"Build Configurations" & "Tool-chains"

Position Paper for Eclipse Languages Workshop

Leo Treggiari

October 12, 2005

Eclipse defines the concept of a Project. I propose that Eclipse also needs to define the concepts of "build configuration" and "tool-chain", in order to support compiled languages and multi-language development.

Tool-chain: An ordered set of tools used to transform the project resources into the final output (build artifacts) of the project.

Build Configuration: A combination of the following that is used to build a user's project in a particular manner:

- a set of source resources (i.e., a project, possibly modified by excluding certain resources)
- a tool-chain (compiler{s), assembler(s), linker/archiver, etc.)
- a set of tool-chain option settings (e.g., compiler options)
- a set of environment variable settings (e.g., various paths)

JDT has not required these concepts, because:

- JDT supports a single Java compiler
- Java doesn't have a preprocessor, or any conditional compilation capability, and compiler options are very limited (and not often changed?)

CDT supports many different C/C++ compilers and therefore needs information about the compiler and the rest of the tool-chain used to build projects. C/C++ projects use the C preprocessor and compiler options to define different ways to build a project (typically debug and release configurations, at a minimum). Each CDT build configuration uses a particular tool-chain to build a project. Different build configurations in a project can use different tool-chains.

Other compiled languages will have the same requirements, for example, Fortran. Will JDT ever support multiple compilers, e.g. compilers that generate "native" code rather than byte code?

These concepts should be common across all languages that need them in order to:

- Make dealing with multiple languages easier for the user
- Provide a mechanism when defining referenced projects to select a particular build configuration