Document Number: <QMS Ref. No.>	***Copy Number:*** <copy number>

TATA CONSULTANCY SERVICES

AHMEDABAD

vILP – Unix - Customizing the shell environment

Content Manual

Version 1.2

May 2014

TCS Internal

DOCUMENT RELEASE NOTICE

Notice No.
Client

Project
Document details

Name	Version No.	Author	Description
Customizing the shell Environment	1	Priyank Patel	Study of Shell variables, The Command history, Misc Utilities. Working with Shell environment,

Revision details:

Action taken (add/del/	Preceding page No.	New page No.	Revision description

Change Register serial numbers covered:
The documents or revised pages are subject to document control.
Please keep them up-to-date using the release notices from the distributor of the document.
These are confidential documents. Unauthorised access or copying is prohibited.

Approved by : _____ Date: _____

Authorised by: _____ Date: _____

DOCUMENT REVISION LIST

Client
Project
Document Name Virtual_ILP_Customizing_the_shell_environment
Release Notice Reference (for release)

Rev.	Revision date	Revision description	Page No.	Prev page No.	Action taken	Addenda /New page	Release Notice reference

CONTENTS

1.1 1. Customizing the shell environment

1.2 Objectives

- Shell Variables
- The Command History
- Misc Utilities

1.3 Shell Variables

- The shell maintains a set of internal variables known as shell variables. These variables cause the shell to work in a particular way. Shell variables are local to the shell in which they are defined; they are not available to the parent or child shells.

- A shell variable is defined by the set command and deleted by the unset command. To define a new variable or change the value of one that is already defined, enter:

set *name*=value

To delete, or unset, a shell variable, enter:

unset *name*

To see the value of a single variable, use the echo command:

echo $*name*

- Built-in variables are automatically set by the shell and are typically used inside shell scripts. Variables set automatically by shell are:

$#	Number of command-line arguments.
$-	Options currently in effect (arguments supplied to sh or to set)
$?	Exit value of last executed command.
$$	Process number of current process.
$!	Process number of last background command.
$0	First word; that is, command name.
$n	Individual arguments on command line (positional parameters).
$*	All arguments on command line ("$1 $2...").
$@	All arguments on command line, individually quoted ("$1" "$2" ...).

1.4

1.5 The Command History

1.4.1 The History Command

- The history command can be used to view the command which we have used before. This log is called the “history”.

history n
This will only list the last n commands.

Example : history 5
It will show last 5 commands executed.

history
Will print a list of commands along with a numeric index.

Example : history
It will show all the commands executed previously.

history -c
Deletes the command history

The -c option causes the history list to be cleared by deleting all of the entries.

1.4.2 The .sh_history/.bash_history file

- A special file in the UNIX user home directory called .sh_history is used by UNIX to record and allow for fast retrieval of prior commands

- The .sh_history file is commonly used as an audit mechanism, since each and every UNIX command entered by the UNIX user is stored in their .sh_history file.

1.6 Misc Utilities

1.4.1 Alias

- The alias command allows you to assign shorthand name to frequently used commands. Aliases are recognized only by the shell and defined in the same way as a variable.
- The format of the alias command is:

    alias [new [old]]

- The First example, a UNIX user often use wc -l command, so if you don't have wl command or alias on your system, you can create an alias:

alias wl = 'wc -l'

- Quoting is necessary for multiple words. Once the alias is defined, you can run wc -l command simply by using wl.
- The second example creates the alias dir to list directory files only:

alias dir =' ls – l | grep ^d '

- To consider another example, we often use the cd command for long pathnames. You can create alias like this:

  alias cdmyproj = “cd/usr/myfolder/myproj”

- Using unalias statement you can unset the alias.

  unalias dir
  unalias wl

1.4.2 Set and Shift command

- The set statement by default displays the variables in the current shell, but in Bash and Korn, using -o option we can change environment settings.
- This option, followed by a keyword, takes care of some of the common hazards faced by the users, like overwriting files and accidental logging out.
- File Overwriting (noclobber) The shell's > symbol overwrites an existing file, and to prevent such accidental overwriting, you have to use the noclobber argument in this way:

  set -o noclobber

- This means that if you redirect output to an existing file, the shell will prompt a message:

  bash: <filename>: cannot overwrite existing file
  ksh: <filename>: file already exists

- To override this protection feature for file foo, use the | after the >:

  head -n 5 emp.lst >| foo

- Accidental Logging Out (ignoreeof) User often inadvertently press [Ctrl-d] with intent to terminate standard input, but end up logging out of the system. The ignoreeof keyword offers protection from accidental logging out:

    set -o ignoreeof

- Now when you use [Ctrl-d] to terminate your session, the shell typically responds like this:

Use 'exit' to terminate this shell

- You now have to use the exit command to take you out of the session.