【Linux Programming】Day2

Class	Understanding Linux/Unix Programming
Date	@March 9, 2022

【Ch1】Unix Systems Programming: The Big Picture(3)

1.5 Unix From the Large Perspective

1.5.1 Connections between Users and Programs

We shall explore how to write programs that work in this larger system. *What is this larger system?*

This larger system is one that consists of more than one user, more than one program, more than one computer, and the connections among other people, programs, and computers.

The following three topics are important in Unix system programming:

- Communication

  How does one user or process communicate with another process?

- Coordination

  How can a program establish coordination between processes so they share resources correctly?

- Network Access

  How can a program connect to other programs using Internet? How does a program gain access to the Inernet?

1.5.3 bc: The Secret Life of a Unix Desk Calculator

Every version of Unix comes with a calculator program called `bc` (stand for bench calculator), a simple, text-based calculator. To start the program, just type

```
$ bc
```

Type an expression and press the Enter key:

```
2 + 3 * 4 + 5 * 10
```

`bc` prints out the correct answer. To exit `bc`, press Ctrl-D

One fascinating thing about `bc` is that it can handle very large integers, such as

```
99999999999999999999999 * 99999999999999999999999
```

`bc` is also a programming language with variables, loops, and a C-like syntax. For example, the following code is acceptable to `bc`:

```
x = 3
if(x == 3) {
  y = x * 3;
}
y
```

Here is the other interesting thing about the `bc` program. `bc` is not a calculator; it does not do any calculations. To see what is going on, try this:

```
$ bc
2 + 3
5
<-- press Ctrl-Z here

Stopped
$ ps
PID   TTY      S          TIME CMD
25102 ttyp2  T       0:00.02 bc
27081 ttyp2  T       0:00.01 dc -
27560 ttyp2  I       0:00.59 -bash
27681 ttyp2  T       0:00.00 bc
$ fg
<-- press Ctrl-D here
```

The `ps` program lists the processes we are running. This display shows four processes. One is `-bash`, a log-in shell. There is also one called `dc -` . *What is this `dc` program?*

The online manual is provided with most versions of Unix. To read the manual page about the dc command, type

```
$ man dc
User Commands dc(1)

NAME
     dc - desk calculator

SYNOPSIS
     dc [ filename ]

DESCRIPTION
     dc is an arbitrary precision arithmetic package. Ordinarily
     it operates on decimal integers, but one may specify an
     input base, output base, and a number of fractional digits
     to be maintained. The overall structure of dc is a stacking
     (reverse Polish) calculator. If an argument is given, input
     is taken from that file until its end, then from the standard
     input.
```

This page says `dc` is a stacking, reverse-Polish calculator. It expects the user to type in the numbers first and the operation later.

```
2
3
+
p
5
```

The manual says that `bc` is a preprocessor for `dc`. `bc` is a parser. It communicates with a process running `dc` through a communication system called pipes, shown in the figure below:

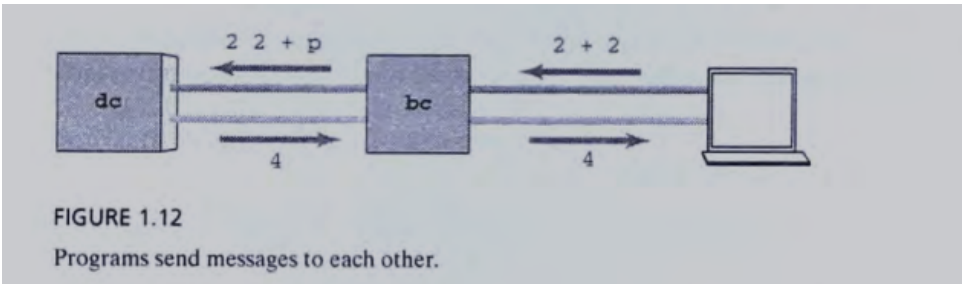

FIGURE 1.12
Programs send messages to each other.

This bc/dc example involves different processes and some sort of communication and cooperation. The separate programs work together as a system; each part does one specific task.

1.5.4 From bc/dc to the Web

In the bc/dc pair, bc is the user interface and dc is the engine that does the work.

With the World Wide Web, the browser is the user interface and the Web server does the work. The architecture is exactly the same:

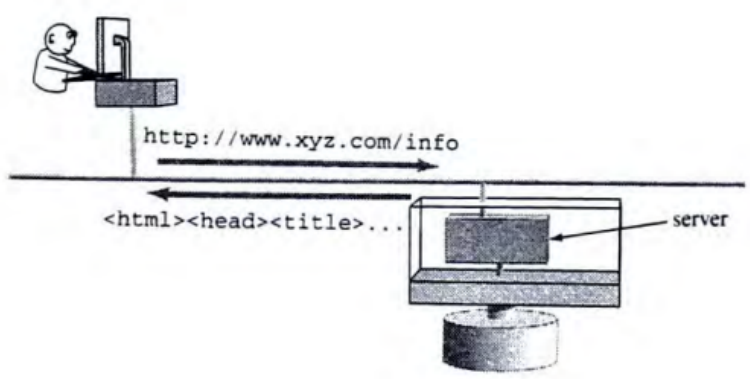

FIGURE 1.13
Separate programs send messages to each other.

The user interacts with the browser. The browser is not where the Web pages are: the Web pages are on servers.

The servers speak a terse, text-based language called HTTP. The user agent translates user input into the terse, text-based language and sends the request to the engine. The user agent receives the reply from the server and formats it for the user.