

The case for using CSO in place of .gnt modules

On UNIX platforms, before the introduction of Server Express, customers would typically develop and debug their code using .int (intermediate code), because it could be Animated and it was portable. Customers would then compile their applications to .gnt (generated native code) or executable format when their code was ready to run in production. The advantage of executable and .gnt was that these formats run substantially faster than .int.

Micro Focus COBOL has been available on UNIX for more than a decade, and a large customer base has grown accustomed to the approach of debugging with .int and running with .gnt or executable modules. Mission-critical applications have been successfully running .gnt for years, and customers have grown to trust this arrangement.

Server Express offers a new object module format called the Callable Shared Object (CSO). This format can be used as an alternative to .gnt. CSOs are created with the "-z" flag of the "cob" command, and they take the file name extension ".so". CSOs are similar to .gnts in that they are dynamically loadable, are native code in format, and run just as fast (or faster). They can do everything that .gnts can do, and they offer other advantages. They run as "shared text", and they allow several subprograms to be combined and loaded at once, possibly including subprograms from other languages such as C or C++.

Shared Text

Generated code files (.gnt) cannot be shared between processes. So, if two UNIX processes call a subprogram compiled to .gnt, a separate copy of that subprogram is loaded into each process's virtual memory space. The Micro Focus dynamic loader reads .gnt files into the .data section, and each UNIX process keeps its own private copy of the .data section.

CSOs are in an operating system format and can be shared between processes. If two UNIX processes call a subprogram created as a callable shared object at the same time, that program is loaded only once. The operating system shares the procedural code and creates separate data areas for each process. Depending on the architecture of the application and the number of simultaneous users, this can be a substantial advantage.

Straightforward Conversion

Although Server Express and its Callable Shared Objects have been available for four years, SupportLine still notices many customers using the .gnt format in production. The .gnts have performed well, and customers imagine that a conversion to CSOs will involve extra effort in engineering and QA. They think, "if it's not broken, don't fix it".

But there is a straightforward way to realize at least the "shared text" advantage of CSOs. In the customer's Makefiles or build scripts, where the "-u" flag of the "cob" command appears, this can be replaced with the "-z" flag, then a ".so" file will be created in place of each .gnt. Replacing .gnt files with CSOs on a one-to-one basis is a conceptually simple change (even within complex applications). It is not likely to introduce any problems and can be tested in one QA cycle. Customers should be encouraged to try this approach.

It is important to realize that, at execution time, the operating system will look for CSOs on the "library path" environment variable, that is, LIBPATH on AIX, SHLIB_PATH on HP, or LD_LIBRARY_PATH on other UNIX versions. The library path must include the location of the CSOs, even if this location happens to be the current working directory.

Combining subprograms for performance, and including C and C++

When .gnt modules are used, for each COBOL source file (.cbl) a separate .gnt file would be created. Then, the dynamic loader would have to find and load each .gnt individually at run time. However, if you choose to do so, several COBOL programs can be combined into a single CSO. This has the advantage that references between the programs can be resolved without the dynamic loader having to search for and load another module. For example:

The case for using CSO in place of .gnt modules

```
cob -z myprog.cbl subprog.cbl entry.cbl
```

compiles the three COBOL programs and links them together, creating myprog.so. The entry point of the CSO is myprog (the first COBOL program listed on the line).

The cob -z option accepts .int files and system object (.o) files as input, as well as C and C++ source or object code. For example:

```
cob -zo myapp.so mycobol.cbl myint.int myc.c myobj.o -e myentry
```

creates myapp.so using the COBOL programs mycobol.cbl and myint.int, the C source code myc.c, and an object file myobj.o, which could be COBOL, C, C++ or some other language. The main entry point is "myentry" which could be an entry point in any of the specified modules.

A CSO does not need to contain COBOL, so "cob -z" can be used to create CSOs containing only C or C++ or other objects. Prior to Server Express, C and C++ needed to be linked into a system executable or a new runtime system. This is no longer necessary, as they can now be linked into callable shared objects, and be brought in dynamically at run time. This allows much greater flexibility when creating mixed-language applications.

For more information about CSOs and their advantages over the more traditional .gnt format, I recommend these areas of the Server Express documentation:

User's Guide Chapter 3: Packaging Applications
Chapter 8: Callable Shared Objects
Chapter 9: Linking to System Executables
and Chapter 10: COBOL System Interface (Cob)

The Server Express documentation (along with documentation for other Micro Focus products) is available on the Documentation CD that comes with the product and is also accessible from the Micro Focus web page www.microfocus.com. Start by selecting the "Support" link at the top of the home page. Then select the "Documentation" link in the "Self-Service" menu.