot. Usted debe garantizar esto incluso si el cliente cambia su dirección IP. Dicho usuario también puede solicitar una lista de usuarios conectados en ese momento.
4) Usted debe diseñar, especificar e implementar del protocolo de la capa de aplicación que permita la lectura de las variables medidas por el robot, el envío de comandos y el control de usuarios. El protocolo debe ser codificado tipo texto. Usted debe describir la especificación del servicio, el vocabulario de mensajes, las reglas de procedimiento y todo lo que usted considere necesario para el buen

funcionamiento del protocolo. Para esto se recomienda considerar la estructura, por ejemplo, de un RFC como el 793 (TCP), que, a grandes rasgos, incluye:

a) **Visión General del Protocolo:** Se define el propósito del protocolo, su modelo de funcionamiento (por ejemplo, cliente-servidor) y la capa de la arquitectura en la que opera.
b) **Especificación del Servicio:** Se describen las primitivas del servicio, que son las operaciones que un usuario del protocolo puede realizar (por ejemplo, GET, MOVE, LIST).
c) **Formato de Mensajes (Encabezado):** Se desglosa la estructura del encabezado del protocolo. En este punto, se describen los campos del encabezado, su tamaño, su significado y los valores posibles (por ejemplo, parámetros solicitados, longitud del movimiento, entre otros).
d) **Reglas de Procedimiento (Flujo de Operación):** Se detallan los algoritmos y las máquinas de estado para el funcionamiento del protocolo.
e) **Ejemplos de Implementación:** Se pueden incluir ejemplos de código o diagramas de secuencia para ilustrar el funcionamiento del protocolo en escenarios específicos.

5) Su protocolo se debe integrar a la arquitectura de red TCP/IP a través de la API de sockets y debe ser implementado en las estaciones cliente y servidor. Usted decide qué tipo de protocolo de transporte es el adecuado para el protocolo de aplicación que está implementando.
6) Su protocolo debe soportar múltiples clientes simultáneos recibiendo información. Para efectos de esta práctica, se permite el uso de hilos para soportar la concurrencia del servidor.
7) Su protocolo debe implementar control de excepciones. Debe establecer procedimientos para el manejo de conexiones fallidas, mensajes con formato incorrecto y demás situaciones que considere.

Detalles de la implementación

- La aplicación de los clientes debe estar escrita en al menos 2 lenguajes diferentes. Unos clientes pueden estar hechos, por ejemplo, en Python y otros en C o Java. El cliente debe tener una interfaz gráfica (puede ser sencilla) que simule la posición del robot en tiempo real y el espacio (con límites bien definidos) por el que se mueve. Así, los clientes que no son “administradores”, pueden ver los movimientos en tiempo real que realiza el robot una vez el administrador haya enviado los comandos. El usuario administrador también puede visualizar la posición del robot y recibirá también los valores de las variables.
- Usted debe simular la generación de al menos 4 variables atmosféricas en el robot. Puede generar valores aleatorios que serán reportados a los usuarios. Los

usuarios, al conectarse, pueden especificar si desean recibir únicamente una medida de una variable, en este caso, el servidor sólo debe reportarle esa variable al usuario. En caso contrario debe reportar todas las medidas.

- La aplicación del servidor sólo puede estar implementado en C. No se debe utilizar una API diferente a la de Berkeley para construir los sockets.
- Su servidor debe ejecutar una funcionalidad de "logging", que muestre en consola las peticiones entrantes y las respuestas enviadas por el servidor. Estas peticiones deben almacenarse también en un archivo de logs.
- Su servidor se debe ejecutar recibiendo los parámetros por consola "puerto" y "archivoDeLogs". El primero corresponde al puerto por el que recibe peticiones y el segundo al archivo donde almacena las peticiones y respuesta. Las peticiones y respuestas deben almacenar el identificador de cada cliente (ip y puerto origen de la petición).
- Se recomienda realizar el proceso de compilación del código del servidor utilizando gcc y la generación de un archivo Makefile.

Evaluación

La calificación del proyecto dependerá de varios componentes. Cada uno de ellos será evaluado de manera presencial, sustentando lo desarrollado de forma oral. En la sustentación el estudiante debe ser capaz de explicar de forma clara, precisa y concisa el funcionamiento de su código y las decisiones de diseño. También debe poder responder preguntas sobre aspectos técnicos, teóricos y de implementación del proyecto. Los aspectos por evaluar comprenden:

1. Programación del Servidor (40%)
    - **Implementación de la API de Sockets (5%):** El servidor utiliza correctamente la API de Sockets Berkeley para crear, enlazar y poner a la escucha un socket.
    - **Manejo de Múltiples Clientes (10%):** El servidor puede aceptar y gestionar múltiples peticiones de clientes de forma concurrente, sin bloquearse
    - **Lógica del Protocolo (25%):** El servidor implementa de forma correcta el protocolo diseñado, procesando los mensajes según las reglas establecidas.

2. Programación de los Clientes (30%)

    - **Conexión y Petición (15%):** Los clientes se conectan correctamente al servidor y construyen las peticiones según el formato del protocolo.
    - **Manejo de Respuestas (15%):** Los clientes procesan y muestran adecuadamente las respuestas del servidor, incluyendo mensajes de éxito y de error.

3. Funcionamiento y Comunicación (20%)

- **Robustez del Protocolo (10%):** El sistema completo (servidor y clientes) funciona de manera estable ante diferentes escenarios (ej. múltiples clientes, peticiones inválidas, desconexiones).
- **Coordinación de Mensajes (10%):** El intercambio de mensajes entre las entidades es coordinado y cumple con las reglas de procedimiento del protocolo.

4. Documentación e Informe (10%)

- **Especificación del Protocolo (10%):** Se describe de forma detallada la especificación del protocolo, incluyendo el formato de los mensajes y las reglas de procedimiento.

Entrega

1. El proyecto debe ser desarrollado en equipos de máximo 3 estudiantes. No debe ser desarrollado de manera individual.
2. Cree un repositorio privado de su proyecto. No comparta su código con otros grupos.
3. El trabajo debe ser entregado antes del 28 de septiembre de 2025 a las 23:59. Usted debe proporcionar el enlace al repositorio en el buzón de interactiva virtual. En el momento de la sustentación se revisará que no haya commits posteriores a esa fecha.
4. La evaluación se apoya en la sustentación del código. El profesor hará preguntas sobre los aspectos a evaluar. Si no son respondidas satisfactoriamente, la nota se verá afectada incluso si el código cumple con los requisitos.