The Seven Deadly Sins of ILE

April 2004 | by Jon Paris, Susan Gantner,

Over the last few years, weve taught and consulted with many shops on topics relating to the Integrated Language Environment (ILE). During this time, weve seen many practices that work well, some that dont work so well, and a few that are outright lethal. Weve concluded that seven ILE practices are either best avoided or should be severely restricted. These "seven deadly sins" are the subject of this article. Well touch on each of them here, although many require a full article in their own right to do them justice. If reader interest warrants, well detail specific "sins" in future issues of iSeries EXTRA.

First deadly sin: Allowing "real" ILE programs to run in the default activation group.
What do we mean by "real" ILE programs? These are programs that are created either via the CRTxxxMOD and CRTPGM commands or by specifying DFTACTGRP(*NO) to one of the CRTBNDxxx commands (where xxx is RPG or CL). Note that for the purposes of this discussion, programs created with the CRTBNDxxx default of DFTACTGRP(*YES) arent considered to be ILE programs; theyre instead original program model (OPM)-compatible programs, and therefore arent bound by any of our ILE rules.

The default activation group (sometimes called DAG) is designed specifically for OPM programs and is the only activation group where these programs can run. You cannot directly request that your ILE programs run in the DAG, but if you specify *CALLER for the activation group parameter and subsequently call the program from an OPM program, your ILE program will run in the DAG. This is contrary to ILEs design (IBMs original intent was that ILE programs not be permitted to run in the DAG at all) and will eventually cause problems of some sort.

Second deadly sin: Using service programs in the DAG.
Many shops commit the first deadly sin and run their ILE programs in the DAG. If you happen to work in one of these shops, you can avoid some of the most serious problems by running your service programs in their own named activation group(s). Service programs are more prone to problems when running in the DAG, particularly if the Reclaim Resource (RCLRSC) command is run while the service program is active with files open. While RCLRSC is often used specifically to close open files, the command doesnt notify the service program logic that the files are closed. This leaves the application in a Catch 22 situation: The files are closed, but the procedures in the service program believe theyre still open.

Third deadly sin: Using *NEW by default rather than by design.
The *NEW option creates a brand new activation group every time the program is called. If specified for a large number of programs, this option can dramatically slow your application. So why would anyone ever use such a value as standard? Its pretty simple: Prior to V5R3, *NEW was the IBM default on the CRTPGM command. Because of this, many developers use it by accident. While *NEW might be an appropriate choice in some cases, they are few and far between. For this reason, a good approach might be to change the default value for activation group on CRTPGM.

In V5R3, the shipped default value for the Activation Group parameter on the CRTPGM command changed from *NEW to *ENTMOD (Entry Module). This means that the system looks at the language of the module used as the program entry procedure to determine the appropriate activation group value. If the entry module language is RPG, COBOL or CL, then the name "QILE" is used for the activation group. If the entry module is C, then the previous default value of *NEW is used. This is a great improvement and will hopefully reduce the number of accidental uses of the *NEW value. Its also more consistent because the default value on the CRTBNDRPG, CRTBNDCBL and CRTBNDCL commands has always been QILE. (Note: If you create a program that uses the teraspace storage model-which would be rare for RPG or COBOL programs-the default activation group name would be QILETS instead of QILE.)

Fourth deadly sin: Changing CRTPGMs default for activation group to *CALLER.
If you do, the first deadly sin will inevitably be broken.

So if *NEW and *CALLER are both bad options for the default for the activation group parameter, whats a better default? Pick a name-QILE is as good as any other. Theres nothing special about QILE, but its the default name supplied on the CRTBNDxxx commands, so if you change your default on CRTPGM to match, it makes for more consistent behaviour.

Fifth deadly sin: Failing to use RCLACTGRP only when necessary and using RCLACTGRP(*ELIGIBLE) in production code at all.
RCLACTGRP(*ELIGIBLE) is very useful to the programmer-but its lethal in production. Heres a hint as to how concerned the IBM developers that you should use this option with great care: Issue the RCLACTGRP command and prompt it (F4). Most special values for parameters appear on the prompt screens. Do you see *ELIGIBLE? Dont strain your eyes; it isnt there! Only by pressing F1 can you confirm that the option exists. This isnt an accident. IBM probably wouldve done away with the option had it immediately realized how much trouble it causes. As is, IBM has buried it as best it could.

The Reclaim Activation Group (RCLACTGRP) command can be used to close files and reclaim ILE programs storage in the users job (similar to what RCLRSC does for OPM programs). However, since recreating the activation group is costly, if the program is likely to be subsequently called, its best to try to avoid over using RCLACTGRP command.

Why is the *ELIGIBLE option useful to programmers? Because it reclaims all activation groups in the job that dont currently have an entry in the call stack. While this makes it very handy for a quick cleanup during program testing, the option should never be used in production programs. You can never be sure what activation groups may be in the users job at the time you issue the command. OS/400 sometimes starts activation groups and most ILE packaged application software runs in named activation groups. Therefore, you cant ever be sure what activation groups might be in the users job when you reclaim them. You could be reclaiming a lot more than you mean to, particularly if service programs are involved. By their very nature, the procedures in these service programs make only transitory appearances on the call stack. If the service program(s) are in a different activation group to the programs that call them, then *ELIGIBLE will destroy that activation group. This will subsequently result in other programs blowing up, as the pointers they have for the service program routines are rendered invalid.

Sixth deadly sin: Allowing scoping parameters to default.
This is probably the toughest "sin" for many shops to avoid. There are scoping parameters on all override commands-OPNDBF, OPNQRYF and STRCMTCTL. Thats a lot of commands to sift through to supply values, but if you want to live in peace in a mixed OPM/ILE environment, this is the best advice we can give.

The specific scoping parameters were talking about here are Override Scope (OVRSCOPE), Open Scope (OPNSCOPE) and Commit Scope (CMTSCOPE). These control which programs in the job "see" the override, shared open data path or commitment control transaction. Most important here is that we recommend you run your ILE programs in named activation groups. All your OPM programs must run in the DAG. If you allow the shipped command defaults to take effect, then any OVRxxxx commands issued by ILE programs will be "invisible" to your OPM programs. This is because the shipped defaults say that overrides done in ILE activation groups are visible only to programs in that group. On the other hand, overrides done in the DAG will behave according to our "traditional" call level rules, regardless of activation group. OPNSCOPE and CMTSCOPE have slightly different default parameter values, but the principals are the same.

Its a bit beyond the scope of this article to fully explain all the options here. However, if you want your shared opens, OPNQRYFs and commitment control commands to work as closely as possible in ILE applications as they did in your OPM applications, then you most likely want *CALLLVL for OVRSCOPE and *JOB for both OPNSCOPE and CMTSCOPE. One last catch here is that your *JOB level overrides should be explicitly deleted when no longer needed; they wont go away by themselves.

Seventh deadly sin: Using the RCLRSC command.
We described the issues with the RCLRSC command related to service programs. Even with ILE program objects, however, RCLRSC doesnt behave the way it did for OPM programs. Because figuring out exactly what the impact is going to be in each program is a bit difficult, RCLRSC is best avoided when other options available to close files and logically end programs.

'Death' Comes Slowly
These are our seven deadly sins of ILE. Of course many wont result in instant death, but they are "deadly" in a slow and insidious fashion, lying in wait to inject their poisonous fangs you when you least expect it.

One last thought: As a general rule, safely mixing OPM and ILE programs requires far more extensive knowledge of ILE than is needed to simply switch completely to ILE.

If youd like to know more about any of our recommendations-or if you disagree with any of them-let us know.