

How_to_build_cross-referencing_DLLs

Definition: *cross-referencing DLLs* are dynamic libraries that depend on each other in terms of using each other's services.

Problem: building such components might be problematic as their build cycle relies upon the fact that the "other" component has already been built beforehand. Note that the problem arises if the components are built *separately*. But in that case it generates a fatal build error.

Solution: write a common bld.inf file in which both components' MMP file are present. The new subsystem, i.e. the one containing at least our two components, is to be treated as a single package, which ensures that the code will compile and build properly.

Explanation: as it pointed out in abld.bat, `abld build` performs the following actions one after another

- export
- makefile
- library
- resource
- target
- final.

In our case, it means that instead of building each component separately one after the other, they are to be built together: first step is to export each component's header (and other required) files, third is to generate LIB files and only the fifth is to make the executable from the generated code. This build sequence ensures that the code compiles (i.e. all required exported header files are present) and can be linked against the imported libraries.

Caution: this article is **NOT** to encourage you to write cross-referencing DLLs as it's usually not an elegant solution. Nevertheless, it may happen in some rare cases (e.g. being constrained in a legacy framework) that you can't avoid using them in your implementation.