Document Number: X3J16/92-0042, WG21/N0119
Date: 4/3/92
Project: Programming Language C++
Ref Doc: X3J16/91-0134, WG21/N0067
Reply To: Michael J. Vilot
ObjectWare, Inc.
Nashua NH 03062 USA
mjv@objects.mv.com

Report from the Library working group
Work done November 1991 - March 1992

This report summarizes the progress made by the Library working group between the Dallas and London meetings, and the work done at the London meeting. The first section of the report provides details of the progress report presented to X3J16/WG21 on Monday, 16 March. The second section provides details of the progress report presented Thursday, 19 March. The report concludes with a summary of work planned for the July meeting in Toronto.

Progress Report — November 1991 - March 1992

X3J16/WG21 agenda item #2.3 presented the Library working group's progress since the November meeting in Dallas. The presentation reviewed the group's overall work plan, a summary of the documents and e-mail discussions between the meetings, and a summary of the currently open issues.

Overview

The purpose of the overview was to acquaint new members with the activities of the Library working group. The presentation followed the contents of the eventual "Standard Library" chapter of the X3J16/WG21 working paper, even though no such chapter presently exists.

What the group is working on

While the language portion of C++ has had over 10 years' evolution and refinement, the standard library has had much less time to develop. The availability of templates and exceptions, only recently added to the language, has a significant impact on the design of libraries for C++.

The main activity of the Library working group is to produce documents to serve as proposals to X3J16/WG21.

Current Focus

The current focus of the Library group remains as it has since the July, 1990 meeting in Seattle. The November, 1991 meeting in Dallas added consideration of simple container classes. The group is concentrating effort on the first five sections of Chapter 18, the chapter in the working paper that will specify the standard library.

During the Dallas meeting, two discussions indicated potential sources of additional C++ library suggestions. Steve Carter mentioned "language independent data types" during a presentation of international concerns.. Randall Swan mentioned numeric types during a presentation of NCEG (X3J11.1) liaison status. Either or both of these might have to be implemented in a standard C++ library.

18.1 Language Support

This section will describe the functions, classes, and templates required to support certain behavior described in the language portion of the standard.

For example, this section will describe the default definitions of the `operator new()` and `operator delete()` functions, `terminate()` and `unexpected()`, and so on. It will also define standard exceptions, such as `xalloc`.

18.2 Input/Output

This section will describe the stream I/O facilities.

As discussed at the March, 1990 meeting in Somerset, and ratified at the July, 1990 meeting in Seattle, these facilities will be a simplified and implementation-independent specification of the AT&T `iostreams` library.

The key simplifications include removing the requirement for using multiple inheritance, and omitting certain specialized streams (such as `stdiostream`).

An important design goal for the streams library is to provide C++ applications with a type-safe and extensible I/O facility. An additional goal is support for national language set character handling.

18.3 ISO C Library

This section will describe the support for the functions defined in the ISO C standard library. Incorporating these facilities into the C++ standard library will support the stated X3J16 goal of preserving maximum compatibility between C and C++.

Some aspects of the C library functions present challenges to incorporating them into C++. For example, the `exit()` function must be augmented to consider the presence of global objects of classes with destructors. The functions `setjmp()` and `longjmp()` interact with the implementation of C++'s exceptions. Functions such as `strchr()` open holes in the C++ type system. Incorporating these functions into the C++ standard library requires some modification of their semantics as C functions.

18.4 String

One problem encountered in existing C++ practice is that conflicts arise when combining libraries provided by different vendors. One point of conflict is the definition of roughly equivalent string classes that differ in detail. These differences are often enough to create both library combination and application portability difficulties.

The standard C++ string facility will define minimal, but sufficiently complete, strings that portable C++ programs can depend upon.

Key design goals for the standard string facility include type-safe versions of (at least) the character handling facilities of the C library, and better-integrated support for national language set character handling.

18.5 Containers

Reviewing available libraries revealed a small set of common classes. Strings were most frequent, and are already in the C++ library (18.4). Array and bit set classes were next most frequent, and appeared "easy" to specify.

The design approach would emphasize Concrete Data Types using templates. Proposals for container classes will add a new section to the library portion of the standard. For now, the subsection numbering will reflect each container in alphabetical order. This numbering would change if new containers are added.

