Informazio-Sistemen Arkitektura

Telekomunikazio Teknologiaren Ingeniaritzako Gradua. 3. Maila.
Proba praktikoa

2016ko maiatzaren 18a — Iraupen osoa: 2½ h.

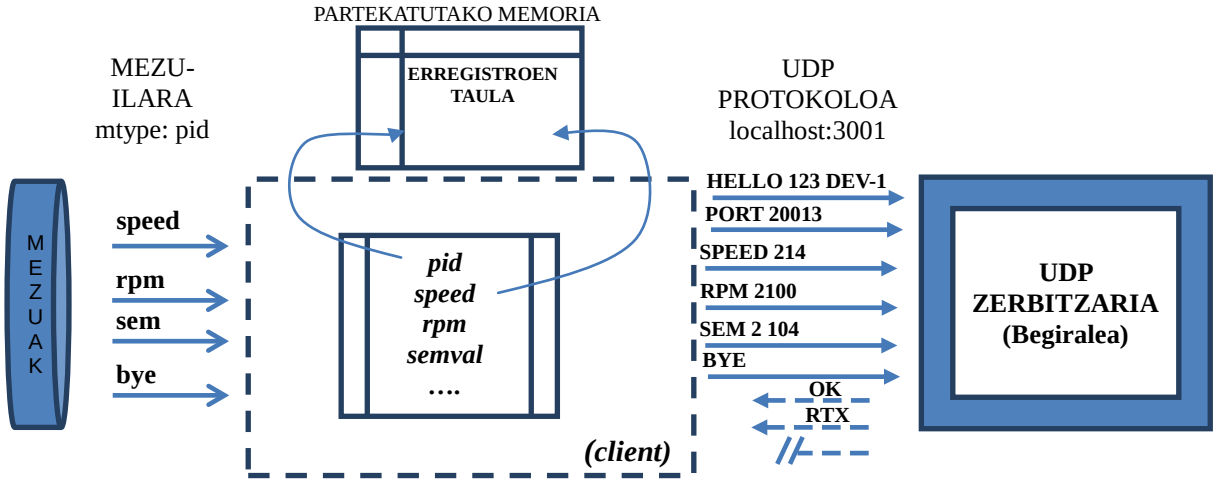
Guztira 16 gako daude. Gakoen erdia jaso behar dira proba gainditzeko

Proba praktikoaren azalpena

Begiraleak UDP zerbitzari moduan lan egingo du 3001 atakan eta bertara konektatuko dira urrutiko gailuak beren kontroleko aldagaien egoerari berriz informatzeko: abiadura (speed), minutuko birak (rpm) eta semaforoen egora (sem).

Ariketa hau modu inkrementalean garatuko da, pausu bakoitzean egindakoari funtzionalitate berriak gehituz eta atzeranzko bateragarritasuna mantenduz. Hau da, 2. ariketa 1. ariketaren hobekuntza izango da. Eta horrela, 3. ariketan funtzionalitate guztiak (gako guztiak) bildu arte.

Sistemaren arkitektura ondoko irudikoa da:



Erregistroen taula partekatutako memoriako segmentu batean dagoen array bat da. Erregistroek sisteman identifikatutako gailu bakoitzaren egoera gordetzen dute. client programa HELLO eta PORT komandoen bidez identifikatu beharko da UDP zerbitzarian eta ondoren, mezu-ilaratik etorriko diren mezuen zain geratuko da (bere pid zenbakia kanal bezala erabiliz mezuak iragazteko) gailuen informazioa lortzeko (*speed*, *rpm* eta *sem*). Jasotako mezuetan datorren informazioa UDP zerbitzariari bidaliko dio gailuen parametroen berri emateko. Parametro hauek gailuari dagokion erregistro egokian idatzi beharko dira partekatutako memorian.

Saio desberdinak ezarri daitezke konkurrenteki terminal desberdinetatik, baina horretarako gailu bakoitza izen desberdin batekin konektatu beharko da. Adibidez, terminal batean $./client device1 eta bestean $./client device2 jarriz. Zerbitzariak saioak etengo ditu epe luze batean (15 segundo) informaziorik jaso ezean.

Baliabide guztiek, partekatutako memoriak, semaforoek eta mezu-ilarak ikaslearen NAN zenbakia erabiliko dute hamaseitarrean eta hizkirik gabe long formatuan (adib: NAN:11234567G eta gakoa 0x11234567L) bere burua identifikatzeko.

1. Ariketa. UDP oinarrizko komunikazioa

Lehen ariketa honetan oinarriko UDP komunikazioa ezarri behar da, identifikatuz, datu bat(zuk) bidaliz, eta deskonektatuz. Zerbitzaria localhost:3001 atakan adi egongo da.

