Plan9 Programming

[0] Plan9 merupakan sebuah sistem operasi yang di kembangkan oleh pencipta UNIX di Bell Lab, sebuah sistem operasi penelitian yang berpengaruh di dunia teknologi, salah satunya adalah UTF-8 dan protokol 9P, Plan9 diciptakan sebagai penerus sistem operasi UNIX tanpa kompatibilitas UNIX karna desain OS yang berbeda, salah satu yang hampir mirip antara UNIX dan Plan9 adalah semuanya adalah file, namun di banding UNIX Plan9 lebih bagus dibanding UNIX, contohnya ketika kita ingin mengkontrol suatu device kita hanya perlu command echo dan cat

contoh penggunaanya adalah:

```
# melihat status volume
cat /dev/volume
# mengubah volume
echo master 60 60 > /dev/volume
```

berbeda ketika kita menggunakan UNIX-Like seperti Linux, kita memerlukan alsactl untuk ALSA, dll.

Plan9 C Dialect

[1] Plan9 menggunakan bahasa program C tetapi tidak menggunakan standar ANSI C melainkan sebuah dialek yang khusus untuk Plan9, seperti inilah perbedaan Hello World pada kedua bahasa, dibawah ini adalah ANSI C yang biasa di tulis programmer

```
#include <stdio.h>

int
main(void)
{
  printf("Hello World\n");
  return 0;
}
```

dan ini adalah program yang ditulis untuk Plan9

```
#include <u.h>
#include <libc.h>

void
main()
{
  print("Hello World\n");
  exits(0);
}
```

bisa dilihat diatas dalam program tidak terdapat #include <stdio.h> melainkan terdapat 2 include, #include <u.h> digunakan untuk kompilasi dengan arsitektur berbeda, karna Plan9 memiliki kompiler yang berbeda setiap arsitektur, setelah itu #include <libc.h> ini digunakan untuk me-link semua library pada Plan9, salah satu perbedaan yang bisa dilihat yaitu fungsi main pada Plan9 tidak memiliki return, hal ini berpengaruh pada sistem error reporting yang dimiliki Plan9 dimana pada ANSI C kita menggunakan int dan menggunakan fungsi errno() untuk mendapatkan pesan error, pada Plan9 C Dialek, langsung menggunakan String, kita bisa dengan langsung menampilkan pesan error dengan fungsi print dan format %r, pada contoh program diatas tidak memiliki error jadi argumennya adalah 0.

Text Editor

untuk dapat menulis program dibutuhkan alat penunjang di antaranya adalah text editor dan build system, untuk graphical text editor Plan9 memiliki [2] sam dan [3] acme, untuk terminal ada [4] ed line editor, berbeda dengan vi/vim dan emacs, untuk [5] shortcut editing pada Plan9 hanya mensupport beberapa shortcut yaitu

```
^U -> Delete from cursor to start of line.
^W -> Delete word before the cursor.
^H -> Delete character before the cursor.
^A -> Move cursor to start of the line.
^E -> Move cursor to end of the line.
```

selain dengan shorcut kita dapat menggunakan mouse untuk berinteraksi dengan text editor namun harus menggunakan mouse dengan 3 tombol, untuk ilustrasi lebih jelas bisa kunjungi [6] Mouse Shortcuts in Acme

Build System

pada GNU/Linux build system yang biasa digunakan adalah make, pada Plan9 build system yang digunakan adalah mk, cara membuat mkfile hampir mirip dengan makefile namun biasanya untuk Plan9 sudah terdapat configurasi, kita hanya perlu mengincludenya saja, seperti inilah contohnya

```
</$objtype/mkfile

TARG=main

OFILES=n
  main.$O

BIN=‘{pwd}

</sys/src/cmd/mkone
```

pada mkfile diatas menggunakan mkone alternatif yaitu mkmany.

pada mkone kita diharuskan membuat variable OFILES yang berisi soure untuk kompilasi, TARG untuk nama file eksekusi yang nanti diinstall, BIN untuk tempat dimana untuk menginstall file eksekusi, biasanya untuk mkone kita menggunakan perintah

```
mk
mk install
```

contoh pada mkmany

```
</$objtype/mkfile

TARG=http_client echo_server

BIN=‘{pwd}

</sys/src/cmd/mkmany
```

untuk dokumentasi lebih jelas kunjungi [7] Plan 9 Mkfiles

Network Programming

berbeda dengan POSIX/UNIX network programming untuk Plan9 sangat mudah, pada POSIX/UNIX kita menggunakan socket dan memerlukan banyak kode, berbeda dengan Plan9 dimana sangat simpel, dan saya perhatikan ini merupakan cara yang

menginspirasi networking pada bahasa program go, pada Plan9 dan Go sama sama menggunakan nama fungis yang sama [8] dial seperti inilah perbedaan http client untuk Plan9 dan Go:

