IT-215

System Software

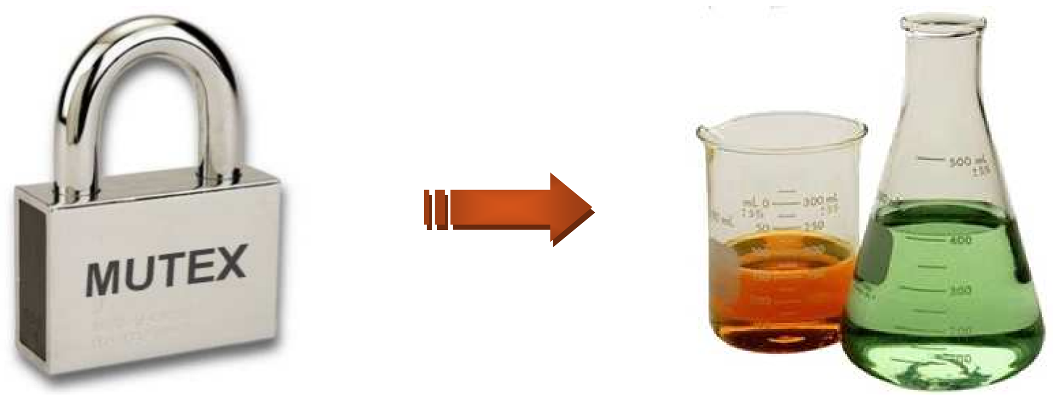
Project Report

Submitted on: 16th April 2012

Simulation of Manufacture of Sulphuric Acid

Guided By: Prof. Sanjay Chaudhary

Mentored By: Anadi Chaturvedi



Submitted By:

Prashant Pandey (201001067)

Swena Gupta (201001104)

Goal

To simulate process automation for a manufacturing plant

Problem Statement

Assume that there is a manufacturing plant, which manufactures finished products based on processing on raw materials. There are few machines in the plant. Each machine inputs raw materials, performs processing and produces semi-finished product, which is supplied as input or raw material to another machine. Thus, there is a chain of machines which are involved in the process inputting raw materials or semi-finished product and produces semi-finished product. Finally finished product will be the final output. Each machine is to be simulated as a process. Various processes can be initiated independently of each other and executed in background. Each process may take input from a specific named pipe and write output as a processed semi-finished product on some other named pipe. Define the sequence of actions among 5-7 such machines, define the flow input and output among them. One machine should be identified as a machine, which generates finished product, i.e. gray cloth. Each machine operates under certain environmental conditions, i.e. machine should be operated within the range of temperature. If temperature level of any of the machine is more than the upper limit then it should generate signal and send it to the main controlling machine, which will inform all other machines to stop operations.

Concepts Used

- Multiple Threading
- Mutual Exclusion and Condition Variables
- Inter Process Communication using Pipes
- Signal Handling and Processing
- Accessing System time.

Programming Language

C Language inclusive of concepts of Shell Programming.

Tools used

gcc compiler.

Header Files Used

1) #include <stdio.h>
   This header file contains declarations used in most input and output and is typically included in all C programs.
   eg. printf() etc.

2) #include <stdlib.h>
   This header file contains Utility functions.
   eg. exit() etc.

3) # include <pthread.h>
This header file defines the threads and its functions.
eg. pthread_create() etc.

4) #include <signal.h>
To catch the various signals implemented in our program, we have included <signal.h> because it contains sigaction as a structure.

5) #include <string.h>
This header file contains String functions.
eg. strlen()

6) #include <time.h>
This header file contains implementation of date and time manipulation operations.

Test Programs

a)

```
time_t now;
time (&now);
printf ("\n\n%s\n\n", ctime (&now));
```

//the above code snippet has been used to access system time.

b)

```
act.sa_handler = catchint;
sigfillset(&(act.sa_mask));
sigaction(SIGINT , &act, NULL);
sigaction(SIGUSR1 , &act, NULL);
void catchint(int signo)   // handler for sigaction
```

```
{
        if(signo==SIGINT)
        printf("\n\n\'Ctrl+C\' pressed...Interrupt generated and
ignored!!!\n\n");
        if(signo==SIGQUIT)
        .
        .
        .
        .
}
```

// The above code snippet is used to call signals at various activities of user such as pressing Ctrl+C to generate SIGINT etc. and to handle them using catchint function.

c)

```
if(pthread_create(&control_thread, NULL, control_machine, (void *)
quantity)!=0)
{
        printf("Failed to create a thread\n");
        exit(1);
}
        pthread_join(control_thread, NULL);
```

// The above part of program is used to create threads and to join them

d)

```
pipe (p);
write (p[1], write_msg, 60);
read(p[0], read_msg, 60);
```

// the above statements are used to declare and do read-write operation on a named pipe.

e)

```
pthread_mutex_lock (&count_lock);
while (taskid >= COUNT)
{
        pthread_cond_wait (&count_cond, &count_lock);
}
pthread_cond_broadcast (&count_cond);
pthread_mutex_unlock   (&count_lock);
```

//the above statements are used to put mutual exclusion amongst all the threads and to make all other threads wait while one thread is executing.

Assumptions

1. The Initial & Favourable specifications of machines are pre-defined as stated in program (while we run the program, it displays on the console).

2. According to the code made by us, the specifications has been set, so that if the required amount is > 3 litres, all the 5 stage proccess machines gets overheated & production of 4th lts onwards acid needs to cool each machine. This has been shown appropriately while production of 4th litre and onwards acid production.

3. This has been done deliberately to introduce and show the concept of temperature in our machine automated manufacturing-plant system.

4. Following SIGNALS have been handled in our program

a. SIGINT : signal called for handling Ctrl+C.

b. SIGQUIT : signal called for handling Ctrl+\

c. SIGALRM : signal called for production of next litre of acid.

d. SIGWINCH: signal called when output terminal window size is changed.

e. SIGTSTP : signal called for handling Ctrl+Z.

f. SIGUSR1 : signal called if the user enters invalid input.

List of Programs

- final_manufacture.c

Log Files

- log_for_3_litres_prodn
- log_for_6_litres_prodn
- log_for_9_litres_prodn
- log_for_wrong_user_input
- log_with_all_signals_handled_6_litres