Open issues from last time

The list of open issues did not change significantly from the list produced at the November, 1991 meeting in Dallas. Members who had attended the last meeting volunteered to write proposals or analysis reports summarizing the issue and recommending resolutions.

General

Templates and exceptions are relatively recent additions to the C++ language, yet their availability has a profound impact on the design of class libraries for C++ Currently, there are few available C++ libraries that use these language features. With little existing practice upon which to standardize, the library working group could be justifiably accused of designing by committee. The library's Rationale will explain the group's design decisions..

The group has discussed the header "file" naming convention used in the standard library. Existing C++ implementations use a variety of header file naming conventions (`.h`, `.hxx`, `.hpp`, `.H`, to name a few). The lack of consistency creates definite problems of portability and library usage for C++ programmers. Settling upon a consistent convention in the standard would encourage C++ implementations to adopt a consistent scheme.

A new aspect of the discussion is the opportunity to differentiate C++ headers from C headers. For example, `string.h` could refer to the existing C `str*` functions, while a `string` header could refer to the C++ string classes. However, changing the name of `iostream.h` to `iostream` would impact existing C++ implementations of iostreams.

The group has also discussed namespace management. However, until the language provides a better mechanism for namespace management, the standard library will have to adopt some techniques. There is also a difference of opinion regarding the use of typedefs that preempt names in the global namespace.

18.1 Language Support

Jerry Schwarz wrote a proposal for standard exceptions and presented it at the November, 1991 meeting in Dallas. As a result of that discussion, the group decided to include definitions of standard exceptions `xmsg` and `xalloc` in the language support section of the library.

The library will provide a default *new-handler* that will throw an `xalloc` exception. This will change the description of the default behavior of *new-expressions*.

Using `set_new_handler(0)` to unset the *new-handler* reverts to present behavior (i.e. `new` returns null)

The group also discussed the `xassert` exception and redefining the `assert()` macro in `<assert.h>`. There was no clear consensus for the change, although there was a consensus that the C++ library facility could be implemented without relying on macros. The semantics of the `assert()` macro will be a C library (18.3) issue, while discussion continues on an appropriate C++ library facility.

18.2 Input/Output

At the Dallas meeting, the group raised and discussed several issues related to the proposal 91-0117/N0050.

1) national language set character data streams
2) file open modes, newline translation, etc.
3) wide character support
4) interaction with other standards (e.g. ASN.1)
5) names of headers (existing C++)
6) streampos, streamoff "types"
7) exceptions thrown by streambuf
8) mode flag "types" (enums and or-ing)
9) name space
10) I/O support for strings, wstrings
11) stdio/streams interaction

Most of the issues resulted in some consensus, and changes will be reflected in the revised proposal. Some items require more discussion.

18.3 ISO C Library

In previous analysis reports (X3J16/90–0105 and 91–0032), Steve Clamage presented the issues involved in incorporating the C library facilities into the C++ standard library. Most of the issues had recommendations that were discussed in the full X3J16 committee. A few items required further work, and a specific proposal needed to be prepared for submission to the committee.

The Library working group reached a consensus to incorporate the relevant portions of the C library by reference, naming specific paragraphs in a specific version of the ISO C standard. The C++ standard would then specify changes in the semantics where necessary. A Rationale document would explain the reasons for such differences.

Details of interactions between elements of the C library facilities and other parts of the C++ library need to be clarified:

- streams & stdio
- new/delete & malloc/free/realloc
- exceptions & signals

During the discussion, the group identified a separate, but related issue: implementation of the library in a mixed C/C++ environment. Most aspects of this issue depend on the description of such an environment by the Environment working group.

18.4 String

During the previous meeting, the Library working group decided to rework the proposal for string classes. This meant abandoning the approach of 91-0078/N0019, and working from existing implementations. The group formed a small team (consisting of Pete Becker, Uwe Steinmueller, Chuck Allison, and several others with implementation experience) to investigate the issue in detail.