Komunikazio protokolo hau UDP datagramen trukean oinarritzen da eta irakurgarriak diren testuak garbian igortzen ditu. Hasierako komandoak honakoak dira:

1.- HELLO pid Izena (Adib: HELLO 2134 DEVICE-1)

2.- PORT konexioaren jatorrizko ataka (Adib: PORT 32456)

3.- BYE

Komando hauei beti erantzungo zaie OK eta iruzkin batekin emaitzaren balorazioaz. BYE salbuespena da; honek ez du erantzunik itxarongo eta programa berehala amaituko du.

- OK iruzkina // Iruzkinean komandoari buruzko informazio etor daiteke

1 gakoa UDP konexioa HELLO mezu-formatu egokiaz lortutakoan jasoko da. Begiraleak HELLOan etorri den pid zenbakia duen prozesuak bizirik diraela ikusitakoan gailua erregistratuko du eta 2 gakoa erakutsiko digu. PORT komandoak bidali duen ataka jatorrizko atakarekin bat badator, 3 gakoa bistaratuko da. Ataka hori gailuaren jarraipena egiteko erabiliko da aurrerantzean. 4 gakoa BYE komando bidali eta berehala saioa eteten irtenez gero agertuko da.

Oharra: *getsockname* funtzioa erabili socketari jatorrizko ataka esleitzeko (informazio gehiagorako $man).

2. Ariketa. UDP komunikazioa OSOA errorerik gabe.

Ariketa honetan dauden komando guztiak erabiliko ditugu komunikazio oso bat egiteko. Saio bakoitza HELLO komando batekin hasiko da, segidan beste PORT datu egokiekin bidaliz.

Ondoren mezu-ilaratik egoera-parametroak jasotzeko zain geratuko da gure programa. Gailuen egoerari buruzko informazioa irakurtzeko prozesuaren pid-arekin bat datozen mezuak bakarrik jasoko ditugu. Ilaran jasotako mezuek ondoko formatua dute:

<Byte char motakoa><parametroak komandoaren arabera >

Ilarako mezu-motak eta formatuak honakoak dira:

- SPEED: (‘2’)(<abiadura **ascci** formatuan>) Adib: 2<324>
- RPM: (‘3’) (RPM balioa **int** bitarrean) Adib: 3 127
- SEM: (‘4’) (semaforo zenbakia **int** bitarrean) Adib: 4 2
- BYE komandoa: (‘5’) Adib: 5

Mezu hauek jasotakoan, UDP zerbitzariari igorriko zaizkio komando egokiak. Ondokoak dira komando horiek:

- SPEED (abiadura **ascci** formatuan) Adib: SPEED 234
- RPM (birak **ascci** formatuan) Adib: RPM 3455
- SEM (zenbakia **ascci**) (balioa **ascci**) Adib: SEM 1 109
- BYE

5 gakoa SPEED komando edo RPM komando egokiekin lortuko da. 6 gakoa biak betez gero. 7 gakoa 4 semaforo dituen arraitik zenbakiari dagokion semaforoaren balioa zuzena bada. 8 gakoa konkurrenteki bi saio mantentzen badira.

Partekatutako memoriaren (SIZE:1024) 0 posizioan 4 erregistrodun taula bat (array bat) gordeta dago (*struct st_data*) egitura duena. Ikasleak “name” eremua erabiliz 4 erregistroetako edozeinetan saio aktiboa aurkitzen badu, eta bertan pid, rpm eta speed datuak gaurkotzen

baditu, gako gehiago lortu daitezke. Speed edo rpm gaurkotuz gero 9 gakoa. Biekin 10 gakoa. Kasu honetan, aukeratutako semaforoaren balioa irakurtzeko 0 semaforoa (arraiko lehena) erabiltzen badugu atal kritikoa egiteko, 11 eta 7 gakoak lortuko dira. 12 gakoa bi saio konkurrenteri dagozkien pid-ak gaurkotzen badira ($./client device1 eta $./client device2 terminal desberdinetan), bakoitza bere erregistroan.