Plan9 http client

```
#include <u.h>
#include <libc.h>

#define RSP 4096

void
main()
{
        const char* server = "tcp!9p.io!http";
        const char* request = "GET / \r\n";
        char response[RSP] = {0};

        int nfd = dial(server, 0, 0, 0);
        if(nfd < 0)
                sysfatal("failed to connect, %r");

        write(nfd, request, strlen(request));
        read(nfd, response, RSP);

        print("%s", response);

        close(nfd);
        exits(0);
}
```

GO http client

```
package main

import (
        "fmt"
        "log"
        "net"
        "os"
)

func main() {
        const server = "9p.io:http"
        const request = "GET / \r\n"
        var response = make([]byte, 4096)

        conn, err := net.Dial("tcp", server)
        if err != nil {
                log.Fatal("failed to connect, ", err)
        }

        conn.Write([]byte(request))
        conn.Read(response)

        fmt.Print(string(response[:]))

        conn.Close()
        os.Exit(0)
}
```

untuk penjelasan lebih jelas kunjungi [8] dial(2)

Thread/Proc/whatever

[9] Thread adalam Plan 9 terinspirasi dari Alef dan Newsqueak, di dalam Plan 9 terdapat sebuah librari thread yang memiliki fungsi untuk thread dan process, untuk melakukan komunikasi interproses kita dapaat menggunakan channel, ya seperti bahasa program GO.

berikut adalah contoh penggunaan thread dan channel untuk komunikasi antar thread, untuk menggunakan thread kita harus mengganti fungsi main dengan threadmainyang menarik yaitu program ini hampir mirip dengan yang ada di GO

```
#include <u.h>
#include <libc.h>
#include <thread.h>

void
t(void *t)
{
        Channel *c;
        c = t;

        int n = 10;

        send(c, &n);
}

void
threadmain(int argc, char *argv[])
{
        USED(argc);
        USED(argv);

        int n;
        Channel *c;
        c = chancreate(sizeof n, 0);

        threadcreate(t, c, 1024);

        recv(c, &n);
        print("Receive: %d\n", n);
}
```

TCP echo server

untuk membuat proses baru kita bisa menggunakan fungsi fork dan rfork, berikut merupakan tcp server yang dapat menerima clien lebih dari 1

```
#include <u.h>
#include <libc.h>
#include <thread.h>

#define RSP 4096

void
connection(void *arg)
{
        USED(arg);

        char buf[RSP];
        int dfd;

        Channel *c;
        c = arg;

        recv(c, &dfd);
        int n;
        for(;;){
                n = read(dfd, buf, sizeof(buf));
                write(dfd, buf, n);
        }
        close(dfd);
        threadexits(0);
}

void
threadmain(int argc, char *argv[])
{
        USED(argc);
        USED(argv);

        char adir[40], ldir[40];
        char* server = "tcp!127.0.0.1!7420";

        int afd = announce(server, adir);
        if(afd < 0)
                sysfatal("failed to announce, %r");

        print("Starting server on %s\n", server);

        for(;;){
                int lfd = listen(adir, ldir);
                if(lfd < 0)
                        sysfatal("failed to listen, %r");
                switch(rfork(RFFDG|RFPROC|RFNOWAIT|RFNAMEG)){
                case -1:
                        close(lfd);
                        continue;
                case 0:
                        int dfd = accept(lfd, ldir);
                        if(dfd < 0){
                                print("failed to accept, %r");
                                continue;
                        }
                        Channel *c;
                        c = chancreate(sizeof dfd, 0);
                        threadcreate(connection, c, 1024);
```

```
                        send(c, &dfd);
                default:
                        close(lfd);
                        break;
                }
        }

        threadexitsall(0);
}
```

Referensi

[0] *Plan 9 from Bell Labs Overview* https://9p.io/plan9/about.html

[1] *C Programming in Plan 9 from Bell Labs* https://doc.cat-v.org/plan_9/programming/c_programming_in_plan_9

[2] *The Text Editor Sam by Rob Pike* http://sam.cat-v.org

[3] *The Acme User Interface for Programmers* https://acme.cat-v.org

[4] *ed(1)* https://man.cat-v.org/unix-1st/1/ed

[5] *Keyboard Shortcuts for Acme* https://acme.cat-v.org/keyboard

[6] *Mouse Shortcuts in Acme* https://acme.cat-v.org/mouse

[7] *Plan 9 Mkfiles* https://9p.io/sys/doc/mkfiles.html

[8] *dial(2)* https://man.9front.org/2/dial

[9] *thread(2)* https://man.9front.org/2/thread