messagefacility - Feature #19960

Proposal for use of self-location in MessageFacility for builtin plugins

05/15/2018 11:40 AM - Ben Morgan

Status:	Under Discussion	**Start date:**	05/15/2018
Priority:	Normal	**Due date:**	
Assignee:	Christopher Green	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour

Description

In relation to Issues #17900 and #18942, I've explored additional ways to allow plugins to be located at runtime without requiring environment configuration by clients. This is for the case of packages like MessageFacility (potentially art) that supply "builtin" plugins, e.g sqlite_mfPlugin. Ideally, these should not **require** the user of the package to configure the environment ((DY)LD_LIBRARY_PATH/MF_PLUGIN_PATH here) to use functionality that the package itself *theoretically* knows how to find.

The "theoretically" part is satisfied *if* the program or library can determine its absolute path on the filesystem at runtime. As the package knows relative paths between its components, fully runtime relocatable packages are possible if self location is (NB: this is orthogonal to RPATH). At least on Unices, the binreloc code developed way back when for the Autopackage project can do this. I've therefore implemented a proof of principle demo of this for MessageFacility on this branch/commit from develop:

https://github.com/drbenmorgan/fnal-messagefacility/tree/use-binreloc-for-plugins
https://github.com/drbenmorgan/fnal-messagefacility/commit/1613f49110bb22621988de86561f22b8c35ca6b7

binreloc comes as a small bit of C code, and is only an implementation detail of the libMF_MessageLogger library. The only change to existing MessageFacility code is to the private getPluginPath function. This simply runs the self location and uses the determined runtime path of libMF_MessageLogger library as a last resort location to look for plugins in deference to MF_PLUGIN_PATH/LD_LIBRARY_PATH. If this path is empty (self location failed), behaviour is identical to before.

This is **very** basic proof of principle (see the comments on error handling, messy environment checks etc), and has only been tested on macOS/Linux (Debug, e15) on local filesystems so far. It is back compatible with existing use of environment variables, which are still used as the first places looked for plugins. Note that to test it on Linux, I ran with an environment exactly as set by setup_for_development, then with an updated LD_LIBRARY_PATH stripped of any messagefacility build path(s). The macOS builds were on non-UPS, full SIP, so unset DYLD_LIBRARY_PATH. In both cases, the tests all pass.

I think it's worth further investigation, but let me know any thoughts/no-gos...

History

#1 - 05/21/2018 10:29 AM - Kyle Knoepfel

- Status changed from New to Under Discussion

- Assignee set to Christopher Green