3. Ariketa. UDP Saio OSOA galerekin

UDP komunikazio batean datagramak galtzeko aukera dago. Baita ere birtransmititzeko premia erantzunik ezean tenporizadoreak iraungitzen badira. Ikasleak prozedura hauek inplementatuko ditu protokoloan. Ariketa honetan Begiraleak datagramen galerak edo/eta birtransmititzeko eskaerak simulatuko ditu ausaz bezeroak protokoloa nola inplementatzen duen aztertzeko.
Bezeroaren komando baten aurrean zerbitzarik modu desberdinetan joka dezake:

- OK iruzkina //komandoa zuzen jaso denean
- RTX iruzkina //azken komandoa birtransmititzeko eskaera
- Erantzunik ez //handik **4 segundura** bezeroak berriz errepikatu mezua

Begiraleak saio bakoitzari birtransmitisio (RTX) eta datagrama-galeren prozedurak simulatuko ditu. Bezeroak ondo erantzuten badio lehenengo saioko birtransmisio (RTX) eskaerei 13 gakoa jasoko du. Saio konkurrenteetan eginez gero, 15 gakoa. Tenporizazio bidez birtransmititzen badu, 14 gakoa eta saio kokurrenteetan bada, 16 gakoa.

Oharra:
Tenporizazio bidezko birtransmisio prozedura inplementatzeko seinale tenporizatu bat erabiltzea proposatzen da (alarm(4)) irakurketa blokeagarri bat eteteko. Sistemaren azken inplimentazioaren zenbait dei ez dira ondo eteten irakurketa blokeagarrietan maskaratze-funtzioak egin ezean. Arazo hau ekiditeko ohiko signal()-en ordez signal_EINTR() funtzioa erabili. Horrela irakurketa blokeagarriak eten ahal izango ditugu denbora agortutakoan.

4. Ariketa. Saioak memorian sortu.

Gerta daiteke UDP zerbitzaria erabilgarri ez egotea, kasu honetan sistemaren gailuek beren saioak sortu ditzakete partekatutako memoria-segmentuko erregistroetan. Memoriaren 0 posizioan 4 erregistro (*struct st_data* motakoak) dituen taula bat dago sistemaren saioak gordetzeko. Arraiko 3. semaforoa (laugarren balioa) saioen taulara atzipen esklusiboa lortzeko erabiliko da, horrela erregistroak modu seguruan idatzeko.

Ariketa hau aurreko ariketen alternatiba bezala pentsatuta dago. Lehenengo hiru ariketak eginez gako guztiak lor daitezke. Ikasleak UDP protokoloaz arazoak baditu, ariketa alternatibo hau eginez 8 gako desberdin lor ditzake (saioaren hasierakoak eta memorian datuak gaurkotzearenak). Zenbait gakok saio konkurrenteak eskatuko dituzte.

Bezeroak libre dagoen erregistro bati alta eman beharko dio. Horretarako bere egoera aldatu beharko dio (ST_FREE egoeratik) 1-era pasetzeko (ST_PID egoerara) eta erregistroan pid eta name eremuak betez. Name eremua 16 byte luze den arren, ahal den neurrian 8 karaktere bakarrik erabili. Begiraleak saio berriak dauden aztertuko du eta dena ondo badago, mezu-ilarara gailuei dagozkien mezuak bidaliko ditu, bezeroak prozesatu ditzan eta memorian behar den bezala idatzi ditzan.

Informazio-Sistemen Arkitektura

Telekomunikazio Teknologiaren Ingeniaritzako Gradua. 3. Maila.
Proba praktikoa

2016ko maiatzaren 18a Iraupen osoa: 2½ h.

Guztira 16 gako daude. Gakoen erdia jaso behar dira proba gainditzeko

Izena:……
Taldea:……...…… NAN:…….
Gela:……. Ilara:…….. Zutabea:….....SINADURA:

1 Ariketa

1 Gakoa:	2 Gakoa:	3 Gakoa:	4 Gakoa:

2 Ariketa

5 Gakoa:	6 Gakoa:	7 Gakoa:	8 Gakoa:
9 Gakoa:	10 Gakoa:	11 Gakoa:	12 Gakoa:

3 Ariketa

13 Gakoa:	14 Gakoa:	15 Gakoa:	16 Gakoa:

4 Ariketa

Ariketa honek aurreko 3 ariketetan lortu daitezkeen 16 gakoetatik zenbait gako lortzen ahalbidetzen du partekatutako memoria ondo erabiliz gero. 2,4,5,6,9,10 gakoak lortzen dira saio on batekin, eta 8 eta 12 gakoak saio konkurrenteekin.

Erreferentziarako client-ref.c erabili arikitetarako zenbait datuen espezifikazioekin arazorik ez izateko. Hemen doa memoriako taularako erregistroen egitura saioetarako:

```
#define ST_FREE 0
#define ST_PID 1
#define ST_DATA 2
#define LEN_NAME 16     //ahal den neurriak 8 byteko izenak erabili
struct st_data {
        int state;      // State of register
        char name[LEN_NAME];    // Name of device
        int speed;      // abiadura
        int rpm;        // bira minutuko
        int port;       // Original port
        int sem;        // Sem number
        int semval;     // Sem value
        pid_t pid;      // Process identifier
};
```