The New C Standard (Excerpted material)

An Economic and Cultural Commentary

Derek M. Jones
derek@knosof.co.uk

5. Environment

environment execution

An implementation translates C source files and executes C programs in two data-processing-system environments, which will be called the *translation environment* and the *execution environment* in this International Standard. 104

Commentary

conforming hosted implementation
freestanding environment startup

For a hosted implementation the two environments are often the same. In some cases application developers do cross-translate from one hosted environment to another hosted environment. In a freestanding environment, the two environments are very unlikely to be the same.

A commonly used term for the execution environment is *runtime system*. In some cases this terminology refers to a more restricted set of functionality than a complete execution environment.

diagnostic shall produce

The requirement on when a diagnostic message must be produced prevents a program from being translated from the source code, on the fly, as statements to execute are encountered.

Rationale

Because C has seen widespread use as a cross-compiled cross-compilation language, a clear distinction must be made between translation and execution environments. The C89 preprocessor, for instance, is permitted to evaluate the expression in a **#if** directive using the long integer or unsigned long integer arithmetic native to the translation environment: these integers must comprise at least 32 bits, but need not match the number of bits in the execution environment. In C99, this arithmetic must be done in `intmax_t` or `uintmax_t`, which must comprise at least 64 bits and must match the execution environment. Other translation time arithmetic, however, such as type casting and floating point arithmetic, must more closely model the execution environment regardless of translation environment.

C++

The C++ Standard says nothing about the environment in which C++ programs are translated.

Other Languages

JIT

Java defines an execution environment. It says nothing about the translation environment. But its philosophy, *write once run anywhere* means that there should not be any implementation-defined characteristics to worry about. There are implementations that perform Just-in-time (JIT) translation on an as needed basis during execution (implementations differ in the granularity of source that is JIT translated).

Coding Guidelines

Coding guidelines often relate to the translation environment; that is, what appears in the visible source code. In some cases the behavior of a program may vary because of characteristics that only become known when a program is executed. The coding guidelines in this book are aimed at both environments. It is management's responsibility to select the ones (or remove the ones) appropriate to their development environment.

Their characteristics define and constrain the results of executing conforming C programs constructed according to the syntactic and semantic rules for conforming implementations. 105

Commentary

The translation environment need not have any effect on the translated program, subject to sufficient memory being available to perform a translation. It is not even necessary that the translation environment be a superset of the execution environment. For instance, a translator targeting a 64-bit execution environment, but running in a 32-bit translation environment, could support its own 64-bit arithmetic package (for constant folding).

translation phase 4

In theory each stage of translation could be carried out in a separate translation environment. In some development environments, the code is distributed in preprocessed (i.e., after translation phase 4) form. Header files will have been included and any conditional compilation directives executed.

In those cases where a translator performs operations defined to occur during program execution, it must follow the execution time behavior. For instance, a translator may be able to evaluate parts of an expression, that are not defined to be a constant expression. In this case any undefined behavior associated with a signed arithmetic overflow could be defined to be the diagnostic generated by the translator.

C++

The C++ Standard makes no such observation.

Coding Guidelines

The characteristics of the execution environment are usually thought of as being part of the requirements of the application (i.e., that the application is capable of execution in this environment). The characteristics of the translation environment are of interest to these coding guidelines if they may affect the behavior of a translator.

---

106 **Forward references:** In this clause, only a few of many possible forward references have been noted.

Commentary

This statement could be said to be true for all of the Forward references appearing in the C Standard.

References