Scheduling UniVerse jobs with UNIX tools

Robert F. Woods
Strategy 7 Corporation

05/20/09

Contents

1 Introduction

1.1 Statement of problem

The UniVerse database does not provide a mechanism for scheduling jobs. Numerous third party software packages exist to perform these function. When UniVerse is hosted on a UNIX server, there are some UNIX tools that provide support for job scheduling. This paper will explore some of these options.

1.2 UNIX tools

1. cron
2. at and batch
3. loging messages to syslog
4. mail
5. IO redirection ">"

1.3 UniVerse tools

1. LOGIN paragraph
2. COMO
3. PHANTOM
4. Paragraph programming
   (a) Inline prompting
   (b) IF THEN programming
   (c) @variables

1.4 Concepts of importance

There are several concepts that are especially important in this paper.

Section 5.2 discusses UniVerse record locking issues.

Section 7.4 demonstrates techniques to establish the environment for cron jobs.

Section 8 demonstrates the use of the fully qualified path for the execution of the "uv" UniVerse shell.

2 UNIX tools documentation

2.1 Periodic Program Execution: The cron facility

Cron allows you to schedule programs for periodic execution.

Cron jobs are run by a system program in an environment that's much different from your normal login sessions. The search path is usually shorter and you may need to use absolute pathnames for programs that aren't in standard system directories. [1]

2.2 The at Command

The at facility submits a command line (or a script) for execution at an arbitrary later time.
[2]

2.3 Batch command

Execute commands entered on standard input. Unlike at, which executes commands at a specific time, batch executes commands one after another
[3]

2.4 Logging messages

Log messages to the system log. [4]

2.5 mail command

Read mail or send mail to other users. [5]

2.6 IO Redirection

A technique in Unix scripting to redirect input and output at the shell level to files or other input sources.
[6]

[1] UNIX Power Tools; Jerry Peek, Tim O'Reilly and Mike Loukides; O'Reilly ISBN; 1-56592-260-3
[2] UNIX Power Tools; Jerry Peek, Tim O'Reilly and Mike Loukides; O'Reilly ISBN; 1-56592-260-3
[3] Unix in a Nutshell, Third Edition.; by Arnold Robbins; O'Reilly ISBN; 1-56592-427-4
[4] Unix in a Nutshell, Third Edition.; by Arnold Robbins; O'Reilly ISBN; 1-56592-427-4
[5] Unix in a Nutshell, Third Edition.; by Arnold Robbins; O'Reilly ISBN; 1-56592-427-4
[6] Unix in a Nutshell, Third Edition.; by Arnold Robbins; O'Reilly ISBN; 1-56592-427-4

3 UNIX scripting techniques

There are as many ways to write UNIX scripts as there are programmers. This section will demonstrate some rudimentary methods.

3.1 Script documentation

Documentation of any program is essential to give clues to those who follow. The following is an example.

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   foo
# Author:       Robert Woods
# Date written: 10/22/08
# Purpose:      This script was written for demonstration
# Additional lines of description can be entered at this point.
# Syntax:       from $ enter ./foo
```

3.2 Script logging

UNIX provides an event logging mechanism called syslog. This facility can be configured to have a local log. Information is time stamped and written to this log by the logger command

4 UniVerse Tools

4.1 UniVerse Login Entry

In the UniVerse environment, the system administrator can set system wide defaults by placing appropriate commands in a paragraph, sentence, proc, menu, or BASIC program named UV.LOGIN in the VOC file of the UV account. The UV.LOGIN entry is executed every time a user logs in to any UniVerse account.

You can also create a local login entry in the VOC file of any UniVerse account. The login entry can be a paragraph, sentence, proc, menu, or BASIC program. The login entry is executed after the UV.LOGIN entry when a user logs in to that particular UniVerse account. The record ID of a UniVerse account's login entry depends on the flavor of the account. In IDEAL, PIOPEN, and INFORMATION flavor accounts, a login entry is always called LOGIN. In PICK, IN2, and REALITY flavor accounts, the login entry can be one of the following:

1. The name of a UniVerse account, read from the UV.ACCOUNT file in the UV account. The UV.ACCOUNT file defines all UniVerse accounts on the system
2. The user's login name
3. LOGIN, the name of a record in the VOC file of the UniVerse account in which you are currently working

UniVerse looks for a login entry in the order shown. [7]

4.2 Como

COMO

Use COMO to start or stop copying terminal output to a record in the &COMO& file. You can also use COMO to print records from the &COMO& file, delete them from the system, or list their names.

Description COMO is short for command output.

If you use COMO without any options, COMO prompts for the necessary information. [8]

4.3 PHANTOM

Use PHANTOM to start a process that executes in the background. A phantom process cannot require input from the terminal.

Syntax

PHANTOM [BRIEF] [SQUAWK] command

Description

[7] IBM UniVerse System Description Manual

[8] IBM UniVerse User Reference

There is a system wide limit on the number of processes that you can initiate. If you can start no more processes when you issue a PHANTOM command, the following message appears:

```
NO FREE PHANTOMS
You cannot run a phantom process now. Wait a while,
then try again.
```

If the space is available, the process begins and displays a message similar to the following:

```
Phantom process started as Process ID pid\#.
```

The operating system assigns the process ID number, pid#.

A phantom process cannot use any terminal services. Use DATA statements in a paragraph to provide input to a phantom process. For example, the following paragraph runs the BASIC program MYPROG and supplies input to it using two DATA statements:

```
BACKGROUND
0001: PA
0002: RUN BP MYPROG
0003: DATA A
0003: DATA B
```

To run MYPROG as a phantom process, enter the following at the system prompt:

```
>PHANTOM BACKGROUND
```

If a process issues a request for input that is not satisfied by DATA statements, UniVerse logs out the process.

Output from phantom processes is stored in the type1 file &PH&. Each phantom process creates a record in the &PH& file with a recordID in the following format:

