Non-Global Customisation of $MONITOR

1. Introduction

The $MONITOR file contains a number of vital GSM customisations that are applied at BACRES generation time and, later, at SYSRES installation time. These customisations are applied using the $F PAM, PUF and PIP commands. The results of these customisations can be displayed using the $MNDISP utility.

The customisations are applied to 2 sections of the $MONITOR file. The first set of customisation are applied to the CP-block which is located at a fixed offset at the start of the program file. The second set of customisations are applied to the ULADE field within the directory label.

$MONITOR is customised **AFTER** it has been created as the result of a standard $COBOL/$LINK development process. Furthermore, the inclusion of a non-customised version of $MONITOR will prevent either a GSM Starter system (i.e. BACRES) or a GSM Target system (i.e. SYSRES) from loading. Thus, it is not possible to simply copy a nascent $MONITOR to a live SYSRES (or BACRES).

This problem is well understood. For example, the $CUSUPD process, which upgrades external versions of GSM without the need for a full installation, explicitly re-customises the upgrade $MONITOR after copying it from BACRES to SYSRES. Nevertheless, the fact that it is not possible to simply copy a new $MONITOR and P.$MON combination to a live SYSRES without the need for subsequent (and potentially dangerous) $F PAM, PIP and PUF commands, makes the upgrade to a new $MONITOR a tedious and error-prone process. In most cases, if anything goes wrong with the re-customisation it is no longer possible to load GSM from SYSRES.

This document describes a technique for GSM (Windows NT) that allows the customisable options in $MONITOR to be held outside the $MONITOR file and/or label. Thus, once these customisations have been enabled, a freshly linked $MONITOR can be copied to SYSRES without the need to re-apply the customisations using the obscure $F commands.

Although this technique applies only to GSM (Windows NT) it could also be applied to GSM (Unix), GSM (Novell) and GSM (DOS) by replacing the use of the Windows registry by a simple host o/s text file.

Note that this technique is independent of the changes to "Global Software Generation, Distributions and Installation" (see IN183) and the changes to hold all Global files in the

native directory format (see W0-484). Although in the context of W0-484, the $MONITOR will be considered a "type B" file (i.e. it will consist of a 256 byte header, containing the Global directory label information) thus the ULADE fields will be emulated in the host o/s version of this file.

2. Technical Details

All the changes are confined to the "sr10" function of the Steering Routine. Just after this function has read the $MONITOR label, the registry is interrogated to determine if any of the fields in the label (see below) are to be overwritten by customised options. Similarly, just after this function has read the $MONITOR data, the registry is interrogated to determine if any of the fields in the CP-block, at the start of the data section of the file (see below) are to be overwritten by customised options.

A new sub-key "Monitor" has been added to the "..\Global\Client" key of the registry. The following ValueNames are defined under the "..\Global\Client\Monitor" key:

$MONITOROverride	This flag MUST be set to "On" etc. to enable the following options. Set this flag to "Off" etc. to temporarily disable the new customisations;
NumberOfMemoryPages	This numeric option updates the MNPA field in the $MONITOR label. This option is displayed by $MNDISP as: **Number of Memory Pages**
CommandUnit	This 3-character string option updates the CPUNIT field in the CP-block. This option is displayed by $MNDISP as: **Command Unit**
NumberOfMemoryBanks	This numeric option updates the MNBB field in the $MONITOR label. This option is displayed by $MNDISP as the *x* in: **Number of Memory Banks *x* of *y* Kbytes**
SizeOfMemoryBanksKb	This numeric option updates the MNBA field in the $MONITOR label. This option is displayed by $MNDISP as the *y* in: **Number of Memory Banks *x* of *y* Kbytes**

SystemStackSize — This numeric option updates the MNST field in the $MONITOR label. This option is displayed by $MNDISP as:

System Stack Size

Unit100AliasedTo — This 3-character string option updates the CP-100 field in the CP-block. This option is displayed by $MNDISP as:

Unit 100 aliased to

StartIPLUnit — This 3-character string option updates the CPIPL field in the CP-block. This option is displayed by $MNDISP as:

Start IPL unit

UserFileUnit — This 3-character string option updates the CPUFUN field in the CP-block. This option is displayed by $MNDISP as:

User File Unit

TargetSystem — This 1-character string option updates the CPTORS field in the CP-block (see below). This option is displayed by $MNDISP as one of:

Starter system
Target system
Diagnostics enabled
Bootstrap message display enabled

SoftwareNodeID — This 1-character string option updates the PLLNID field in the CP-block. This option is displayed by $MNDISP as one of:

Software node-id = xx
Hardware node-id

Note that the customisations are NOT applied if the $MONTOR is for a Starter System (i.e. if CPTORS = "S").

No validation is applied to any of the numeric or 3-character string values. The "TargetSystem" must be one of either "D" (Diagnostics enabled), "T" (Target system), "S" (Starter system) or <SPACE> (Bootstrap message display enabled).

Note that the ..\Client\Software\Monitor\NumberOfMemoryPages value is equivalent to the existing ..\Client\Software\Nucleus\MaximumMemoryPages value. If both are present in the registry, the latter will over-ride the former.