iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 2 **Issue:** X **Month of publication:** October 2014

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology(IJRASET)

Interactive Shell Script & Advance Features of Shell Programming

Ankit Saxena[1], Himanshi Jhamb[2]
[1,2] Information Technology Department, Dronacharya College of Engineering, Gurgaon

Abstract: ***In this research paper, the concept of shell scripting and programming is discussed and various aspects of shell programming are also studied. A shell script is a computer program designed to be run by the UNIX shell which is a command line interpreter. The various dialects of shell scripts are considered to be scripting languages. Typical operations performed by shell scripts include file manipulation, program execution, and printing text. A shell script can provide a convenient variation of a system command where special environment settings, command options, or post-processing apply automatically, but in a way that allows the new script to still act as a fully normal UNIX command. The major concepts like Programming in the Borne and C-shell, in which it would be explained that how shell programming can be done in Borne and C-shell. This paper also undergoes the study about origin of Shell Programming, about the Shell Programming Constructs and advance features of shell programming are also studied and followed. In this paper, it would also be discussed that how Shell Programming in how many ways is beneficial in the UNIX Shell.***

I. INTRODUCTION

A shell script is a computer program designed to be run by the UNIX shell, a command line interpreter. The various dialects of shell scripts are considered to be scripting languages. Typical operations performed by shell scripts include file manipulation, program execution, and printing text. In its most basic form, a shell script can provide a convenient variation of a system command where special environment settings, command options, or post-processing apply automatically, but in a way that allows the new script to still act as a fully normal UNIX command.

At the heart of UNIX is a kernel whose routines aren't easy to use directly. There are also many pre-written programs. They're easier to use, but they don't understand the use of '*'as a wildcard character, and besides, you need some sort of editor when you type commands in if you're going to use the command line. The 'shell' is the interface between the user and the system. The shell isn't only a line editor and a command line interpreter that understands wild cards; it's also a language with variables, arrays, functions and control structures. Command line scan be put into a file and executed. These so-called "shell scripts" can quickly be written and tested and should be tried in association with other standard UNIX utilities before embarking on a higher level language, at least for prototyping purposes.

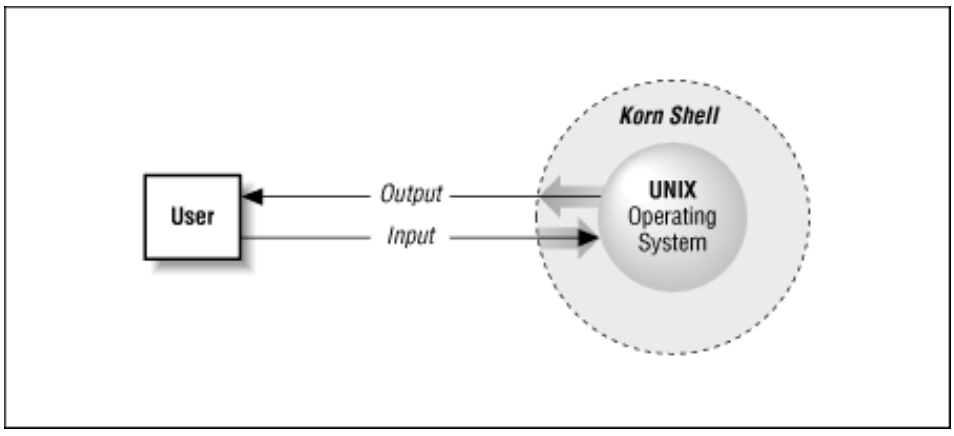


Fig1. Shell in UNIX operating system

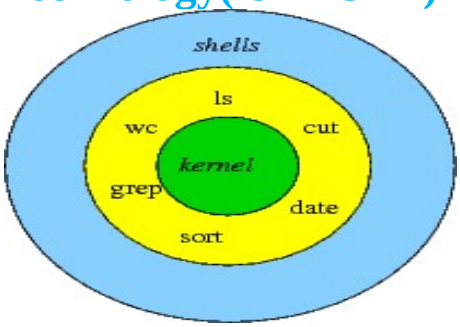


Fig2. Shell & Kernel in Unix Operating System

Shell scripts allow several commands that would be entered manually at a command-line interface to be executed automatically, and without having to wait for a user to trigger each stage of the sequence. For example, in a directory with three C source code files, rather than manually running the four commands required to build the final program from them, one could instead create a C shell script, here named build and kept in the directory with them, which would compile them automatically.

