# Simplify CMNBAT90 - SERCOPY usage to activate components

- [SCMBeans] |
- [ZMF Administrator] |
- [1.0-Orval]

Release info**Release date:** Mon, 24/10/2011
**Applicable ChangeMan ZMF:** Any ChangeMan ZMF release

Staging processes for like source components typically produce one or more staging outputs (so called 'ILOD' related components). For each library type, this is done via 2 jobsteps that perform PGM=CMNBAT90, followed by a jobstep with PGM=SERCOPY.

Using PGM=CMNBAT90 is not obvious, because of its SYSIN cards that need to be set correctly. While PGM=SERCOPY also has a few features that are very handy, but not commonly known.

Also, PGM=CMNBAT90 seems to be pretty sensitive to ChangeMan ZMF release changes (output DSNs change, extra DDNAMEs may be added, etc). Not to speak about incorrect usages of PGM=CMNBAT90, which may even lead to invalid, missing or corrupted info in some CMNPMAST records.

Here is an illustration of the kind of problems you may run into if PGM=CMNBAT90 is not correctly used (customized): in some ChangeMan ZMF release (some 5.x version), some of the jobsteps with PGM=CMNBAT90 all of a sudden also needed an extra SYSLIB-ddcard (constructed via embed of CMN$$SYL), which should create a SYSLIB concatenation that exactly matches the SYSLIB concatenation of the LINK step (typically with PGM=IEWL). If that extra SYSLIB would not be available in the PGM=CMNBAT90 jobstep, or with a concatenation that is different from the one in the LINK step, your audit results might be incorrect (missing SYNCHx-situations, invalid SYNCHx-flags, etc). To make it even more difficult: imagine you'd be facing the upgrade to that 5.x version (where this new SYSLIB-ddcard first came up):

- How would you REALIZE that there was a new SYSLIB needed in jobsteps with PGM=CMNBAT90 (was it documented in the 5.x release upgrade docu?)?
- How would you FIND OUT where in your own customizations you also had to add this extra SYSLIB-ddcard (different from the customized CMN$$LNK, that one was easy to realize ...)?
- How many customized skeletons would you have to correct to adapt to this new feature?

What is needed, is something to simplify the skeleton coding related to the creation of source-to-load relationships and copying the staging outputs to the target staging libraries, something where you just have to specify the values of a few parms, like:

- The DSN containing the temporary staging output (e.g. &&&&LOAD as in the vendor CMN$$LNK skeleton).
- The target library type to used to create the source-to-load relationship (it could be &TLODTLP, but also just 'DBR', or any library type you'd want).

And to make all this AbitMORE (oeps) challenging, wouldn't it be nice if the staging job would NOT run into a JCL error (target staging DSN not found), in case the staging library for the selected target library type has not (yet) been allocated. Something that typically happens when you introduce a new staging output library type in the skeleton logic, but you did not (yet) customize CMNEX026


Copyright © 2000-2025 AbitMORE ® BV. All rights reserved

accordingly. Instead of debugging such JCL error, it'd be much more easy if there was some type of error message (and an RC=08) in that staging job ...

Consider the approach to address this issue as documented in the Z-Clues (login required).

**Source URL (retrieved on *2026-11-04 04:07*):**
http://dr.chgman.com/z-factory/z-issues/scmbeans/s007



Copyright © 2000-2025 AbitMORE ® BV. All rights reserved