```
phantom.verb_time_date
```

phantom.verb is the first word in command and time_date is a unique identifier based on the time and date the process was started. UniVerse directs terminal output to this record. If a phantom process does not produce any output, it creates an empty record. Delete an &PH& file record when you no longer need it, so the &PH& file does not become too large. (For more information, see &PH&.)

You can display the output of a phantom process from your terminal. For more information on displaying another users output, see the TANDEM command. Use STATUS ME to monitor the phantom process. The phantom process has the same user ID as your own.

If you are logged in when the phantom process finishes, UniVerse notifies you. If NOTIFY ON is set, the message appears on your screen immediately.

Otherwise the message appears when the next UniVerse prompt (>) appears. See NOTIFY for more information.

Use the LOGOUT command to stop a phantom process from the same terminal where you started it. A UniVerse Administrator can log out any phantom process on the system.

To make a phantom job the same as a job run from the command line, a phantom process runs the LOGIN paragraph or proc when it starts up.

When a phantom process runs your LOGIN paragraph which in turn runs a menu program, undesirable results can occur (when, for example, the phantom process produces a report).

You can structure your LOGIN paragraph to avoid these problems. Put commands to be executed by regular users and phantoms in the top section of the LOGIN paragraph. Follow this section with a command line that exits the LOGIN paragraph if it is being executed by a phantom process.

To check whether a phantom process is running the LOGIN paragraph, put the following line near the beginning of the paragraph:

```
IF @TTY = 'phantom' THEN GO END.OF.LOGIN
```

Put commands that phantoms should not execute in the last section of the LOGIN paragraph. The last line of the LOGIN paragraph must be the following label:

```
END.OF.LOGIN:
```

[9]

4.4 The Phantom output file

&PH&

Description

&PH& is a local type 1 file used to store the output produced by a task started with the PHANTOM command. The PHANTOM command starts a background process. PHANTOM automatically creates a &PH& file record every time you start a phantom process. All terminal output produced by the PHANTOM process is stored in the record. You can verify the operation of a phantom process after the process has completed by examining the &PH& file record. You can also edit or spool the &PH& file records.

When you no longer need an &PH& file record, delete it from the &PH& file.

Each record in the &PH& file has a record ID consisting of the following:

verb_time_date verb is a phantom verb or command. It is taken from a PHANTOM sentence and is the first word of the process to be executed as a phantom task. For example, the record ID is SORT_time_date for the following sentence:

PHANTOM SORT VENDOR

[9] IBM UniVerse User Reference

time is generated by the UniVerse BASIC INT function (TIME( )) which produces an integer indicating the time that the process started.

date is generated by the BASIC DATE function( ), the internal date. This suffix guarantees the uniqueness of &PH& file records in an account, even if several users begin the same phantom process at nearly the same time. To ensure unique recordIDs, use a SLEEP command between PHANTOM commands.

If &PH& does not exist in the account when you use PHANTOM, the command creates the &PH& file. &PH& is a type 1 file and can be accessed from UniVerse or from your operating system.
[10]

4.5 Paragraph Programming

4.5.1 Inline Prompting

Using Inline Prompts in Paragraphs

You can leave some values unspecified until the paragraph is invoked. The user supplies the missing value by responding to the prompt. This is the syntax for an inline prompt in paragraphs:

```
<< [ control, ] text [ , option ] >>
```

control is an option that controls the display of the in-line prompt.

text is the text of the prompt.

option verifies that the prompt response matches a specific pattern. It can be an input (ICONV) conversion code or a matching pattern. Enclose conversion codes in parentheses.

The following example displays the prompt Enter state repeatedly until the user presses Enter. The option 2A specifies that the user must enter two alphabetic characters.

```
<<R,Enter state,2A>>
```

All prompts are forgotten within menus unless you use the P control option. [11]

4.5.2 IF THEN Programming

The IF Command

The IF command introduces a conditional statement and changes the execution of the paragraph based on the result of an expression. The syntax is as follows:

IF expression THEN statements

The syntax of expression is as follows:

value1 operator value2

[10] IBM UniVerse User Reference
[11] IBM UniVerse System Description Manual

value1 and value2 can be constants, inline prompts, or @variables. A constant can be a number or a string. If the value includes spaces, enclose it in single or double quotation marks.

For a list of relational operators, see UniVerse BASIC. [12]

4.5.3 @variables

You can use the following @variables with the IF command:

```
@DATE
@DAY
@LOGNAME
@MONTH
@SYSTEM.RETURN.CODE
@TIME
@USERNO
@USER.RETURN.CODE
@WHO
@YEAR
```

[13]

[12] IBM UniVerse System Description Manual
[13] IBM UniVerse System Description Manual

5 UniVerse Knowledge Base

5.1 How to create a CRON to run a Universe jobs

Problem How to create a CRON to run a Universe Basic program/programs.
Solution

1. Check your Unix system man pages for your system's specifics on CRONs.
2. In this case the customer had Universe running on a Unix Sun Solaris operating System.
3. Log in as user authorized to run cron jobs. May be root.
4. Add an entry for the cron job by creating a cron file. For example: # crontab -e filename (this will put you into the vi editor for cron jobs). The entries in the cron file should be in the following format: min hour day/month month day-of-week command 0-59 0-23 1-31 1-12 0-6 shell file or command.

   ```
   Minute 0-59
   Hour 0-23
   Day of month 1-31
   Month 1-12
   Day of week 0-6
   ```

   Use a comman between multiple values, a hyphen to indicate a range, and an asterisk to indicate all possible values. For example of cron file entries:

   ```
    0 21 * * 1-5 /usr/local/bin/script.to.run

   The above entry will run the script Monday through Friday at 9 PM.