The script would allow a user to save the file being edited, pause the editor, and then just run. Build to create the updated program, test it, and then return to the editor. Since the 1980s or so, however, scripts of this type have been replaced with utilities like make which are specialized for building programs.

Shell scripts often serve as an initial stage in software development, and are often subject to conversion later to a different underlying implementation, most commonly being converted to Perl, Python, or C. The interpreter directive allows the implementation detail to be fully hidden inside the script, rather than being exposed as a filename extension, and provides for seamless reimplementation in different languages with no impact on end users.

II. RELATED WORK

There are various operations performed under shell programming, some of them are file manipulation, program execution, and printing text. Each of the operations performed under shell programming is studied in detail.

III. FILE MANIPULATION IN SHELL PROGRAMMING

This introduction to text manipulation on UNIX platforms provides an overview of some common commands widely available and installed standard on most UNIX-based releases. Many times these standard utilities are ignored in favor of more modern text-processors such as Perl, Python, or Ruby, which are not always installed on a system. An introductory review of these tools helps practitioners who are learning UNIX or Linux or those who may be looking to renew forgotten knowledge

IV. PROGRAM EXECUTION IN SHELL PROGRAMMING

A UNIX shell is a command language interpreter, the primary purpose of which is to translate command lines typed at a terminal into system actions. The shell itself is a program through which other programs are invoked. Although there are several different UNIX shells, among them the C shell, the Bourne shell and the Korn shell, the one most frequently used on Blake within Berkeley UNIX is the C shell.

The shell has some built-in functions, which it performs directly, but most commands that you enter cause the shell to execute programs that are external to the shell. This sets the shell apart from other command interpreters, because it is just another user program at the same time that it functions almost exclusively as a mechanism for invoking other programs.

The UNIX shell program interprets user commands, which are either directly entered by the user, or which can be read from a file called the shell script or shell program. Shell scripts are interpreted, not compiled. The shell reads commands from the script line per line and searches for those commands on the system, while a compiler converts a program into machine readable form, an executable file - which may then be used in a shell script.

Apart from passing commands to the kernel, the main task of a shell is providing a user environment, which can be configured individually using shell resource configuration files.

V. PROGRAMMING IN BOURNE AND C SHELL

A Bourne Shell Script is a file containing a series of Bourne Shell commands, as well as control structures such as if statements, while loops, etc. Parameters can be passed to the script and data can be read from the keyboard.The Bourne shell (sh) is a shell, or command-line interpreter, for computer operating systems.

The Bourne shell was the default UNIX shell of Unix Version 7. Most Unix-like systems continue to have /bin/sh—which will be the Bourne shell, or a symbolic link or hard link to a compatible shell even when other shells are used by most users.

Developed by Stephen Bourne at Bell Labs, it was a replacement for the Thompson shell, whose executable file had the same name—sh. It was released in 1977 in the Version 7 Unix release distributed to colleges and universities. Although it is used as an interactive command interpreter, it was also intended as a scripting language and contains most of the features that are commonly considered to produce structured programs.

Example of Bourne shell programming is given below:

To read commands from the terminal and process them:

```
#! /bin/sh
  # usage: process sub-directory
  dir=`pwd`
  for i in *
  do
  if test -d $dir/$i
  then
    cd $dir/$i
    while echo "$i:"
    read x
    do
      eval $x
    done
    cd ..
  fi
done
```

The user types the command:

A. Process Sub-Directory

This script will read and process commands in the named sub-directory. The user is prompted to supply the name of the command to be read in. This command is executed using the the builtin eval function.

The C shell (csh or the improved version, tcsh, on most machines) is a UNIX shell that was created by Bill Joy while he was a graduate student at University of California, Berkeley in the late 1970s. It has been distributed widely, beginning with the 2BSD release of the BSD Unix system that Joy began distributing in 1978. Other early contributors to the ideas or the code were Michael Ubell, Eric Allman, Mike O'Brien and Jim Kulp.

The C shell is a command processor typically run in a text window, allowing the user to type commands. The C shell can also read commands from a file, called a script. Like all Unix shells, it supports filename wildcarding, piping, documents, command, variables and control structures for condition-testing and iteration. What differentiated the C shell from others, especially in the 1980s, were its interactive features and overall style. Its new features made it easier and faster to use. The overall style of the language looked more like C and was seen as more readable.

Example of C shell programming is given below:

```
#!/bin/csh

If($days> 365) then

Echo This is over a year.

Endif
```

VI. ADVANTAGES OF SHELL PROGRAMMING

A. Easy to use.

B. Quick start and interactive debugging.

C. Time saving.

D. System admin task automation.

E. Shell scripts can execute without any additional effort on nearly any modern UNIX / Linux / BSD / Mac OS X operating system as they are written an interpreted language.

