ingenieur
wissenschaften
htw saar

Multiplexing in C-select()

Ein Vergleich mit modern C++-Thread

ingenieur
wissenschaften
htw saar

→

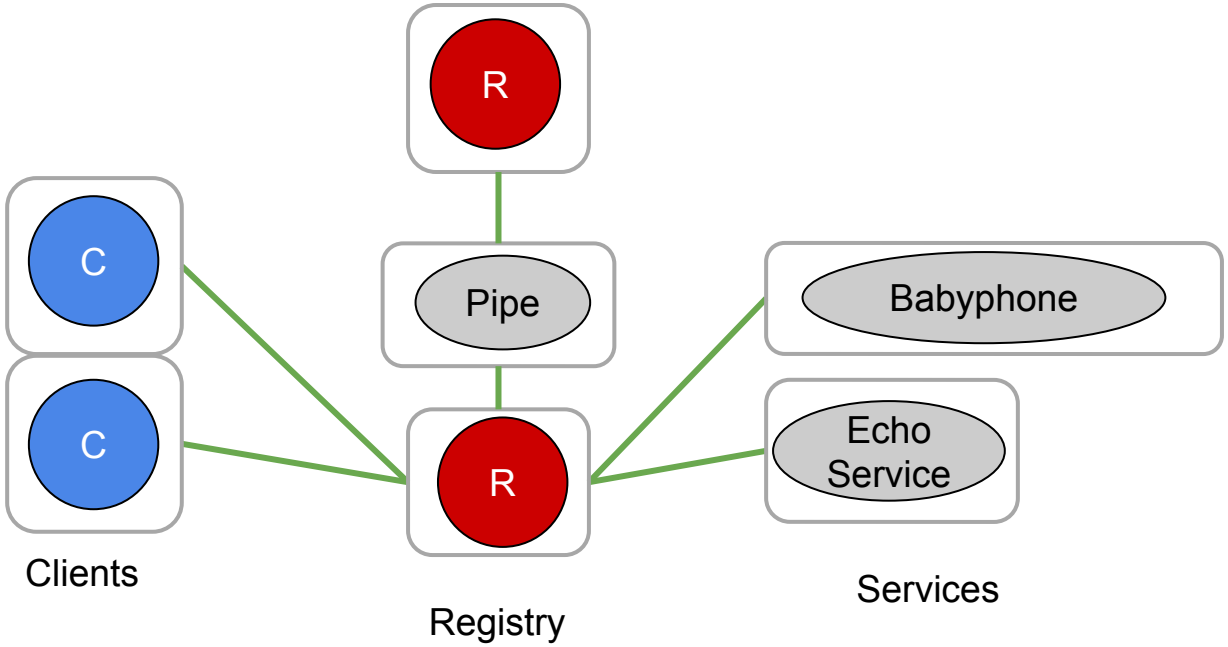
Inhaltsverzeichnis:

Anforderungen

Veranschaulichung der Unterschiede folgender Programmierkonzepte:

- Verwendung von synchronem Multiplexing mit select() in C
- Verwendung von asynchronen Threads in C++ nach dem modern C++11 Standard

Beispielanwendung die verschiedene Deskriptoren verwendet:



Deskriptoren

- Zeiger auf Dateien
  - Einfache Integer Value

- Nach dem Linux-Prinzip : „Everything is a File“
  - Netzwerk deskriptoren
  - Datei deskriptoren
  - … etc.

select() with C

- Seit 1984 standard
- Nachfolger ist poll()

FD_ZERO(*set)	Löscht den Inhalt des fd_set's
FD_SET(s, *set)	Fügt den Socket s dem fd_set hinzu
FD_CLR(s, *set)	Entfernt den Socket s aus dem fd_set
FD_ISSET(s, *set)	Prüft ob der Socket s im fd_set vorhanden ist

threads() with C++

- Seit C++11 ISO Standard (2011)
  - https://www.iso.org/standard/50372.html neuster ~185€
  - Header <thread>
  - Vorgänger: POSIX threads <pthread.h>