    21 4 1,7,9,21 * * /usr/local/bin/script.to.run

   The above entry will run the script on the 1, 7, 9 and 21st
   of each month at 4:21 AM.
   ```

   Note: it's recommended to use PHANTOM to run Universe Basic Program or programs.

5. vi the Unix shell script and add the following lines:

```
cd (go to the UniVerse account directory,
 where the Basic Program/programs are located)

/u1/uv/bin/uv << start (this command starts up UniVerse)
PHANTOM RUN BP basic program 1 (to run first basic program)
PHANTOM RUN BP basic program 2 (to run second basic program)
PHANTOM RUN BP basic program 3 (to run third basic program)
start (to end the command)
```

The script example above is using a technique known as "here" document in UNIX shell scripting. You may also create a single UniVerse paragraph to start all three PHANTOMS.

5.2 UniVerse background jobs and record locking

Discussions with IBM on the subject of cron jobs running UniVerse programs reveal that the PHANTOM structure should always be employed. Indications are that the locking process does not differentiate users if the cron process just executes a program. This can result in locks being released when you are expecting the lock to be in force. Also it is possible that the two jobs would not honor each others locks.

6 UNIX scripting examples

6.1 Script documentation

6.1.1 Description

The UNIX script comment flag is the #. Here is a sample of script documentation I will be using in the scripts for this document.

6.1.2 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   foo.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# additional documentation may be inserted here
# Syntax:       from $ enter ./foo.sh
```

6.2 Utilizing the UNIX syslog facility

6.2.1 Description

UNIX systems provide an event logging facility. For purposes of demonstration, my system is configured with a local log. Refer to your local system administrator for configuration the local logging conventions. The following script will place two entries in the system log.

6.2.2 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   sample1.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will demonstrate the use of the system logging facility
# Syntax:       from $ enter ./sample1.sh

logger -p local1.info "Logging the start of a script"

sleep 5

logger -p local1.info "The script finishes normally"
```

6.2.3 UNIX job initiation

```
[bob.cron@owl bin]$ ./sample1.sh
```

6.2.4 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 07:52:22 owl bob.cron: Logging the start of a script
Oct 29 07:52:27 owl bob.cron: The script finishes normally
```

6.3 Writing a job log file

6.3.1 Description

UNIX provides the facility of redirection of output. This allows a script to redirect all the output to a log file from standard output. Using this technique, the script will not send the output to your interactive session or to the mail when running as a cron job.

6.3.2 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   sample2.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will demonstrate the use of redirection for
#  output of commands to a log file.
# Syntax:       from $ enter ./sample2.sh

echo "Logging the start of a script ‘date‘" > ~/tmp/sample2.log

echo "The current working directory is ‘pwd‘" >> ~/tmp/sample2.log

echo "The script finishes normally ‘date‘" >> ~/tmp/sample2.log
```

6.3.3 UNIX job initiation

```
[bob.cron@owl bin]$ ./sample2.sh
```

6.3.4 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/sample2.log
Logging the start of a script Wed Oct 29 07:59:33 MDT 2008
The current working directory is /home/bob.cron/bin
The script finishes normally Wed Oct 29 07:59:33 MDT 2008
```

6.4 Capturing error messages and testing exit status

6.4.1 Description

UNIX scripting has two facilities that enhance your job execution.

The shell maintains two output streams, one for normal messages and the other for errors. Called stdin and stdout, you can capture both with a few tricks in your scripts.

The other facility is the exit status of each command. This status is 0 for successful execution and another number if an error was encountered.

The following script will document the use of both of these facilities.

6.4.2 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   sample3.sh
# Student:      Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will demonstrate capturing error messages and
# testing for return status
# Syntax:       from $ enter ./sample3.sh

echo "Logging the start of a script ‘date‘" > ~/tmp/sample3.log

# Note the nosuchdir is not a valid directory and will generate an error
# the 2>&1 syntax will redirect the error to the log
cp sample3.sh nosuchdir/sdf >> ~/tmp/sample3.log 2>&1

# The following will capture the exit status of the aborted copy command.
ExitStatus=$?

echo "The exit status is "$ExitStatus >> ~/tmp/sample3.log

if [ $ExitStatus -eq 0 ]
then
echo "The script finishes normally ‘date‘" >> ~/tmp/sample3.log
else
echo "The script ended with errors ‘date‘" >> ~/tmp/sample3.log
fi
```

6.4.3 UNIX job initiation

```
[bob.cron@owl bin]$ ./sample3.sh
```

6.4.4 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/sample3.log
Logging the start of a script Wed Oct 29 15:08:45 MDT 2008
cp: cannot create regular file ‘nosuchdir/sdf’: No such file or directory
The exit status is 1
The script ended with errors Wed Oct 29 15:08:45 MDT 2008
```

7 Understanding the UNIX environment

The ability to properly develop UNIX scripts, enter the UniVerse shell and achieve desired results requires an understanding of the UNIX environment. We will explore three different states of UNIX.

1. Interactive shell
2. Background batch jobs
3. Cron jobs

Each of these environments is subtly different and requires understanding. The following scripts will demonstrate the differences.

7.1 Test case 1.1 an interactive execution of a script

7.1.1 Description

The first case is the interactive environment. The script will write the environment variables to the job log.

7.1.2 Results discussion

The UNIX job log below contains the output of the env command. There are a couple of things to notice. First the environment is setup for an interactive session. Notice the variables LS_COLORS; TERM and INPUTRC. These indicate interactive session. Also notice the SHELL=/bin/bash. This is the interactive shell that will interpret all the commands you type. The PATH variable contains the path to the UniVerse bin directory. The SHLVL=2 indicates that this is a child process. If you enter env at the command prompt SHLVL will = 1.

7.1.3 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   testcase_1.1.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will save the current environment to a log file.
# Syntax:       from $ enter ./testcase_1.1.sh

logger -p local1.info "Start of a job testcase_1.1"
echo "Start of script ‘date‘" > ~/tmp/testcase_1.1.log

env >>  ~/tmp/testcase_1.1.log

logger -p local1.info "End of job testcase_1.1"
echo "testcase_1.1 finishes normally ‘date‘" >> ~/tmp/testcase_1.1.log
```

7.1.4 UNIX job initiation

```
[bob.cron@owl bin]$ ./testcase_1.1.sh
```

7.1.5 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 08:26:26 owl bob.cron: Start of a job testcase_1.1
Oct 29 08:26:26 owl bob.cron: End of job testcase_1.1
```

7.1.6 UNIX job log

Note: the LS_COLORS variable is <sniped> for brevity.

