Machine Problem II: Working with Processes
CSCE 313-504

Derek Heidtke

October 11, 2015

Objective

The purpose of this assignment is complete the foundation of an ICU patient monitoring system. This includes: setting up the initialization of the client and server processes, testing the request responses from the dataserver, and measuring the request-reply delay time.

How to Run

Compile by navigating to the working directory and running the command:

```
make all
```

Then, to start the client process, by running:

```
./client
```

The dataserver process will automatically be forked from the client process.

Results & Discussion

To perform a request, the channel is established (which involves the creation of two named pipes), then a request is sent to the dataserver process via the send_request() member function (through one of the pipes). Next, the request is processed by the server and an appropriate response is sent back to the client (also though the created pipes). This sequence of actions takes a relatively long amount of time, when compared to a normal function call.

In order to learn the limitations of the system we are developing, it is necessary to measure how long a typical request takes to be fulfilled. This was done with the printtimediff() function from machine problem I. To have something to compare against, a function, fake_request() which takes a C-string and returns the same C-string, was created.

The program was run multiple times, and the average duration of a request and of the fake_request() function are shown in the following table:

Sample Size = 30	request[μs]	fake_request()[μs]	request time/fake_request() time
Avgerage	174.3	9.4	20.2
Std. Dev.	48.9	3.9	5.6

Table 1: Average duration of a request and a fake_request() function call.

Although the absolute measurements (i.e., request time and fake_request() time) will most likely vary from platform to platform, the ratio of the two times is probably invariant across machines. So, in general, we can say that a server request takes approximately 25 to 35 times longer to process than a call to the fake_request() function.

This is an important result when we need to consider the synchronization needs of different threads that may be running at the same time within these processes.