The string proposal team worked in parallel with the rest of the Library working group, and identified the issues they did not have time to resolve:

- an overview and justification/rationale for the proposal
- class specification
- national language set character details
- trial implementation(s)
- use of exception specifications and semantics of string member functions involving exceptions
- substrings and `char*` conversions, under the present C++ language rules regarding lifetimes of temporaries

18.5 Containers

At the November, 1991 meeting in Dallas, the X3J16 committee agreed to allow the Library working group to pursue specification of two new components. These components, identified after reviewing several existing library implementations, appeared to be common enough to be useful, and small enough to specify in relatively short order.

The BitSet component will describe an set abstraction that can be implemented efficiently. The Array component will describe self-describing arrays that avoid some of the problems associated with the array derived type defined in the language.

Progress since last meeting

Documents

18.1 Language Support

Mike Vilot provided a revised proposal for language support, including standard exceptions (91-0126/N0059)

The proposal describes a common base class `xmsg` that can be used to catch all exceptions thrown from the standard library.

The proposal defines standard exceptions `xalloc`, and `xassert` derived from `xmsg`.

18.2 Input/Output

Jerry Schwarz did not revise the input/output proposal (91-0117/N0050, 20 Sep 91)

18.3 ISO C Library

Steve Clamage provided a proposal to incorporate the ISO library (92-0024/N0101)

18.4 String

Because of e-mail difficulties, both Pete Becker and Uwe Steinmueller provided revised proposals (92-0006/N0084, 92-0025/N0102).

There was little change from the previous version.

18.5 Containers

Chuck Allison produced a proposal for BitSets (91-0142/N0075).

Uwe Steinmueller provided a proposal for dynamic arrays (92-0005/N0083).

Other

Members of the University of Waterloo in Canada submitted a proposal to incorporate concurrency support, in the form of their μC++ facility (91-0133/N0066). Dag Brück submitted an analysis of the proposal's extensive impact on the current language definition (91-0130/N0063).

Discussion on the `x3j16-lib` mail reflector

File naming

Some comments from John Wilkinson in the Environment group indicated that it would be useful to standardize source file naming conventions, since each vendor seems to pursue a different convention. The Library group decided that source file naming was indeed the vendor's concern, and beyond the purview of the standard.

Dynamic Arrays

Comments from Keith Gorlen and Tony Hansen indicated some minor problems with usability of the proposed classes. In particular, the constructors and assignment/append operations needed to provide more flexibility.

Assertions and exceptions

Philippe Gautron submitted a copy of his Usenix C++ paper as message x3j16-lib-189. This proposal would introduce standard template classes to report assertion failures by throwing exceptions. It would also be possible to redefine the `assert()` macro in terms o this mechanism.

Other meetings and discussions

Steve Teale forwarded comments he received on the article he published in the October, 1991 issue of *Dr Dobbs Journal*. The article presented his version of what he thought the standard string class should be, and raised several concerns among those who read the article.

Working Group Progress — 16 to 19 March

X3J16/WG21 agenda item #9 presented a summary of the Library working group's discussions during the WG sessions at the meeting. The presentation reviewed progress against, and changes to, the issues presented as part of agenda item #2.3. The summary followed the outline of the contents of the library portion of the X3J16/WG21 working document.

General

During the discussion of both the iostreams proposal and the C library proposal, the group identified some issues that apply to the entire library.

One issue is a policy towards header file dependencies. The C library forbids an implementation to create dependencies among headers. This allows conforming C programs to predict the impact of reserved identifiers on their name space.

The iostreams headers do depend on others, due to the need to have full class definitions available. The iostreams library also uses the constants EOF and NULL from the C library.

The group decided to leave the C library rules (of not including other headers) the same for the C library portion of the C++ library, but to allow other parts of the C++ library headers to include others. Of course, the library will specify which headers are so included.

With the present definition of the language, the main impact of this rule is to prevent a C++ library from implementing the string class with inline member functions that include the C string handling facilities.

18.1 Language Support

The group discussed the 91-0126/N0059 and uncovered some issues that it did not address adequately .

The characteristics of the `operator new()` and `operator delete()` functions are unique — they are the only functions in the library that may be replaced by programmer-supplied versions. Their special status imposes constraints on programmers who supply their own replacements, and these need to be specified.

