Proposal for C2y
WG14 N3863

Title:	Redefining implementation-defined
Author, affiliation:	Aaron Ballman, Intel
Date:	2026-03-23
Proposal category:	Normatively editorial
Target audience:	WG14, UB study group, implementers

Abstract: C technically requires implementation-defined behavior to be chosen from an explicit list of options given by the C standard, but most uses of the term of art in the C standard do not follow this requirement. This proposal rewords the definition to better match reality.

Prior art: C as it is specified and implemented in practice, C++ per specification

Redefining implementation-defined

Reply-to: Aaron Ballman (aaron@aaronballman.com)
Document No: N3863
Date: 2026-03-23

Summary of Changes

N3863

- Initial proposal

Introduction and Motivation

C has several defined terms of art for describing the behavior of code, one of which allows for the implementation to define the behavior. However, the use of this term of art does not match its definition in most of the places it is used across the standard. This has recently caused some concern within the committee as we've been removing undefined behavior and replacing them with constraint violations or implementation-defined behavior.

N3783 3.5.1p1:
implementation-defined behavior
unspecified behavior where each implementation documents how the choice is made

N3783 3.5.4p1:
unspecified behavior
behavior, that results from the use of an unspecified value, or other behavior upon which this document provides two or more possibilities and imposes no further requirements on which is chosen in any instance

Because implementation-defined behavior relies on the definition of unspecified behavior, it means that implementation-defined behavior requires two or more possibilities to be defined in the standard. However, a brief look at its uses in the standard clearly shows that there isn't always a list of potential behaviors to pick from.

N3783 5.2.1.2p1.1: Physical source file multibyte characters are mapped, in an implementation-defined manner, to the source character set (introducing new-line characters for end-of-line indicators) if necessary.

N3783 5.2.1.3p1: A conforming implementation shall produce at least one diagnostic message (identified in an implementation-defined manner) if a preprocessing translation unit or translation unit contains a violation of any syntax rule or constraint, even if the behavior is also explicitly

specified as undefined or implementation-defined. Diagnostic messages are not required to be produced in other circumstances.

N3783 5.2.2.2p1-2: In a freestanding environment (in which C program execution can take place without any benefit of an operating system), the name and type of the function called at program startup are implementation-defined. / The effect of program termination in a freestanding environment is implementation-defined.

That is just a very brief sample of uses where there is not a given list of behaviors to pick from. In general, the committee seems to use implementation-defined more akin to the C++ definition of the term where the behavior is left to the implementation to define. However, that definition relies on other C++-specific terms of art like "well-formed" which C does not have, so an equivalent definition is proposed for C.

Note: "implementation-defined value" also relies on the definition of unspecified value, which says "on which value is chosen in any instance" which suggests that implementation-defined value is also a deficient definition. However, this is left for later repair because uses of the term of art in the standard appear to be correct and it's a larger research project to see whether implementations actually make use of the "in any instance" allowance for implementation-defined values in practice. Further, this paper does not attempt to answer broader questions about what kinds of implementation-defined behaviors are permissible in practice (can the behavior become undefined, can there be additional constraints, etc).

Impacts

While this is a normative wording proposal, it is expected to behave like an editorial change in that no implementation is expected to change behavior. Every instance of "implementation-defined" is expected to have the same interpretation; the new wording is intended to allow for an implementation to document behavior for which the standard has no list of choices.

Proposed Wording

The wording proposed is a diff from the N3783 working draft of ISO/IEC 9899. Green text is new text, while ~~red~~ text is deleted text.

Modify 3.5.1p1:
implementation-defined behavior
~~unspecified behavior where each implementation documents how the choice is made~~ behavior, for a nonportable program construct used with non-erroneous data, that depends on the implementation and that each implementation documents

Acknowledgements

We would like to recognize the following people for their help in this work: David Svoboda, Rajan Bhakta, and Joseph Myers.