→

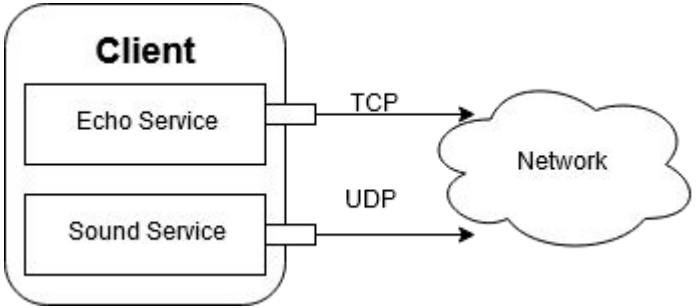
Inhaltsverzeichnis:

Konzept:

- Server hat Services die benutzt werden können
  - **Echo-Service (tcp)**
  - **Named-Pipe-Service (ipc)**
  - **Sound-Service (udp)**
- Client der die Services des Servers in Anspruch nimmt
- Die Verwaltung der ein und ausgänge aller Services sind zu handeln mit:
  - **C : select()**
  - **C++ : thread()**

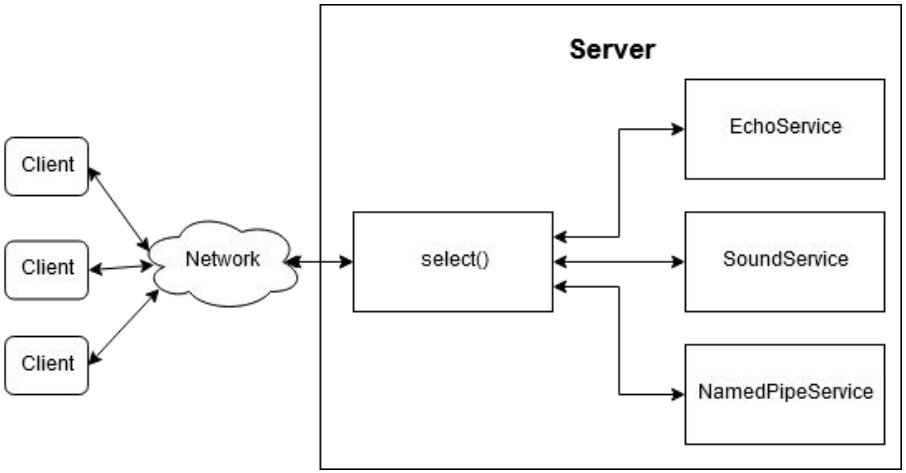
→ Client:

- Funktionen:
  - Echo Service
  - Sound Service



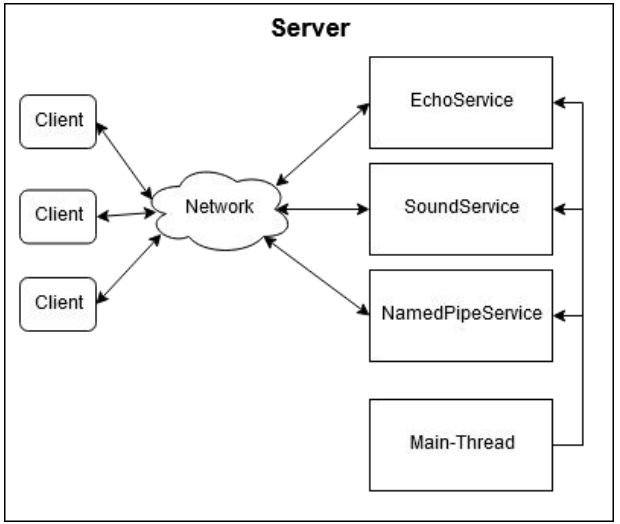
Server select():

- Listener werden initialisiert und dem FD_SET hinzugefügt
- Bei I/O-Operation an einem Deskriptor im FD Set, wird die Prozessausführung auf die jeweilige Funktion verzweigt.
- Pro Anfrage auf den Echo Service werden entsprechend viele Worker-Deskriptoren dem Set hinzugefügt



