Scheduling UniVerse jobs with UNIX tools

Robert F. Woods
Strategy 7 Corporation

10/28/08

Contents

1 Introduction

1.1 Statement of problem

The uinVerse 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

2 UniVerse Tools

2.1 UniVerse Login Entry

In the UniVerse environment, the system administrator can set systemwide 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. [1]

2.2 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. [2]

2.3 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

There is a systemwide 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\#.
```

Description

There is a systemwide 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\#.
```

[1] IBM UniVerse System Description Manual

[2] IBM UniVerse User Reference

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 user.s 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:
```

[3]

2.4 &PH&

&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

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.

[4]

[3] IBM UniVerse User Reference

[4] IBM UniVerse User Reference

2.5 Paragraph Programming

2.5.1 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. [5]

2.5.2 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

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. [6]

2.5.3 @variables

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

```
@DATE
@DAY
@LOGNAME
@MONTH
@SYSTEM.RETURN.CODE
```

[5] IBM UniVerse System Description Manual
[6] IBM UniVerse System Description Manual

```
@TIME
@USERNO
@USER.RETURN.CODE
@WHO
@YEAR
```

[7]

3 UniVerse Knowledge Base

3.1 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 1-59 1-12 1-12 1-12 0-6 shell file or command For example of cron file entry: * 20 * * 1-5 ... * 21 * * 1-5 path/ shell script 3. Note: it's recommended to use PHANTOM to run Universe Basic Program or programs. 4. 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.

[7] IBM UniVerse System Description Manual

4 Junk

4.1 junk

```
redirect 2$>$/dev/console
who > /tmp/logfile.txt
```

8

[8] Unix in a Nutshell, Third Edition.; by Arnold Robbins