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_1.1.log
Start of script Wed Oct 29 08:26:26 MDT 2008
HOSTNAME=owl.maxbr.com
TERM=xterm
SHELL=/bin/bash
HISTSIZE=1000
QTDIR=/usr/lib/qt-3.3
QTINC=/usr/lib/qt-3.3/include
USER=bob.cron
LS_COLORS= <snip>
PATH=/usr/lib/qt-3.3/bin:/usr/kerberos/bin:/usr/lib/ccache:
/usr/local/bin:/bin:/usr/bin:/usr/ibm/uv/bin:/home/bob.cron/bin
MAIL=/var/spool/mail/bob.cron
PWD=/home/bob.cron/bin
INPUTRC=/etc/inputrc
KDE_IS_PRELINKED=1
LANG=en_US.UTF-8
KDEDIRS=/usr
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
HOME=/home/bob.cron
SHLVL=2
LOGNAME=bob.cron
CVS_RSH=ssh
QTLIB=/usr/lib/qt-3.3/lib
LESSOPEN=|/usr/bin/lesspipe.sh %s
DISPLAY=:0.0
G_BROKEN_FILENAMES=1
XAUTHORITY=/home/bob.cron/.xauthG5ZlA0
_=/bin/env
testcase_1.1 finishes normally Wed Oct 29 08:26:26 MDT 2008
```

7.2 Test case 1.1 running as an "at" job

7.2.1 Description

This is the same test case script initiated by the at command.

7.2.2 Results discussion

Notice the addition of the first echo message. This insures that the script will produce mail output. Notice the results are nearly the same. The sequence is a bit different. Notice the DISPLAY= variable is missing.

7.2.3 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   testcase_1.1.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will save the current environment to a log file.
# Syntax:       from $ enter ./testcase_1.1.sh
echo "Message to trigger mail"
logger -p local1.info "Start of a job testcase_1.1"
echo "Start of script ‘date‘" > ~/tmp/testcase_1.1.log

env >>  ~/tmp/testcase_1.1.log

logger -p local1.info "End of job testcase_1.1"
echo "testcase_1.1 finishes normally ‘date‘" >> ~/tmp/testcase_1.1.log
```

7.2.4 UNIX job initiation

```
[bob.cron@owl bin]$ at now < ~/bin/testcase_1.1.sh
job 5 at Wed Oct 29 11:51:00 2008
```

7.2.5 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 11:51:17 owl logger: Start of a job testcase_1.1
Oct 29 11:51:17 owl logger: End of job testcase_1.1
```

7.2.6 UNIX job log

Note: the LS_COLORS variable is <sniped> for brevity.

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_1.1.log
Start of script Wed Oct 29 11:51:17 MDT 2008
HOSTNAME=owl.maxbr.com
SHELL=/bin/bash
HISTSIZE=1000
QTDIR=/usr/lib/qt-3.3
QTINC=/usr/lib/qt-3.3/include
USER=bob.cron
LS_COLORS= <snip>
PATH=/usr/lib/qt-3.3/bin:/usr/kerberos/bin:/usr/lib/ccache:
```

```
/usr/local/bin:/bin:/usr/bin:/usr/ibm/uv/bin:/home/bob.cron/bin
MAIL=/var/spool/mail/bob.cron
PWD=/home/bob.cron/bin
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
KDE_IS_PRELINKED=1
KDEDIRS=/usr
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
SHLVL=2
HOME=/home/bob.cron
LOGNAME=bob.cron
QTLIB=/usr/lib/qt-3.3/lib
CVS_RSH=ssh
LESSOPEN=|/usr/bin/lesspipe.sh %s
G_BROKEN_FILENAMES=1
XAUTHORITY=/home/bob.cron/.xauthG5ZlA0
_=/bin/env
testcase_1.1 finishes normally Wed Oct 29 11:51:17 MDT 2008
```

7.2.7 UNIX mail

```
[bob.cron@owl bin]$ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/bob.cron": 1 message 1 new
>N  1 bob.cron@owl.maxbr.c  Wed Oct 29 11:51 16/657
 "Output from your job"
& 1
Message 1:
From bob.cron@owl.maxbr.com  Wed Oct 29 11:51:17 2008
Date: Wed, 29 Oct 2008 11:51:17 -0600
From: Cron Job Admin <bob.cron@owl.maxbr.com>
Subject: Output from your job        5
To: bob.cron@owl.maxbr.com

Message to trigger mail

&
```

7.3 Test case 1.1 run as a cron job

7.3.1 Description

Now take the same script and run it as a cron job.

7.3.2 Results discussion

Notice the abbreviated environment, reproduced below under UNIX job log. This is due to the nature of the cron facility. All UNIX implementations do not set up an interactive environment for these type of jobs.

Also note the PATH settings. They do not contain the UniVerse bin path our jobs will require.

Also note the SHELL=/bin/sh entry. This is not the shell specified in our password file entry. Nor is it the shell specified in the first line of our script. It so happens that the /bin/sh entry on a Linux server is linked to the bash shell. This will allow our scripts to function. Other systems may not be so friendly.

7.3.3 UNIX script

Notice the addition of the first echo message. This insures that the script will produce mail output.

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   testcase_1.1.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will save the current environment to a log file.
# Syntax:       from $ enter ./testcase_1.1.sh
echo "Message to trigger mail"
logger -p local1.info "Start of a job testcase_1.1"
echo "Start of script ‘date‘" > ~/tmp/testcase_1.1.log

env >>  ~/tmp/testcase_1.1.log

logger -p local1.info "End of job testcase_1.1"
echo "testcase_1.1 finishes normally ‘date‘" >> ~/tmp/testcase_1.1.log
```

7.3.4 UNIX job initiation

```
[bob.cron@owl bin]$ crontab -l
# Crontab entry for bob.cron
#
#
04 12 * * * /home/bob.cron/bin/testcase_1.1.sh
```

7.3.5 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 12:04:01 owl logger: Start of a job testcase_1.1
Oct 29 12:04:01 owl logger: End of job testcase_1.1
```

7.3.6 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_1.1.log
Start of script Wed Oct 29 12:04:01 MDT 2008
SHELL=/bin/sh
USER=bob.cron
PATH=/usr/bin:/bin
```