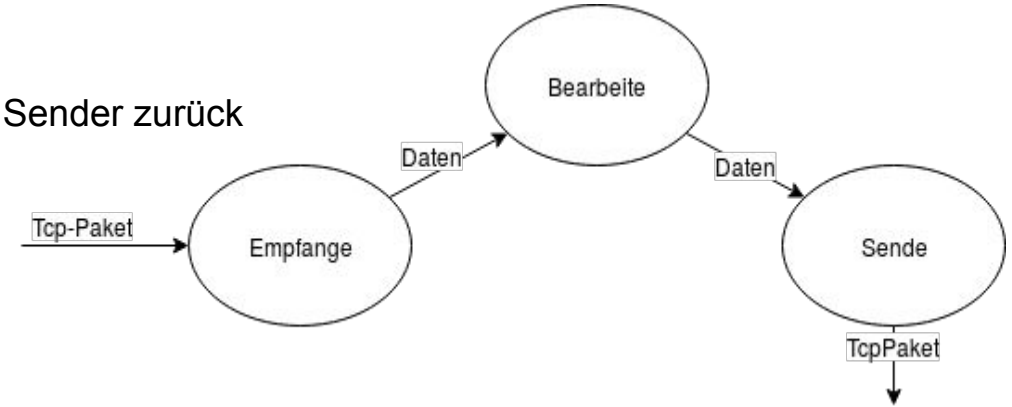
Server threads():

- Initialisierung der Service Threads mit jeweiligen Deskriptoren
- Ein Listen-Thread Pro Service
- Pro Anfrage auf den Echo Service werden entsprechend viele Worker-Threads gestartet
- Main Thread verwaltet die anderen Threads



Echo Service:

- Akzeptiert Verbindungsanfragen
- Empfängt TCP-Datenpakete
- Bearbeitet Datenpackete
- Sendet Datenpakete an den Sender zurück

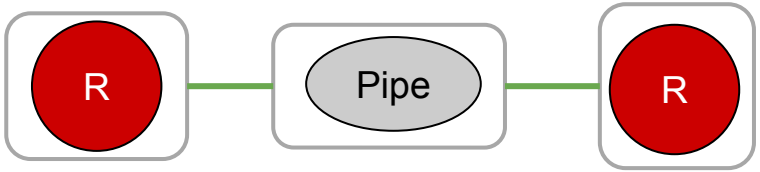


Sound Service:

- Empfängt UDP-Datenpakete
- Konfiguriert die Soundkarte
- Nimmt Geräusche vom Mikrofon auf
- Gibt Geräusche auf den Lautsprechern aus

Named Pipe Service:

- Konfiguriert eine Named Pipe
- Liest von der Named Pipe
- Schreibt auf die Named Pipe



ingenieur
wissenschaften
htw saar

→

Inhaltsverzeichnis:

ingenieur
wissenschaften
htw saar

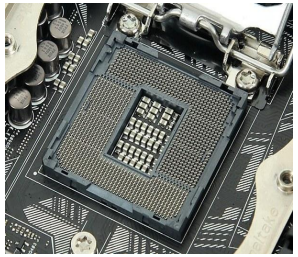
Genutzte Technologien:

GitHub

THE C PROGRAMMING LANGUAGE

C++

LaTeX



Auszug Select-Server:

```
while (1) {
        //Re-/initializing the FS_Set because of modifing through System Calls
        FD_ZERO(&fdset);
        [...]
        FD_CLR(STDIN_FILENO, &fdset);
        [...]
        FD_SET(STDIN_FILENO, &fdset);
        [...]
        if ((err = select(FD_SETSIZE, &fdset, NULL, NULL, NULL) )< 0) {
        [...]
        }
        //Handles STDIN-I/O
        if(FD_ISSET(STDIN_FILENO, &fdset)){
        [...]
        //Handles Sound Service
        }else if(FD_ISSET(*udpListener, &fdset)){
                        handleSoundService(*udpListener, pcmHandle);
        //Handles Pipe Service
        }else if(FD_ISSET(pipeListener, &fdset)){
                        handleNamedPipeServiceRead(&pipeListener, path);
        //Handles TCP-Connection wishs
        }else if (FD_ISSET(*tcpListener, &fdset)) {
                for (int i = 0; i < MAX_WORKER; i++) {
                        if (workers[i] == -1) {
                                [...]
                        }
                }
        }
        //Some I\O happens at a worker-socket descriptors
        else {
                for (int i = 0; i < MAX_WORKER; i++) {
                        if (FD_ISSET(workers[i], &fdset)) {
                                [...]
                        }
                }
        }
```

