Calling Windows (and Unix) programs using SHCMD$

1. Introduction

This note describes the SHCMD$ subroutine which will provide a method for a Global Cobol or Global Speedbase program to initiate a program running on the host operating system. The host o/s command may be specified explicitly or indirectly, via a Schema File.

Two methods of initiating a host o/s program from within Global System Manager are possible. The first method, using SVC 70, is only available on some Global System Manager configurations:

Configuration	*Is SVC-70 available?*
GSM (BOS)	No
GSM (Unix)	Yes
GSM (DOS)	No
GSM (Novell)	No
GSM (Win NT)	Yes

In the context of this design, the SVC-70 interface will be referred to as the "Server interface".

The second method, using a special escape sequence is only available when the program is running on the Global Windows Workstation terminal emulator (i.e. when $$GUI = 1). In the context of this design, the escape sequence interface will be referred to as the "Desktop interface".

2. User Interface

2.1 SHCMD$

The SHCMD$ subroutine will be called in the following way:

CALL SHCMD$ USING *opcode prompt fd*

or:

CALL SHCMD$ USING *opcode prompt string [ex*]

opcode is a PIC 9(4) COMP field or literal which can be set to the following values:

0	Use schema file (see below)
1	Use Esc Seq to display string (Desktop)
2	Use SVC-70 to display string (Server)

3 desktop command 254 bytes
4 server command 254 bytes

prompt is a PIC 9(4) COMP field or literal which can be set to 1 to indicate that 'Key <CR> to continue' should be displayed before refreshing the screen when the Unix shell returns control to GSM, or 0 to refresh the screen immediately. This field has no effect when the string is displayed in "Desktop mode";

fd is the closed FD of the schema file;

string is the command string terminated by a byte of binary zero. **The length of the command string + the terminator byte must not exceed 79 characters.**

ex is an optional extension block for Win NT systems and has the following structure:

```
01   EX
 02  EXSHOW       PIC 9(2) COMP    * Window display flag
                                   * 0 - Normal Window
                                   * 1 - Maximised Window
                                   * 2 - Minimised Window
                                   * 3 - Minimised Window in background
 02  EXWAIT       PIC 9(2) COMP    * 0 return control immediately
                                   * 1 Wait until window command
                                   * completes
```

The following exception will be returned from SVC 70:

$$COND = 1 I/O error on schema file
$$COND = 2 Schema file not found
$$COND = 3 Invalid version of GSM (i.e. SVC-70 not available)
$$COND = 4 Invalid schema file or command parameter
$$COND = 5 In use by another partition on this system
$$COND = 6 Command string too long
$$COND = 7 Command failed
$$COND = 8 Not running on the GUI (i.e. $$GUI = 0)
$$COND = 9 Unable to return command exit code

$$CRES will contain the result of the command if issued.

Note that for a desktop command error conditions 5, 6, and 7 will not be returned but will be handled by the GUI (or will be ignored completely e.g. the GUI may never know that a command has failed).

2.2 $SCHEMA

A new menu entry in $SCHEMA (see SJ138) will allow the creation of command schema files. The $SCHEMA dialogue is very simple: It will prompt for the type of Schema File (i.e. Server or Desktop) and then for the command string.

3. Internal design

3.1 $SCHEMA

A new function will be added to the $SCHEMA program which will allow schema files for SHCMD$ to be produced. This schema file will be as follows:

```
01      SC
02      SCTYP                PIC 9(4) COMP       * 0 desktop, 1 server
02      SHCMD       PIC X(79)            * always terminated with#00
02      SCSHOW      PIC 9(2) COMP        * Windows NT
                                         * 0 Normal Window
                                         * 1 Maximised Window
                                         * 2 Minimised Window
                                         * 3 Minimised Window in background
    02   SCWAIT     PIC 9(2) COMP        * 0 Return control immediately
                                         * 1 Return control when command
                                         * completes
    02   FILLER     PIC X(498)
                    VALUE LOW-VALUES
```

The schema file will have file of a new type SI, organisation 117.

Since $SCHEMA cannot tell the difference between a Unix command file and a Windows NT command file it will prompt to ask if you want to update the Windows NT parameters. Keying <CTRL C> to either of the window parameter prompts will describe the values of the flags available.

3.2 SHCMD$

The Desktop interface will pass the command string to the GUI in the following way:-

send bypass-on sequence
send #1B73 string (terminated by CR)
send bypass-off, sequence

The Server interface will use the SVC-70 interface as documented in the V8.1 Global Operating Manual (Unix) and GSM (Windows) manuals.

Appendix A Extensions for long command strings

A.1 Introduction

This note describes the changes required to SHCMD$ and related routines and programs to allow command string of 254 bytes (including the terminator) as well as the current length of 79.

A.2 SHCMD$

SHCMD$ needs to recognise schema files with the two new TBTYPE fields 3 and 4 as described below and expect 254 byte command strings for these format Schema Files. It must also move up to 254 bytes from a passed string (rather than the current 79 bytes) into its work area.

A.3 SCHEMA$

Two new file types will be added to the TBTYPE field in SCHEMA$:

3	desktop command 254 bytes
4	server command 254 bytes

In both these cases the TBCMD string expected will be 254 bytes.

A.4 $SCHEM/$SFMAIN

If a new command schema file is to be created (rather than an update of an existing command schema file) then the user must be asked if a 254 byte command string is required.

If a command string of more than 78 characters is to be accepted then these programs must accept it over more than a single line.

The appropriate type (as above) must be placed in the schema file.