```
PWD=/home/bob.cron
HOME=/home/bob.cron
SHLVL=2
LOGNAME=bob.cron
_=/usr/bin/env
testcase_1.1 finishes normally Wed Oct 29 12:04:01 MDT 2008
```

7.3.7 UNIX mail

```
[bob.cron@owl bin]$ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/bob.cron": 1 message 1 new
>N  1 root@owl.maxbr.com    Wed Oct 29 12:04  23/887
   "Cron <bob.cron@owl> /"
& 1
Message 1:
From bob.cron@owl.maxbr.com  Wed Oct 29 12:04:02 2008
Date: Wed, 29 Oct 2008 12:04:01 -0600
From: root@owl.maxbr.com (Cron Daemon)
To: bob.cron@owl.maxbr.com
Subject: Cron <bob.cron@owl> /home/bob.cron/bin/testcase_1.1.sh
Content-Type: text/plain; charset=UTF-8
Auto-Submitted: auto-generated
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/home/bob.cron>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=bob.cron>
X-Cron-Env: <USER=bob.cron>

Message to trigger mail

&
```

7.4 Test case 1.2 establish our profile

7.4.1 Description

Test case 1.2 demonstrates the need for explicitly using the profile commands to set your environment exactly to the requirements of your script.

7.4.2 Results discussion

Notice that some of the interactive variables are back. The most important addition for us is the /usr/ibm/uv/bin to allow the execution of UniVerse commands.

7.4.3 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
```

```
# Program id:   testcase_1.2.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will save the current environment to a log file.
# The addition of the . commands request that the shell
# read and process our profile.
# Syntax:       from $ enter ./testcase_1.2.sh

. /etc/profile
. ~/.bash_profile

echo "Message to trigger mail"
logger -p local1.info "Start of a job testcase_1.2"
echo "Start of script `date`" > ~/tmp/testcase_1.2.log

env >>  ~/tmp/testcase_1.2.log

logger -p local1.info "End of job testcase_1.2"
echo "testcase_1.2 finishes normally `date`" >> ~/tmp/testcase_1.2.log
```

7.4.4 UNIX job initiation

```
[bob.cron@owl bin]$ crontab -l
# Crontab entry for bob.cron
#
#
50 12 * * * /home/bob.cron/bin/testcase_1.2.sh
```

7.4.5 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 12:50:02 owl logger: Start of a job testcase_1.2
Oct 29 12:50:02 owl logger: End of job testcase_1.2
```

7.4.6 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_1.2.log
Start of script Wed Oct 29 12:50:02 MDT 2008
HOSTNAME=owl.maxbr.com
SHELL=/bin/sh
HISTSIZE=1000
QTDIR=/usr/lib/qt-3.3
QTINC=/usr/lib/qt-3.3/include
USER=bob.cron
MAIL=/var/spool/mail/bob.cron
PATH=/usr/lib/qt-3.3/bin:/usr/kerberos/bin:/usr/lib/ccache:/usr/bin:
/bin:/usr/ibm/uv/bin:/home/bob.cron/bin
INPUTRC=/etc/inputrc
PWD=/home/bob.cron
```

```
LANG=en_US.UTF-8
KDE_IS_PRELINKED=1
KDEDIRS=/usr
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
HOME=/home/bob.cron
SHLVL=2
LOGNAME=bob.cron
QTLIB=/usr/lib/qt-3.3/lib
CVS_RSH=ssh
LESSOPEN=|/usr/bin/lesspipe.sh %s
G_BROKEN_FILENAMES=1
_=/usr/bin/env
testcase_1.2 finishes normally Wed Oct 29 12:50:02 MDT 2008
```

7.4.7 UNIX mail

```
[bob.cron@owl bin]$ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/bob.cron": 1 message 1 unread
>U  1 root@owl.maxbr.com    Wed Oct 29 12:50  24/897
  "Cron <bob.cron@owl> /"
& 1
Message 1:
From bob.cron@owl.maxbr.com  Wed Oct 29 12:50:02 2008
Date: Wed, 29 Oct 2008 12:50:02 -0600
From: root@owl.maxbr.com (Cron Daemon)
To: bob.cron@owl.maxbr.com
Subject: Cron <bob.cron@owl> /home/bob.cron/bin/testcase_1.2.sh
Content-Type: text/plain; charset=UTF-8
Auto-Submitted: auto-generated
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/home/bob.cron>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=bob.cron>
X-Cron-Env: <USER=bob.cron>

Message to trigger mail

&
```

8 UniVerse jobs

8.1 Test case 2.1

8.1.1 Description

Demonstrate the execution of a simple UniVerse verb.

8.1.2 Results discussion

Note the count of VOC items is found in the UNIX job log.

8.1.3 UNIX script

```
[bob.cron@owl bin]$ cat testcase_2.1.sh
#!/bin/bash
# Organization: Strategy 7
# Program id:   testcase_2.1.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will now execute a simple UniVerse verb to count
# the number of items in the VOC file
# Syntax:       from $ enter ./testcase_2.1.sh

. /etc/profile
. ~/.bash_profile

echo "Message to trigger mail"
logger -p local1.info "Start of a job testcase_2.1"
echo "Start of script `date`" > ~/tmp/testcase_2.1.log

env >>  ~/tmp/testcase_2.1.log

cd /home/BOB.CRON
echo "Current directory `pwd`" >>  ~/tmp/testcase_2.1.log
/usr/ibm/uv/bin/uv "COUNT VOC" >>  ~/tmp/testcase_2.1.log


logger -p local1.info "End of job testcase_2.1"
echo "testcase_2.1 finishes normally `date`" >> ~/tmp/testcase_2.1.log
```

8.1.4 UniVerse Paragraph

The UniVerse login paragraph has been removed for this test.

