

# Estructura Completa de la Práctica 5

## Archivos Creados - Resumen

### Ejercicios Previos (Sección 4 del Enunciado)

Archivo	Propósito	Ejercicio
Cliente.cpp	Cliente que cuenta vocales con el servidor	Ejercicio 1
Servidor.cpp	Servidor básico (atiende 1 cliente)	Ejercicio 1
ServidorMulticliente.cpp	Servidor con threads (atiende N clientes)	Ejercicio 2

**IMPORTANTE:** Estos archivos corresponden a los ejercicios previos del enunciado que los estudiantes deben completar **ANTES** de hacer la práctica 5.

### Práctica 5 - Sistema Distribuido Principal

Archivo	Propósito	Descripción
ServidorTareas.cpp	Servidor de tareas	Gestiona el multibuffer y distribuye tareas a controladores
ServidorMatriz.cpp	Servidor de matriz	Gestiona acceso concurrente a la matriz de resultados
Controlador.cpp	Proceso controlador	Cliente que procesa tareas (se conecta a ambos servidores)

## Archivos de Soporte

Archivo	Propósito
Makefile	Compilación de todo (práctica 5 + ejercicios)
README.md	Documentación principal
GUIA_EJERCICIOS_PREVIOS.md	Guía detallada de ejercicios 1 y 2
lanzar_sistema.sh	Script para ejecutar sistema completo
tareas.txt	Archivo de ejemplo con tareas

## Archivos de la Práctica 4 (Reutilizados)

Estos archivos ya existían y se reutilizan:

librerias/Monitores/

- BufferTareas.hpp
- BufferTareas.cpp
- GestionResultados.hpp
- GestionResultados.cpp

librerias/MultiBuffer/

- MultiBuffer.hpp
- MultiBuffer.cpp

tarea.hpp

## 💡 Librería de Sockets (Proporcionada)

Socket/

- Socket.hpp
- Socket.cpp

## 📁 Estructura de Directorios Final

practica5/

### EJERCICIOS PREVIOS

- |- Cliente.cpp ← Ejercicio 1
- |- Servidor.cpp ← Ejercicio 1
- |- ServidorMulticliente.cpp ← Ejercicio 2

### 🚀 PRÁCTICA 5 PRINCIPAL

- |- ServidorTareas.cpp
- |- ServidorMatriz.cpp
- |- Controlador.cpp

### DOCUMETACIÓN

- |- README.md
- |- GUIA\_EJERCICIOS\_PREVIOS.md
- |- ESTRUCTURA\_COMPLETA.md (este archivo)

### HERRAMIENTAS

- |- Makefile
- |- lanzar\_sistema.sh
- |- tareas.txt

### ksam DE LA PRÁCTICA 4 (reutilizados)

```

    |   └── tarea.hpp
    |   └── librerias/
    |       ├── Monitores/
    |       |   ├── BufferTareas.hpp
    |       |   ├── BufferTareas.cpp
    |       |   ├── GestionResultados.hpp
    |       |   └── GestionResultados.cpp
    |       └── MultiBuffer/
    |           ├── MultiBuffer.hpp
    |           └── MultiBuffer.cpp
    |
    └── LIBRERÍA DE SOCKETS (proporcionada)
        └── Socket/
            ├── Socket.hpp
            └── Socket.cpp

```

## Correspondencia con el Enunciado

### Sección 4.1 - Ejercicio 1

Archivos: [\(Cliente.cpp\)](#), [\(Servidor.cpp\)](#)

- Cliente acepta IP y puerto como parámetros
- Servidor acepta puerto como parámetro
- Se puede ejecutar en local (localhost)
- Se puede ejecutar en remoto (IP diferente)

### Sección 4.2 - Ejercicio 2

Archivos: [\(ServidorMulticliente.cpp\)](#), [\(Cliente.cpp\)](#) (reutilizado)

- Servidor atiende múltiples clientes
- Crea un thread por cada cliente
- Servidor y cliente aceptan parámetros de IP/puerto

### Sección 5 - Trabajo a Desarrollar

Archivos: [\(ServidorTareas.cpp\)](#), [\(ServidorMatriz.cpp\)](#), [\(Controlador.cpp\)](#)

- Servidor de Tareas gestiona multibuffer
- Servidor de Matriz gestiona resultados
- 10 Controladores se conectan a ambos servidores
- Reutiliza monitores de la Práctica 4
- Protocolo de finalización coordinado

# 🎯 Flujo de Trabajo del Estudiante

## Fase 1: Ejercicios Previos (OBLIGATORIO)

1. Leer `GUIA_EJERCICIOS_PREVIOS.md`
2. Compilar ejercicios: `make ejercicios`
3. Completar Ejercicio 1: `Servidor.cpp` + `Cliente.cpp`
4. Completar Ejercicio 2: `ServidorMulticliente.cpp` + `Cliente.cpp`
5. Probar en local y en red

## Fase 2: Práctica 5

1. Leer `README.md`
  2. Compilar práctica 5: `make`
  3. Probar en local: `./lanzar_sistema.sh`
  4. Entender el código de los 3 componentes
  5. Probar en modo distribuido
- 

## 🔍 Diferencias Clave Entre Archivos

### Cliente.cpp vs Controlador.cpp

Aspecto	Cliente.cpp	Controlador.cpp
Propósito	Ejercicio previo - contar vocales	Práctica 5 - procesar tareas
Conexiones	1 servidor	2 servidores (tareas + matriz)
Protocolo	Enviar frase → Recibir count	Solicitar tarea → Procesar → Enviar resultado
Entrada	Usuario (stdin)	Automática (servidor de tareas)

### Servidor.cpp vs ServidorMulticliente.cpp vs ServidorTareas.cpp

Aspecto	Servidor.cpp	ServidorMulticliente.cpp	ServidorTareas.cpp
Clientes	1 cliente	N clientes	10 controladores (fijo)
Threads	No usa threads	1 thread por cliente	1 thread representante + 1 master
Funcionalidad	Contar vocales	Contar vocales	Distribuir tareas
Sincronización	No necesita	Mutex en socket	Monitor BufferTareas
Duración	Hasta END OF SERVICE	Timeout 60s	Hasta procesar todas las tareas



## Comparativa de Complejidad

## Ejercicio 1 (Cliente + Servidor)

- **Líneas de código:** ~150 líneas total
- **Conceptos:** Sockets básicos, Send/Receive
- **Dificultad:** ★☆☆☆☆

## Ejercicio 2 (Servidor Multicliente)

- **Líneas de código:** ~200 líneas
- **Conceptos:** Threads, vector<thread>, concurrencia básica
- **Dificultad:** ★★☆☆☆

## Práctica 5 (Sistema Completo)

- **Líneas de código:** ~600 líneas + monitores de P4
- **Conceptos:**
  - Múltiples servidores
  - Múltiples conexiones por cliente
  - Monitores y sincronización compleja
  - Protocolo de finalización distribuido
  - Round-robin de distribución
- **Dificultad:** ★★☆☆☆

---

## Comandos de Compilación

```
bash
```

```
# Compilar solo ejercicios previos  
make ejercicios
```

```
# Compilar solo práctica 5  
make
```

```
# Compilar todo  
make all  
make ejercicios
```

```
# Limpiar todo  
make clean
```

# Comandos de Ejecución Rápida

## Ejercicio 1

```
bash
```

# Terminal 1

./Servidor 3000

# Terminal 2

./Cliente localhost 3000

## Ejercicio 2

```
bash
```

# Terminal 1

./ServidorMulticliente 3000

# Terminales 2, 3, 4...

./Cliente localhost 3000

## Práctica 5

```
bash
```

# Automático

./lanzar\_sistema.sh

# Manual - 12 terminales

./ServidorTareas 3000 # Terminal 1

./ServidorMatriz 3001 # Terminal 2

./Controlador localhost 3000 localhost 3001 # Terminales 3-12 (10 veces)

## Checklist de Entrega

- Cliente.cpp (Ejercicio 1)
- Servidor.cpp (Ejercicio 1)
- ServidorMulticliente.cpp (Ejercicio 2)
- ServidorTareas.cpp (Práctica 5)
- ServidorMatriz.cpp (Práctica 5)
- Controlador.cpp (Práctica 5)
- Makefile (compila todo)
- README.md (documentación)

- GUIA\_EJERCICIOS\_PREVIOS.md (guía de ejercicios)
  - lanzar\_sistema.sh (script de ejecución)
  - tareas.txt (archivo de prueba)
  - Carpeta Socket/ (librería)
  - Carpeta librerias/ (monitores de P4)
  - Memoria/informe de la práctica
- 

## Notas Finales

1. **Los ejercicios previos son OBLIGATORIOS** antes de la práctica 5
  2. **No se deben modificar** los archivos de Socket/ (proporcionados)
  3. **Se reutilizan** los monitores de la Práctica 4 sin modificación
  4. **La práctica 5** es una extensión distribuida de la Práctica 4
  5. **Todos los archivos** están documentados con comentarios
- 

## Objetivos de Aprendizaje

Al completar esta práctica, el estudiante habrá aprendido:

- Programación con sockets TCP en C++
  - Modelo cliente-servidor básico
  - Servidor multicliente con threads
  - Sistemas distribuidos multi-servidor
  - Sincronización con monitores en sistemas distribuidos
  - Protocolos de comunicación personalizados
  - Gestión de conexiones múltiples
  - Coordinación de finalización en sistemas distribuidos
- 

**Fecha de creación:** Diciembre 2025

**Versión:** 1.0

**Autor:** Práctica 5 PSCD - Universidad de Zaragoza