Doc No: X3J16/94-0087 WG21/N0474
Date: 30 May 94
Project: Programming Language C++
Reply to: Michael J. Vilot
ObjectCraft, Inc.
7 Colby Ct., suite #4–321
Bedford NH USA 03110–6427
`mjv@objects.mv.com`

Clause 17 Introduction (Editorial)

This proposal is for an editorial change in the description of the Standard C++ library. The contents are much the same as the Working Paper, but the sections have been re-arranged and important information has been added.

This proposal addresses many of the small editorial issues (“boxes”) that were raised during the review of the San Diego rewrite of Clause 17. Strictly speaking, editorial proposals do not require formal proposals, motions, and votes. However, the changes proposed are widespread enough to warrant an explicit proposal and discussion.

2. Proposal

Re-arrange the material in §17.1, “Introduction” (_lib.introduction_), as shown in the attached document labeled 17, Library.

Move some of the material on differences from the C library to Annex C, “Compatibility” (_diff.library_), as shown in the attached document labeled 18, Compatibility.

3. Discussion

The San Diego rewrite of Clause 17 imported far more of the Stnadard C library than the Library WG ever intended with proposal 92–0024/N0101, “ISO C Library Inclusion,” as voted on and approved at the London meeting in March, 1992. Basically, the rewrite defines its terms using the vocabulary of the C standard, defines the C++ library as an extension of the C library, specifies the C++ library in terms of C library facilities, and uses the same functional style as the C standard to document the classes in the C++ library.

On a strictly technical basis, the rewrite might be considered an adequate specification for the Standard. However, when taken together, the results are inappropriate for C++ and inadequate as a Standard reference: necessary information is hard to find, important constraints are documented obscurely, important features of C++ are ignored or underutilized, and some aspects of the library are substantially overspecified.

The advantages of the San Diego rewrite include an exhaustive treatment of the library’s contents, a consistent organization, and wording in terms of accepted “standardese.”

This proposal attempts to keep the advantages of the San Diego rewrite, while eliminating most of its disadvantages. The new version balances several competing demands:

- a comprehensive, yet understandable, framework for organizing the library
- reuse of C library terms and concepts, *as appropriate to C++*
- exhaustive summary of the library’s contents, organized alphabetically
- a useful reference to the library’s contents, organized by concept

Perhaps the most important advantage of the proposed new version is that it provides an integrated description of the entire C++ library, including the features incorporated from the C standard.

The new version relies on machine-generated tables (and the Index) for the alphabetical summaries (see pages 17–9 through 17–14). Much of the reorganization, therefore, focuses on a topical framework. This framework is organized into 6 main sections:

Subclause	Title
17.1.1	Scope
17.1.2	Normative references
17.1.3	Definitions
17.1.4	Method of description
17.1.5	Library-wide requirements
17.2–17.12	Detailed specifications

Subclauses 17.1.1–17.1.3 are straightforward, and their organization parallels that of Clause 1, "General" (_intro_). Subclauses 17.1.4 and 17.1.5 contain the main bulk of the reorganized material, with 17.1.4 being primarily informational and 17.1.5 being primarily normative. Subclauses 17.2–17.12 are included to illustrate the consequences of this revised organization.

Subclause 17.1.4, "Method of description" (_lib.description_), provides a central explanation of the editorial style used throughout the rest of Clause 17. It allows us to introduct concepts such as "Implementation types" (§17.1.4.3, _lib.implementation.types_), that are used in the specification of the Standard C++ library — but never appear in the library's technical contents. This approach is very important for the STL components, whose specifications rely heavily on constraints documented though "Iterator types" (which may be either classes or built-in types) and "Predicate types" (which may be either classes or pointers to functions).

Subclause 17.1.5, "Library-wide requirements" (_lib.requirements_), provides a central reference for the library's technical overview. It is intended to be useful for both library users and implementers, and has four subclauses:

Subclause	Title
17.1.5.1	Contents & organization
17.1.5.2	Using the library
17.1.5.3	Constraints on programs
17.1.5.4	Conforming implementations

Subclause 17.1.5.1 provides the overview and alphabetical summary of the library as a whole. A key feature of this subclause is that it provides an integrated view of the C++ library, including the features incorporated from the ISO C standard and its Amendment 1. Subclause C.4, "Standard C library" (_diff.library_), provides a similarly comprehensive treatment of the differences between this library and the one specified by the C standard.

Subclause 17.1.5.2 describes how a well-formed C++ program makes use of components from the library. It covers both the use of headers (phase 4 in §2.1, "Phases of translation," _lex.phases_) and linkage (phase 8).

Subclause 17.1.5.3 describes the constraints a well-formed C++ program must meet to use the library's components in a well-defined way. This includes not only namespace constraints, but also run-time behavior. A key feature of this subclause is a more effective treatment of "replacement functions" and "handler functions," which have been a challenging topic of appropriate "standardese" since the Toronto meeting.

Subclause 17.1.5.4 provides a centralized description of the library implementer's latitude in conforming to the Standard. While some of the elements of this section still need to be refined, it provides a much-needed summary for library implementers.