8.1.5 UNIX job initiation

```
# Crontab entry for bob.cron
#
#
18 13 * * * /home/bob.cron/bin/testcase_2.1.sh
```

8.1.6 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 13:18:01 owl logger: Start of a job testcase_2.1
Oct 29 13:18:01 owl logger: End of job testcase_2.1
```

8.1.7 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_2.1.log
Start of script Wed Oct 29 13:18:01 MDT 2008
HOSTNAME=owl.maxbr.com
SHELL=/bin/sh
HISTSIZE=1000
QTDIR=/usr/lib/qt-3.3
QTINC=/usr/lib/qt-3.3/include
USER=bob.cron
MAIL=/var/spool/mail/bob.cron
PATH=/usr/lib/qt-3.3/bin:/usr/kerberos/bin:/usr/lib/ccache:/usr/bin:
/bin:/usr/ibm/uv/bin:/home/bob.cron/bin
INPUTRC=/etc/inputrc
PWD=/home/bob.cron
LANG=en_US.UTF-8
KDE_IS_PRELINKED=1
KDEDIRS=/usr
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
HOME=/home/bob.cron
SHLVL=2
LOGNAME=bob.cron
QTLIB=/usr/lib/qt-3.3/lib
CVS_RSH=ssh
LESSOPEN=|/usr/bin/lesspipe.sh %s
G_BROKEN_FILENAMES=1
_=/usr/bin/env
Current directory /home/BOB.CRON

782 records counted.
testcase_2.1 finishes normally Wed Oct 29 13:18:02 MDT 2008
```

8.1.8 UNIX mail

```
[bob.cron@owl bin]$ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/bob.cron": 1 message 1 new
```

```
>N  1 root@owl.maxbr.com    Wed Oct 29 13:18  23/887
   "Cron <bob.cron@owl> /"
& 1
Message 1:
From bob.cron@owl.maxbr.com  Wed Oct 29 13:18:02 2008
Date: Wed, 29 Oct 2008 13:18:01 -0600
From: root@owl.maxbr.com (Cron Daemon)
To: bob.cron@owl.maxbr.com
Subject: Cron <bob.cron@owl> /home/bob.cron/bin/testcase_2.1.sh
Content-Type: text/plain; charset=UTF-8
Auto-Submitted: auto-generated
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/home/bob.cron>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=bob.cron>
X-Cron-Env: <USER=bob.cron>

Message to trigger mail
```

8.2 Test case 2.2 adding error logging and status testing

8.2.1 Description

This test case adds the error logging and status testing to our UniVerse script.

8.2.2 Results discussion

The output of our report is being captured in the log file. There are may other things we can do now that we are successfully inside UniVerse.

8.2.3 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   testcase_2.2.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# This script will now execute a simple UniVerse verb to count
# the number of items in the VOC file
# Added to the script is testing for exit status and error capturing.
# Syntax:       from $ enter ./testcase_2.2.sh

. /etc/profile
. ~/.bash_profile

echo "Message to trigger mail"
logger -p local1.info "Start of a job testcase_2.2"
echo "Start of script `date`" > ~/tmp/testcase_2.2.log
```

```
cd /home/BOB.CRON
echo "Current directory `pwd`" >>  ~/tmp/testcase_2.2.log
/usr/ibm/uv/bin/uv "COUNT VOC" >>  ~/tmp/testcase_2.2.log 2>&1
ExitStatus=$?
echo "Exit status is $ExitStatus" >> ~/tmp/testcase_2.2.log

if [ $ExitStatus -eq 0 ]
then
logger -p local1.info "End of job testcase_2.2"
echo "testcase_2.2 finishes normally `date`" >> ~/tmp/testcase_2.2.log
else
logger -p local1.info "Job testcase_2.2 ended with errors"
echo "testcase_2.2 finishes with errors `date`" >> ~/tmp/testcase_2.2.log
fi
```

8.2.4 UniVerse Paragraph

The UniVerse login paragraph has been removed for this test.

8.2.5 UNIX job initiation

```
[bob.cron@owl bin]$ crontab -l
# Crontab entry for bob.cron
#
#
24 15 * * * /home/bob.cron/bin/testcase_2.2.sh
```

8.2.6 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 15:24:01 owl logger: Start of a job testcase_2.2
Oct 29 15:24:01 owl logger: End of job testcase_2.2
```

8.2.7 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_2.2.log
Start of script Wed Oct 29 15:24:01 MDT 2008
Current directory /home/BOB.CRON

782 records counted.
Exit status is 0
testcase_2.2 finishes normally Wed Oct 29 15:24:01 MDT 2008
```

8.2.8 UNIX mail

```
[bob.cron@owl bin]$ mail
Mail version 8.1 6/6/93.  Type ? for help.
```

```
"/var/spool/mail/bob.cron": 1 message 1 new
>N  1 root@owl.maxbr.com    Wed Oct 29 15:24  23/887
   "Cron <bob.cron@owl> /"
& 1
Message 1:
From bob.cron@owl.maxbr.com  Wed Oct 29 15:24:01 2008
Date: Wed, 29 Oct 2008 15:24:01 -0600
From: root@owl.maxbr.com (Cron Daemon)
To: bob.cron@owl.maxbr.com
Subject: Cron <bob.cron@owl> /home/bob.cron/bin/testcase_2.2.sh
Content-Type: text/plain; charset=UTF-8
Auto-Submitted: auto-generated
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/home/bob.cron>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=bob.cron>
X-Cron-Env: <USER=bob.cron>

Message to trigger mail
```

8.2.9 Return status

The return status was 0 indicating a successful job.

8.3 Test case 2.2 UniVerse LOGIN paragraph prompting for input

8.3.1 Description

Now we add a UniVerse LOGIN paragraph with prompting. This will prevent our command from being executed.

8.3.2 Results discussion

The problem is that UniVerse returns a 0 exit status. This is of no use to tell if the job functioned properly.

The job did not function properly due to the presence of an input request. The COUNT was not executed.

8.3.3 UNIX script

8.3.4 UniVerse Paragraph

```
>CT VOC LOGIN

     LOGIN
0001 PA Paragraph to do simple prompting
```

```
0002 * Organization: Strategy 7
0003 * Program id:   LOGIN
0004 * Author:      Robert Woods
0005 * Date written: 10/29/08
0006 * Purpose:      Paragraph executed when UniVerse shell is opened
0007 DISPLAY <<A,WHO ARE YOU?>>