Auszug Thread-Server:

```
thread tcpListenThread(executeTcpThread, *tcpListener);
thread udpListenerThread(executeUdpThread, *udpListener);
thread pipeThread(executePipeThreadRead, ref(pipeListener),path);

while(true){
        command = dialog();
        if(command == STOP){
                unique_lock<mutex> lock(downMutex);
                cout << "Triggered server command STOP."<< endl;
                cout << "Server will now shutdown." << endl;
                down = true;
                lock.unlock();
                break;
        }else if(command == PIPES){
                unique_lock<mutex> lock(pipeMutex);
                writeToPipe = true;
                char* message = (char*)calloc(MAX_BUFF_SIZE, sizeof(char));
                cout <<"\n Write to Pipe ~> "; cin >> message;
                handleNamedPipeServiceWrite(-1, &pipeListener, path, message, strlen(message));
                writeToPipe = false;
                free(message);
                lock.unlock();
        }else if(command == CANCEL){
                unique_lock<mutex> lock(closeMutex);
                kill = getInt("\nFile descriptor Id: ");
                connectionClose = true;
                lock.unlock();
        }
}

tcpListenThread.join();
udpListenerThread.join();
pipeThread.join();
```

→

Aufbau Udp-Thread():

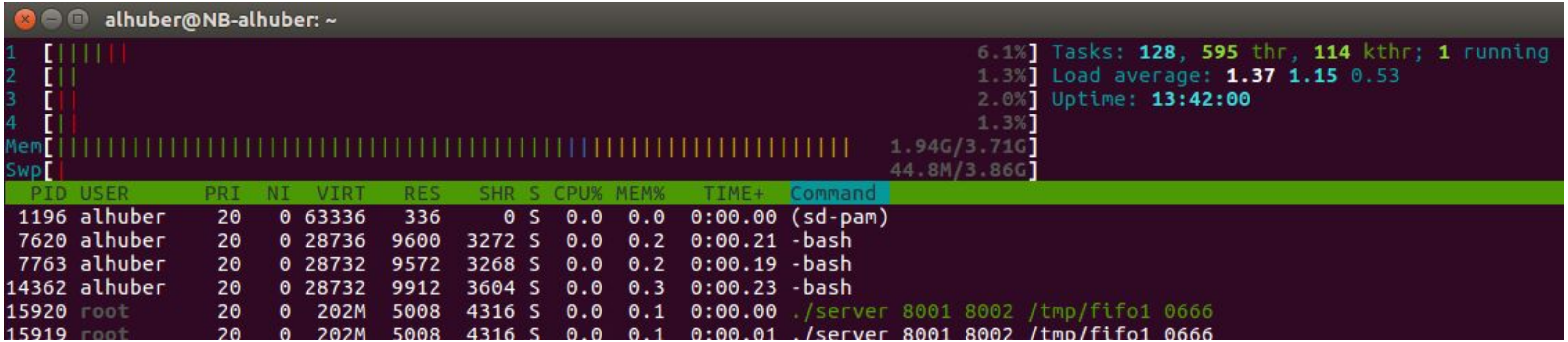
```
1 void executeUdpThread(int udpListener){
2         timer stopwatch;
3         while(!down){
4                 handleSoundService(udpListener, pcmHandle);
5         }
6         closeSoundService(pcmHandle);
7         cout << "Udp Thread exits" << endl;
8     timerUdpThread=stopwatch.elapsed();
9 }
```

→

Inhaltsverzeichnis:

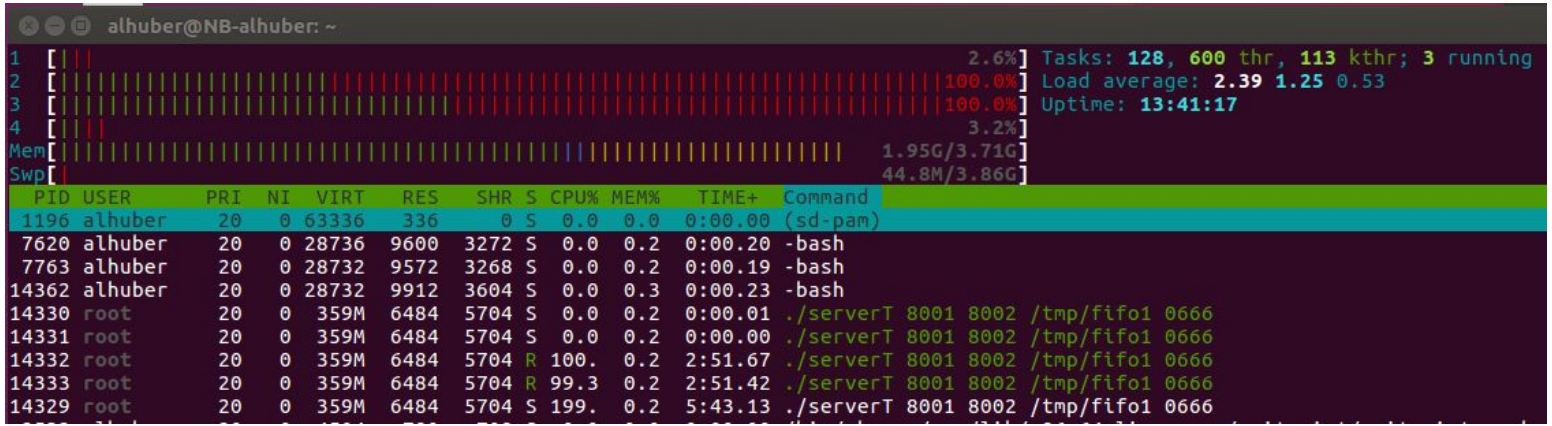
→

Performance Prozess Multiplexing:



ingenieur
wissenschaften
htw saar

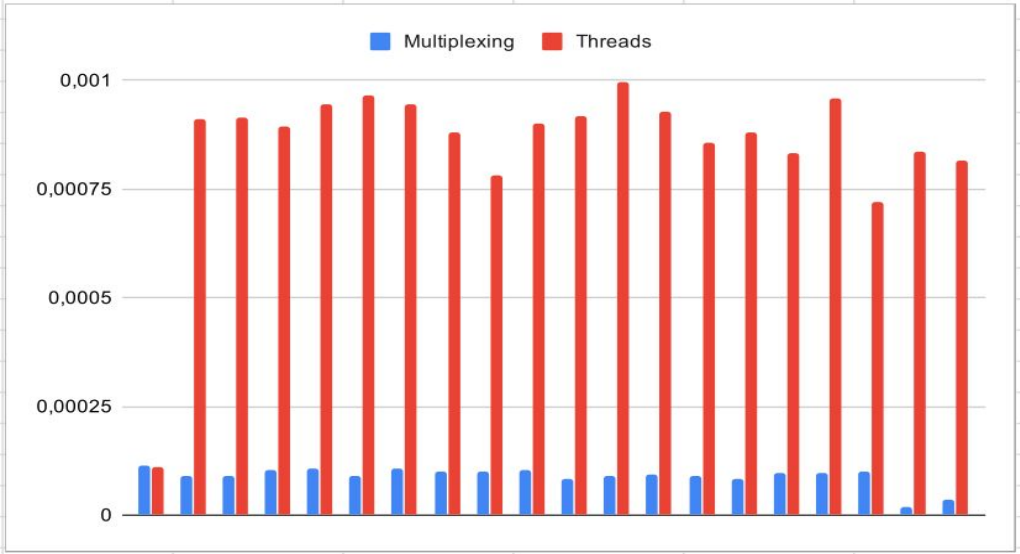
Performance Prozess Threading:



ingenieur
wissenschaften
htw saar

Performance Echo-Anfragen:

Threads	Multiplexing	
0,000109	0,000114	
0,000912	0,000091	
0,000916	0,00009	
0,000895	0,000104	
0,000945	0,000106	
0,000964	0,00009	
0,000945	0,000108	
0,000879	0,0001	
0,000781	0,0001	
0,000901	0,000104	
0,000918	0,000084	
0,000997	0,000091	
0,000927	0,000092	
0,000856	0,00009	
0,000881	0,000084	
0,000833	0,000098	
0,00096	0,000098	
0,000719	0,0001	
0,000835	0,00002	
0,000817	0,000037	
0,0008495	0,00009005	**Mittelwert**
850 Mikrosekunden	90 Mikrosekunden	Faktor 9,4



ingenieur
wissenschaften
htw saar

→

Inhaltsverzeichnis:

ingenieur
wissenschaften
htw saar

DEMO

ingenieur
wissenschaften
htw saar

Was ist noch zu erledigen?

- Umsetzung für Windows
- Mögliche Performanceoptimierung

ingenieur
wissenschaften
htw saar

Vielen Dank!
Fragen?