The New C Standard (Excerpted material)

An Economic and Cultural Commentary

Derek M. Jones
derek@knosof.co.uk

5.2.4 Environmental limits

environmental limits

Both the translation and execution environments constrain the implementation of language translators and libraries. 273

Commentary

In the past the available storage capacity of the translation environment has been an implementation design consideration for translator vendors. Translating programs containing large numbers of some constructs sometimes exceeded the available host capacity. This is rarely the case today.

In some environments, particularly freestanding ones, there can be severe constraints on the execution environment.

C++

There is an informative annex which states:

Annex Bp1 *Because computers are finite, C++ implementations are inevitably limited in the size of the programs they can successfully process.*

Common Implementations

While it might be theoretically possible to create a translator that is not affected by its own environment, the cost is likely to outweigh the benefits.

Coding Guidelines

Some limits, imposed by the translation environment, are best dealt with as they are encountered. The cost of structuring a program to deal with all lowest common denominators is rarely recouped in future savings (in reduced porting costs). Programs are often targeted at a particular class of environments (e.g., workstations or hand-held devices). Execution time constraints can have a large impact and may affect the choice of algorithms as well as how the source is structured. Both of these issues are dealt with, by these coding guidelines, as they are encountered in the C Standard wording.

The following summarizes the language-related environmental limits on a conforming implementation; 274

Commentary

The intent is that these are base limits and commercial pressure will encourage vendors to create implementations that improve on them. By specifying such limits the Committee is providing a guide as to what can be expected, by a developer, of an implementation.

C++

There is an informative annex which states:

Annex Bp2 *The bracketed number following each quantity is recommended as the minimum for that quantity. However, these quantities are only guidelines and do not determine conformance.*

Other Languages

Most language standards are silent on the subject of environmental limits and provide no guide on the number of constructs that a translator might be expected to handle. The Modula-2 Standard specifies minimum translator limits for many of the language constructs covered by the C Standard (Pronk[1] tests the minimum values supported by a number of translators).

Common Implementations

Most translators allocate space for the symbol table and other information, dynamically as the source code is processed. This choice of implementation technique does not remove the limit on the total amount of

memory available to a translator, but it does provide flexibility. There are a few cases where limits may be imposed because an implementation has chosen to use fixed-size data structures.

One limit not mentioned in the standard is the maximum number of characters in a macro, during expansion. Several implementations have limits in this area, sometimes as low as 256. The limit for macro definitions is covered by the logical line length.

macro replacement

limit characters on line

Coding Guidelines

Exceeding any defined minimum limits is a calculated risk. Some of the limits may be hard to design around; for instance, the number of identifiers with external linkage. What is the cost, both in developer effort and loss of design integrity, of adapting a program to fit within these limits? What is the likelihood of encountering a translator that cannot process a source file that exceeds some limit? Is it worth paying the cost to be certain of having source that is translatable by such translators? While the environments where translators are resource-limited are becoming rare, many translators continue to contain some of their own, internal, fixed limits.

A translator may have other limits that are not described in the C Standard. These will have to be dealt with, by developers, as they are encountered.

---

275 the library-related limits are discussed in clause 7.

C++

Clause 18.2 contains an *Implementation Limits:*.

References

1. C. Pronk. Stress testing of compilers for Modula-2. *Software–Practice and Experience*, 22(10):885–897, 1992.