VII. DISADVANTAGES OF SHELL PROGRAMMING

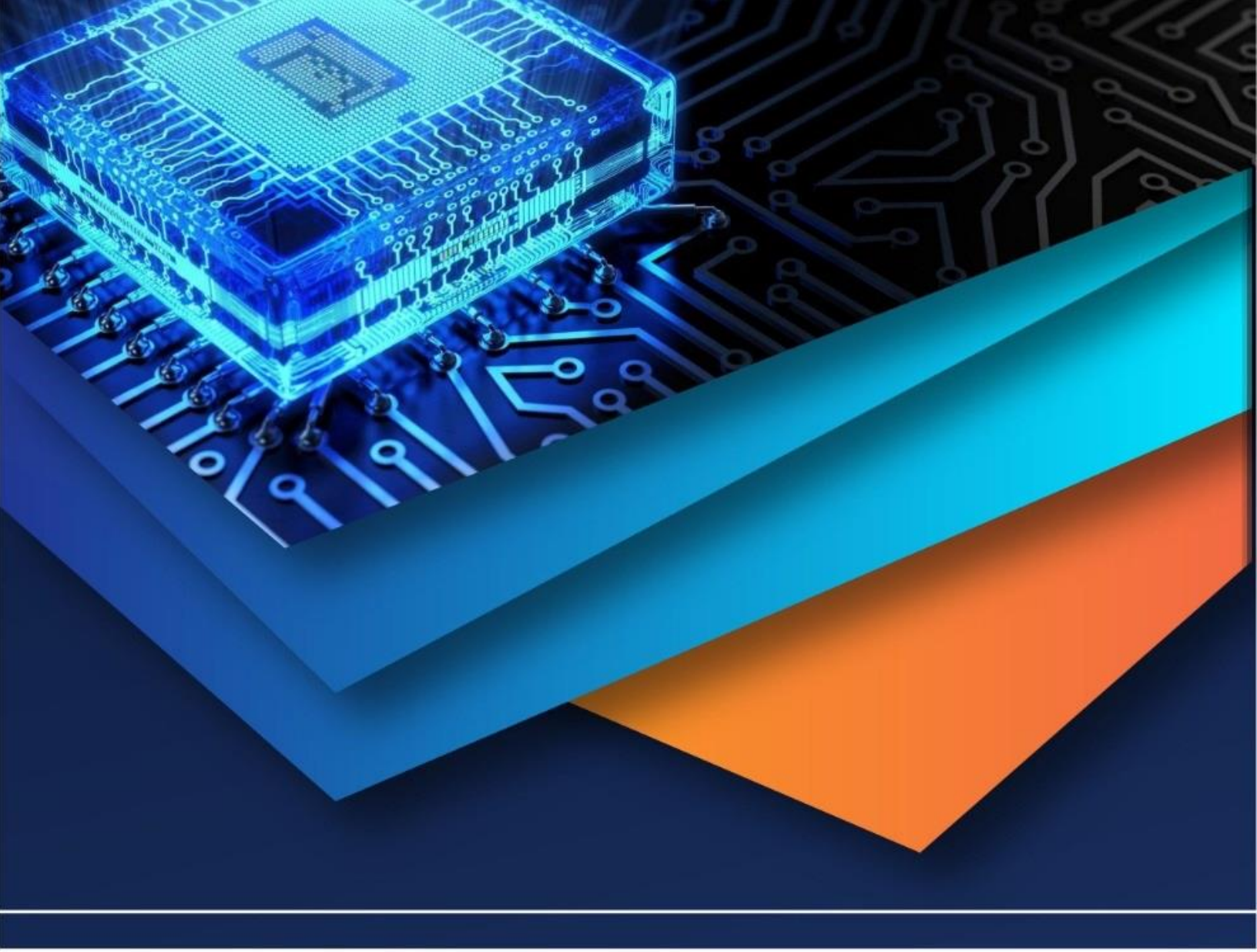
A. Compatibility problem between different platforms.

B. Slow execution speed.

C. A new process launched for almost every shell command executed.

REFERENCES

[1] http://programmers.stackexchange.com/
[2] http://techforum4u.com/
[3] http://linuxshellscripting.blogspot.in/
[4] Concept and applications of UNIX- Sumitabha Das
[5] Windows NT shell scripting- Timothy Hill
[6] Shell Scripting Recipes: A Problem-Solution Approach- Chris F.A Johnson.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY