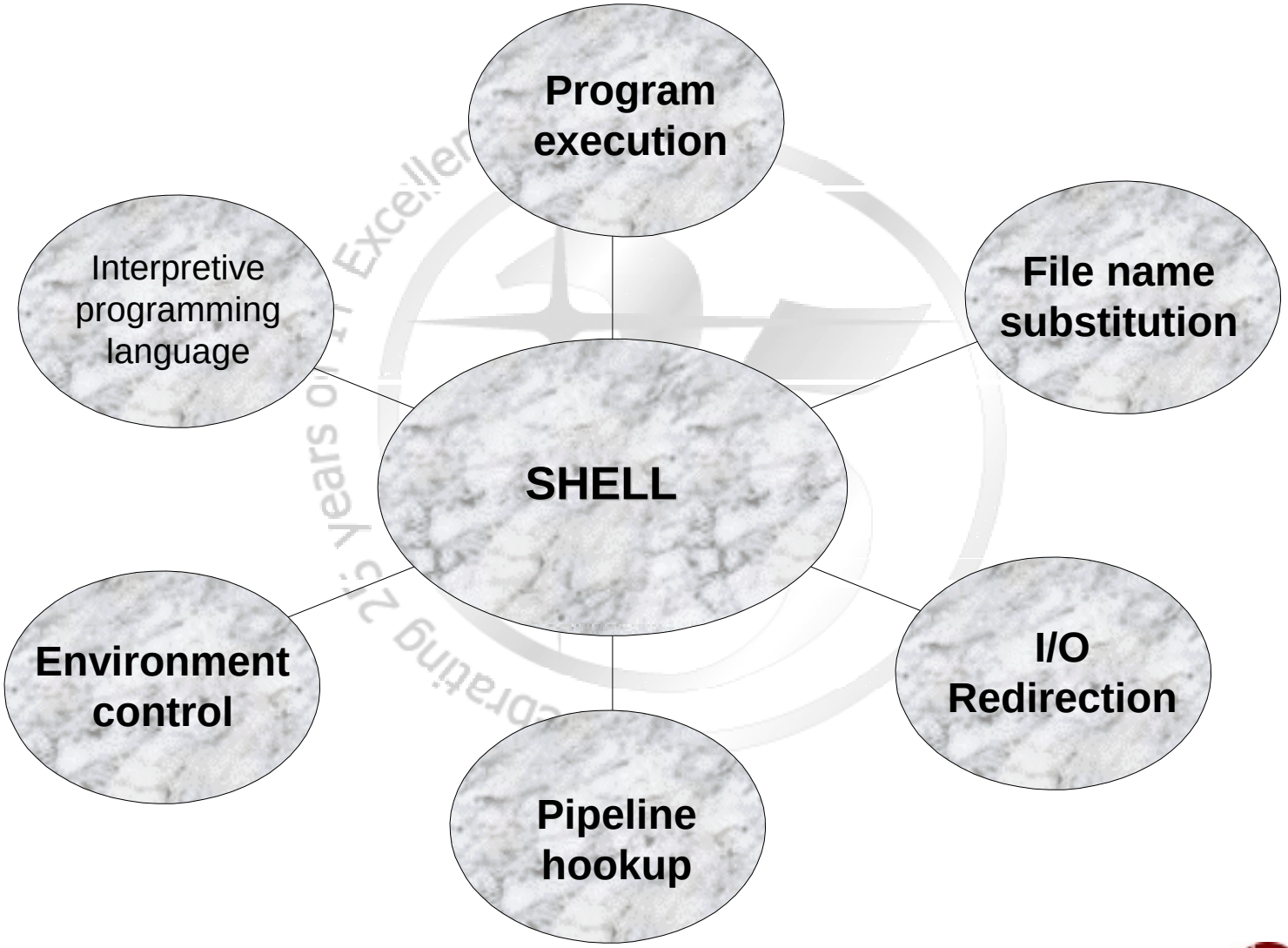
The Shell

Shell

- It is a program that interprets the commands.
- If the command is valid then the shell directs the kernel to carry out the request, and if invalid then an error message is displayed.
- Shell starts when an user logs in, and terminates when the user logs out.
- Its presence is indicated by a special symbol known as the shell prompt ($ or #).

- **Several shells are available to handle the same hardware in different ways.**
- **Redirection of data : the shell facilitates chaining or "pipelining" of commands, i.e. the output of one program flows down the pipe and becomes an input to the next program.**


Program execution
Interpretive programming language
File name substitution
SHELL
Environment control
I/O Redirection
Pipeline hookup

Activities done by a shell

- When user logs in, a shell starts
- It then issues a shell prompt and waits for user to enter a command.
- After a command has been entered the shell scans the command line for some special characters, and then rebuilds the command line after processing is complete.
- The command is then passed on to the kernel for execution, and the shell waits for its completion.
- The shell prompt appears again, and then shell waits for the next command.

Program Execution

- Command Line
- Every command that is executed under the UNIX system has the same general format, as far as the shell is concerned

Command Arguments

- The shell treats the first characters on the line up to the first blank as the name of the command to be executed
- Any characters appearing after the command name are interpreted as the command's arguments

File Name Substitution

- **If file name substitution is specified on the command line with the characters *,?,or […], then the shell performs the substitution. This happens before the program gets executed; the program itself never has to worry about it**

I/O Redirection

- **If input and/or output redirection is specified on the command line, then this to is handled by the shell. This happens before the execution of the program; the program doesn't even know that its input/output has been redirected**

Pipeline Hookup

- If the command line contains two programs connected by a pipe, then the shell takes responsibility for connecting the standard output of the first program to the standard input of the second. It then starts execution of both the programs. As with I/O redirection, the fact that these two programs have been connected by a pipe is unknown to eitherof the two programs

- **Environment Control**
- **The shell gives you some flexibility in customizing your “Environment” to suit your needs**

Interpretive Programming Language

- The shell provides a powerful programming language. Statements in this language can be typed in directly at the terminal for execution, or they can be entered into a file.

Interpretive Programming Language

- The shell provides a powerful programming language. Statements in this language can be typed in directly at the terminal for execution, or they can be entered into a file.

Different shells

Bourne shell or Standard shell (sh) :

- **Introduced in 1978 and is widely used in AT&T Unix.**
- **Gives "$" as the prompt to the user and " # " to the superuser (root).**
- **One disadvantage is that it does not provide command history.**

C shell (csh)

Cont...

- **Gives "%" as the prompt to user.**
- **It is different from the sh.**
- **Syntax is similar to the C language programming.**
- **It is quite popular among the programmers.**
- **Provides aliasing of the command i.e. the ability to customize the command names.**

Restricted shell (rsh) :

- **Basically designed to restrict the rights of normal users**
- **It cannot execute**
  - **cd command**
  - **path variable**
  - **command containing "/”**
  - **redirect outputs**

UUCP : UNIX To UNIX Communication Protocol

- **Basically designed to communicate between two UNIX system**
- **It checks the transmitted data and if necessary it retransmits**
- **Using it, one can copy, transfer, load, remove and clean files and directories on the UNIX system at a remote location.**

Visual shell (vsh):

- **It is user friendly and menu driven interface.**
- **Good for the new users.**

Korn shell (ksh) :

- **By David Korn of AT&T Bell laboratories.**
- **More efficient then sh (it basically combines features of the C shell and incorporates some more functionality while maintaining backward compatibility with the bourne shell).**

- **No need for subshells as there are many functions in it.**
- **It also offers history by redirecting commands to vi with ksh -o vi & commands can be reused by esc k & esc j.**

Identifying your shell

Type the following line and press enter

echo $RANDOM

Results for the different shells are

- **Bourne** **Blank line**
- **C Shell** **RANDOM : Undefined variable**
- **Korn Shell** **Five digit random number**

Generation Of Argument List (Wild Cards)

In UNIX there special characters that can be included in a pattern.

The Special Chara. Are

- **The Asterisk which matches any string**
- **? The Question mark which matches any one chara [and] Any sequence of charas. enclosed within brackets which matches any one of the enclosed charas.**

- **If a pair of chars are separated by - dash inside brackets, matches any chara with in that range of that pair**

Note:

If the first character of a filename is a dot (.), it can be matched only by an argument that literally begins with a dot (.)

Quoting Mechanism

- Characters including <, >, *, ?, [and] give special meaning to the shell. To remove the special meaning of these characters some form quoting is required.
- This is done by using (‘) mark or (“) marks to surround a string. A (\ ) before a single character provides the same function.

- **All characters within the (‘) mark are taken literally.**
- **Within (“) the special meaning of certain characters persist.**
- **The characters which retain their special meaning are $, \, ‘, * and “.**

- **Command Lines & Pipelines**
- **A sequence of commands separated by the vertical bar (|) makes up a pipeline.**
- **A filter is a command that reads its standard input,transforms it in some way,then writes it as its standard output.A pipeline normally consists of a series of filters.**

Examples

- Following are examples of typical pipelines :-
- who – prints the list of logged in users
- who >>log – appends the list of logged-in users to the end of file “log”
- who | pr – prints a paginated list of logged in users

Shell startup DOT files

- Helps in customizing UNIX system
- Executes .profile for Bourne Shell when users logs in
- Executes .login & .cshrc for C shell when user logs in
- Executes .logout for C shell when user logs out from C-shell

The Bourne Shell

- This shell has been developed by Steve Bourne.This is one of the most widely used shells .If you list out the /bin directory , you will see an executable program named sh. This is the Bourne shell.

The C shell

- The C shell program csh is a command language interpreter.
- The C shell can be invoked by typing in csh at the command line .
- The C shell maintains a set of variables, to which values can be assigned by using the set command .The syntax is :-
- Set name = value

The Korn Shell

- The Korn shell is a product of AT&T Bell labs & was developed by David Korn .
- It has many novel features like the facility to edit the command line,the history mechanism,aliases & job control.
- Moreover the Korn shell also lets you handle computations awk-like formatted printing & simplified menu routines.