

David Spiegel

Large Systems Consultant

**Introduction**

I have been considering writing articles on Real World Systems Programming. The purpose of these articles is to discuss practical Systems Programming strategies and tactics which will produce efficient, repeatable results. The ideas presented here are also meant to reduce errors and prevent failed changes. Most of what I discuss here has been learned on the job by trying different things over a period of 40 years.

One more thing ... My methodology is not the only way to do this properly, it is just one way to do it. The important thing is consistency. The more consistent you are, the easier systems changes are to produce. This mindset is similar to code reuse. I call it instruction reuse.

**General Approach**

Before I mention any actual details, I think that it is important to set the stage so that everyone is "on the same page".

**The Software Change: Start to Finish**

Starting approximately 2 weeks before the scheduled software change, anyone wanting to make system change must fill out (virtual) paperwork and attend "Change Board" meeting to present their proposed change. Needless to say you need to have "all your ducks in a row". If you're not adequately prepared, the Change Board folks will tear apart your change and it will be rejected. (This is somewhat reminiscent of the proverbial sharks smelling blood which causes a feeding frenzy.)

During the week prior to the change, you should review and tweak your step-by-step change documentation and have it peer reviewed to ensure that nothing obvious is missing.

The Change Window (during which the changes are actually done) is usually 4 hours (starting Sunday at 02:00) and you MUST be complete before 06:00. There will also be other groups of people making changes simultaneously. This means that the Project Manager will be messaging you periodically with updates and questions as to your progress. Since the adrenaline is already flowing (along with caffeine), one of the goals is to limit the stress.

Here are my "rules" for systems changes:

* All steps must be documented in a consistent way. I store this documentation in a $$README member of the PDS containing the implementation jobs. I use this name because "$" is lexically less than the alphabetic characters. This means that when the PDS is opened, the $$README is at the top.
* (Almost) All activity must be done via Batch Jobs. This is necessary for repeatability and tracking purposes. If there is no Batch method to do a given step, that is an exception, but, it must still be documented.
* Member names (I've struggled with this one for a long time) should be of the form: XXXYYYnn where XXX is the software product, YYY is the function used and nn are ordinal numbers. For example, to DFSMSdss COPY a dataset, the member name will be DSSCOP01, DSSCOP02, etc. Copying PDS/PDSE members via IEBCOPY will be stored in IEBCOP01, IEBCOP02 etc.
* Try to implement via RENAMEs (Member or Dataset) rather than copying/replacing. This mitigates the risk involved. When copying, there can be a variety of potential errors including: DASD Volume/SMS Storage Group out of space, Dataset out of space and insufficient number of directory blocks.