Sistemas Distribuidos

Práctica RDIR

Objetivo

- Diseño de servicio con y sin estado
- Uso de sockets y RPC de Sun/ONC
- Tres versiones:
  - Servicio RDIR con estado basado en sockets
  - Servicio RDIR sin estado basado en sockets
  - Servicio RDIR sin estado basado en RPC de Sun/ONC
- Interfaz de servicio:
  - r_opendir
  - r_readdir
  - r_closedir

Arquitectura de servicio

- En el cliente:
  - Aplicaciones cliente (rls y rls_mix): ya programadas
  - Biblioteca de servicio (**dirops**): debe desarrollarse
- En el servidor:
  - Código de servicio (**rdird**): debe desarrollarse
- Sobre las comunicaciones:
  - Basadas en TCP
  - Se puede usar 1 petición o múltiples por conexión
    - Si múltiples, sin estado y caída de servidor, cliente debe reconectarse
  - Debe asumirse un sistema heterogéneo

Versiones basadas en sockets

- Servicio concurrente (*threads* o procesos)
  - Cuidado con procesos y 1 conexión/petición
- Codificación de mensajes a criterio del desarrollador
- Ejecución del servidor:

  rdird n_puerto
- Ejecución del cliente:

  export SERVIDOR=máquina
  export PUERTO=n_puerto
  rls dir1 dir2
  rls_mix dir1 dir2

Versión con estado y sockets

- Estrategia que se debe usar:
  - *r_opendir* retorna valor devuelto por *opendir* remoto
  - *r_readdir/r_closedir* envían ese valor que se usa en *readdir/closedir*
- Servidor no debe caer ante mensajes con contenido erróneo
  - P.ej. *r_readdir/r_closedir* no envían valor devuelto por *opendir* previo
  - Cuidado en Linux *readdir/closedir* con parámetro erróneo → SEGV

Versión sin estado y sockets

- Diseño de peticiones autocontenidas
  - Cliente debe recordar la posición del directorio donde se encuentra
  - 2 alternativas (al menos):
    - Uso de *seekdir/telldir* en servidor: OK en Linux pero no válido POSIX
    - Leer *N* entradas y quedarse con la que interesa
- Pruebas de re-arranque del servidor en mitad del servicio
  - Si cliente mantiene conexión → debe reconectarse ante error

Versión basada en RPC

- Especificación de IDL a criterio del desarrollador
- Ejecución del servidor:

  rdird
- Ejecución del cliente:

  export SERVIDOR=máquina
  rls dir1 dir2