* Following is a test of the script

>LOGIN
WHO ARE YOU?=BOB WOODS
BOB WOODS
```

8.3.5 UNIX job initiation

```
[bob.cron@owl bin]$ crontab -l
# Crontab entry for bob.cron
#
#
48 15 * * * /home/bob.cron/bin/testcase_2.2.sh
```

8.3.6 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 15:48:01 owl logger: Start of a job testcase_2.2
Oct 29 15:48:02 owl logger: End of job testcase_2.2
```

8.3.7 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_2.2.log
Start of script Wed Oct 29 15:48:01 MDT 2008
Current directory /home/BOB.CRON
WHO ARE YOU?=Exit status is 0
testcase_2.2 finishes normally Wed Oct 29 15:48:02 MDT 2008
```

8.3.8 UNIX mail

```
[bob.cron@owl bin]$ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/bob.cron": 1 message 1 new
>N  1 root@owl.maxbr.com    Wed Oct 29 15:48  23/887
   "Cron <bob.cron@owl> /"
& 1
Message 1:
From bob.cron@owl.maxbr.com  Wed Oct 29 15:48:02 2008
Date: Wed, 29 Oct 2008 15:48:01 -0600
From: root@owl.maxbr.com (Cron Daemon)
To: bob.cron@owl.maxbr.com
Subject: Cron <bob.cron@owl> /home/bob.cron/bin/testcase_2.2.sh
```

```
Content-Type: text/plain; charset=UTF-8
Auto-Submitted: auto-generated
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/home/bob.cron>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=bob.cron>
X-Cron-Env: <USER=bob.cron>

Message to trigger mail
```

8.3.9 Return status

The UniVerse shell we invoked returned a 0 exit status even when there was a prompt for input from a batch job. The COUNT verb was not executed.

8.4 Test case 2.3 Testing other UniVerse errors

8.4.1 Description

8.4.2 Results discussion

8.4.3 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   testcase_2.3.sh
# Author:       Robert Woods
# Date written: 10329/08
# Purpose:      This script was written for demonstration
# This script will execute an invalid UniVerse verb
# Syntax:       from $ enter ./testcase_2.3.sh

. /etc/profile
. ~/.bash_profile

echo "Message to trigger mail"
logger -p local1.info "Start of a job testcase_2.3"
echo "Start of script `date`" > ~/tmp/testcase_2.3.log

cd /home/BOB.CRON
echo "Current directory `pwd`" >>  ~/tmp/testcase_2.3.log
/usr/ibm/uv/bin/uv "COUNTT VOC" >>  ~/tmp/testcase_2.3.log 2>&1
ExitStatus=$?
echo "Exit status is $ExitStatus" >> ~/tmp/testcase_2.3.log

if [ $ExitStatus -eq 0 ]
then
logger -p local1.info "End of job testcase_2.3"
```

```
echo "testcase_2.3 finishes normally `date`" >> ~/tmp/testcase_2.3.log
else
logger -p local1.info "Job testcase_2.3 ended with errors"
echo "testcase_2.3 finishes with errors `date`" >> ~/tmp/testcase_2.3.log
fi
```

8.4.4 UniVerse Paragraph

```
The UniVerse login paragraph has been removed for this test.
```

8.4.5 UNIX job initiation

```
[bob.cron@owl bin]$ crontab -l
# Crontab entry for bob.cron
#
#
57 15 * * * /home/bob.cron/bin/testcase_2.3.sh
```

8.4.6 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 15:57:01 owl logger: Start of a job testcase_2.3
Oct 29 15:57:01 owl logger: End of job testcase_2.3
```

8.4.7 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_2.3.log
Start of script Wed Oct 29 15:57:01 MDT 2008
Current directory /home/BOB.CRON
Verb "COUNTT" is not in your VOC.
Exit status is 0
testcase_2.3 finishes normally Wed Oct 29 15:57:01 MDT 2008
```

8.4.8 UNIX mail

```
[bob.cron@owl bin]$ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/bob.cron": 1 message 1 new
>N  1 root@owl.maxbr.com    Wed Oct 29 15:57  23/887
   "Cron <bob.cron@owl> /"
& 1
Message 1:
From bob.cron@owl.maxbr.com  Wed Oct 29 15:57:01 2008
Date: Wed, 29 Oct 2008 15:57:01 -0600
From: root@owl.maxbr.com (Cron Daemon)
To: bob.cron@owl.maxbr.com
Subject: Cron <bob.cron@owl> /home/bob.cron/bin/testcase_2.3.sh
Content-Type: text/plain; charset=UTF-8
Auto-Submitted: auto-generated
X-Cron-Env: <SHELL=/bin/sh>
```

```
X-Cron-Env: <HOME=/home/bob.cron>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=bob.cron>
X-Cron-Env: <USER=bob.cron>

Message to trigger mail
```

8.4.9 Return status

The return status is again 0. UniVerse does not return the error to the UNIX shell. If you want to test for the presence of a UniVerse error, you would need to write code to test for the errors.

8.5 Test case 2.4 enhanced LOGIN paragraph

8.5.1 Description

The UniVerse Login paragraph has been enhanced to test @LOGNAME against a known cron user.

8.5.2 Results discussion

The COUNT is executed.

8.5.3 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   testcase_2.4.sh
# Author:       Robert Woods
# Date written: 10/29/08
# Purpose:      This script was written for demonstration
# Executing the UniVerse COUNT.
# Removed the status checking.
# Syntax:       from $ enter ./testcase_2.4.sh

. /etc/profile
. ~/.bash_profile

echo "Message to trigger mail"
logger -p local1.info "Start of a job testcase_2.4"
echo "Start of script `date`" > ~/tmp/testcase_2.4.log

cd /home/BOB.CRON
echo "Current directory `pwd`" >>  ~/tmp/testcase_2.4.log
/usr/ibm/uv/bin/uv "COUNT VOC" >>  ~/tmp/testcase_2.4.log 2>&1

logger -p local1.info "End of job testcase_2.4"
```

```
echo "testcase_2.4 finishes normally ‘date‘" >> ~/tmp/testcase_2.4.log
```

8.5.4 UniVerse Paragraph

```
>CT VOC LOGIN

     LOGIN
0001 PA Paragraph to test for cron user and skip prompting
0002 * Organization: Strategy 7
0003 * Program id:   LOGIN
0004 * Author:      Robert Woods
0005 * Date written: 10/29/08
0006 * Purpose:      Paragraph executed when UniVerse shell is opened
0007 IF @LOGNAME = "bob.cron" THEN GO DOCRONJOB
0008 DISPLAY <<A,WHO ARE YOU?>>
0009 DOCRONJOB:

* Execute the LOGIN paragraph.

>LOGIN
WHO ARE YOU?=BOB WOODS
BOB WOODS

* I am prompted because I am not logged in as bob.cron

>WHO
19 BOB.CRON From bob.cron
```

8.5.5 UNIX job initiation

```
[bob.cron@owl bin]$ crontab -l
# Crontab entry for bob.cron
#
#
19 16 * * * /home/bob.cron/bin/testcase_2.4.sh
```

8.5.6 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 29 16:19:01 owl logger: Start of a job testcase_2.4
Oct 29 16:19:01 owl logger: End of job testcase_2.4
```

8.5.7 UNIX job log

```
[bob.cron@owl bin]$ cat ~/tmp/testcase_2.4.log
Start of script Wed Oct 29 16:19:01 MDT 2008
Current directory /home/BOB.CRON
```

```
782 records counted.
testcase_2.4 finishes normally Wed Oct 29 16:19:01 MDT 2008
You have new mail in /var/spool/mail/bob.cron
```

8.5.8 UNIX mail

```
[bob.cron@owl bin]$ mail
Mail version 8.1 6/6/93.  Type ? for help.
"/var/spool/mail/bob.cron": 1 message 1 new
>N  1 root@owl.maxbr.com    Wed Oct 29 16:19  23/887
   "Cron <bob.cron@owl> /"
& 1
Message 1:
From bob.cron@owl.maxbr.com  Wed Oct 29 16:19:01 2008
Date: Wed, 29 Oct 2008 16:19:01 -0600
From: root@owl.maxbr.com (Cron Daemon)
To: bob.cron@owl.maxbr.com
Subject: Cron <bob.cron@owl> /home/bob.cron/bin/testcase_2.4.sh
Content-Type: text/plain; charset=UTF-8
Auto-Submitted: auto-generated
X-Cron-Env: <SHELL=/bin/sh>
X-Cron-Env: <HOME=/home/bob.cron>
X-Cron-Env: <PATH=/usr/bin:/bin>
X-Cron-Env: <LOGNAME=bob.cron>
X-Cron-Env: <USER=bob.cron>

Message to trigger mail
```

8.6 Test case 2.5 UniVerse PHANTOM

8.6.1 Description

This test pulls all the questions into one example. Using cron to start the job. Using the UniVerse PHANTOM command to start a job.

8.6.2 Results discussion

The COUNT is executed by the PHANTOM command and the output is written to the &PH& item.

8.6.3 UNIX script

```
#!/bin/bash
# Organization: Strategy 7
# Program id:   testcase_2.5.sh
# Author:       Robert Woods
# Date written: 10/31/08
# Purpose:      This script was written for demonstration
# Executing the UniVerse PHANTOM command.
# Removed the status checking.
```

```
# Removed the trigger for mail
# Syntax:       from $ enter ./testcase_2.5.sh

. /etc/profile
. ~/.bash_profile

logger -p local1.info "Start of a job testcase_2.5"
echo "Start of script `date`" > ~/tmp/testcase_2.5.log

cd /home/BOB.CRON
echo "Current directory `pwd`" >>  ~/tmp/testcase_2.5.log
/usr/ibm/uv/bin/uv "PHANTOM COUNT VOC" >>  ~/tmp/testcase_2.5.log 2>&1

logger -p local1.info "End of job testcase_2.5"
echo "testcase_2.5 finishes normally `date`" >> ~/tmp/testcase_2.5.log
```

8.6.4 UniVerse Paragraph

```
>CT VOC LOGIN

     LOGIN
0001 PA Paragraph to test for cron user and skip prompting
0002 * Organization: Strategy 7
0003 * Program id:   LOGIN
0004 * Author:      Robert Woods
0005 * Date written: 10/29/08
0006 * Purpose:      First paragraph executed when entering UniVerse
0007 IF @LOGNAME = "bob.cron" THEN GO DOCRONJOB
0008 DISPLAY <<A,WHO ARE YOU?>>
0009 DOCRONJOB:
```

8.6.5 UNIX job initiation

```
[bob.cron@owl bin]$ crontab -l
# Crontab entry for bob.cron
#
#
23 12 * * * /home/bob.cron/bin/testcase_2.5.sh
```

8.6.6 UNIX logging

```
[bob.cron@owl ~]$ tail -2 /var/log/local1.log
Oct 31 12:23:01 owl logger: Start of a job testcase_2.5
Oct 31 12:23:02 owl logger: End of job testcase_2.5
```

8.6.7 UNIX job log

```
23 12 * * * /home/bob.cron/bin/testcase_2.5.sh
[bob.cron@owl bin]$ cat ~/tmp/testcase_
```

```
testcase_1.1.log  testcase_2.1.log  testcase_2.3.log  testcase_2.5.log
testcase_1.2.log  testcase_2.2.log  testcase_2.4.log
[bob.cron@owl bin]$ cat ~/tmp/testcase_2.5.log
Start of script Fri Oct 31 12:23:01 MDT 2008
Current directory /home/BOB.CRON
Creating file "&PH&" as Type 1.
Creating file "D_&PH&" as Type 3, Modulo 1, Separation 2.
Added "@ID", the default record for RetrieVe, to "D_&PH&".
Phantom process started with process ID 5332.
testcase_2.5 finishes normally Fri Oct 31 12:23:02 MDT 2008
```

8.6.8 UNIX mail

```
[bob.cron@owl bin]$ mail
No mail for bob.cron
```

8.6.9 UniVerse PHANTOM Record

```
LIST &PH& 12:25:35pm  31 Oct 2008  PAGE    1
&PH&.........................

COUNT_44582_14915

1 records listed.
>CT &PH& *

     COUNT_44582_14915
0001
0002 782 records counted.
```