This issue also highlighted the vagueness in the current description of the language, in that the time at which the replacements take effect is not defined. If an implementation of the standard library that uses operator new() and operator delete() internally for storage management, it is unclear if that usage is redefined to use the programmer-supplied replacements. Although that appears to be the intent, the current working paper is not definitive.

A related issue involves any programmer-supplied functions for a new-handler, terminate handler, or unexpected handler. The conditions under which they are invoked, and the semantics they are expected to support (and avoid) are not clearly specified.

18.2 Input/Output

The group raised and discussed several issues related to the proposal 91-0117/N0050.

The iostreams proposal presents certain types that are often implemented as integers (such as `streampos` and `streamoff`) as "`typedef ...`" and leaves the type unspecified. The group decided to endorse this approach, since it admits both existing implementations (using enumerations or long integers) and classes to meet the specification.

The language on these types will be vague in an attempt to meet Jerry Schwarz' goal of letting existing iostreams libraries meet the standard without requiring recompilation. Other types in the standard library will be specified more precisely, most likely as classes.

The group also discussed the impact of national language support on iostreams, specifically the multibyte sequence extension proposed by the Japanese delegation as a normative addendum to ISO C. The discussion centered on a paper presented by Tatsunori Hashimoto, describing two possible iostreams designs that would address the needs of multibyte sequencing.

The most difficult, and still unresolved, issue involves attempts to seek to arbitrary positions in a multibyte sequence stream. It appears necessary to place limits on the valid streampos values that can be reported and used for stream positioning, to avoid the difficulties involved with positioning in the middle of a multibyte sequence.

The group also discussed the interaction of the stdio and iostream facilities. Existing C++ practice seems to expect that calls to printf and ostream inserters can be arbitrarily intermixed, and the group decided to make this part of the iostreams specification.

It also appears that this practice is limited to the predefined streams (stdin/cin, stdout/cout, stderr/cerr). P.J. Plaugher suggested limiting the iostreams specification to only the predefined streams. Uwe Steinmueller pointed out that this raises a question about the validity of reopening a standard stream to a file. Although the issue needs more discussion, this appears to be a workable solution.

18.3 ISO C Library

The Library group discussed the proposal and agreed to recommend a vote by the full X3J16 committee to incorporate the text in the next version of the working paper. Most of the comments were minor.

The interaction between signals and exceptions is presently undefined. The group decided to leave it undefined, but continue investigating whether the standard should define the semantics of their interaction.

The types `wchar_t`, `ptrdiff_t`, and `size_t` are defined to be one of the integral types. This is adequate for C, but not precise enough for C++ (which relies on distinctness of types for function overloading). Changing the definitions of these types to be distinct from any existing integral types would cause an incompatibility with C. The group agreed to leave the wording as is, but continue investigating the issue.

Parameter passing using the facilities of stdarg.h were left unchanged. The wording is adequate for the built-in types, but programmer-defined types and references raise some questions. The group agreed to continue investigating whether this issue needed clarification and additional wording in the standard.

The C standard reserves function and macro names for future use. It might be useful to overload C++ versions of these functions. However, doing so might cause a conflict with the C standard. This issue will also be considered further.

18.4 String

Pete Becker, Uwe Steinmueller, and several others revised the proposal for a string class. They agreed to a description that included the following kinds of operations:

— construction
— assignment
— concatenation
— insert
— search, replace
— select
— compare
— convert
— memory management (e.g. pre-reserve)

The group discussed some of the open issues in the proposal. Overloading the member operations with operators increases the size of the string classes. There were some questions whether the operator overloading was necessary.

There was a discussion about the effect of locales on string operations, especially comparisons. P.J. Plaugher pointed out that comparison operations typically fall into three cases:

1) fast, "raw" byte comparisons
2) execution character set order
3) fully locale-sensitive

The group will analyze the locale-neutral operations and separately analyze the locale-sensitive operations to consider alternatives.

Uwe Steinmueller agreed to revise proposal.

18.5 Containers

18.5.1 Bit Sets

The group discussed the observation that the proposal contained two different sets of issues. One was the semantics of abstract set behavior vs. a collection of bit flags. The other was a fixed-size collection vs. a dynamically-sized collection. The group agreed to table consideration of an abstract set class, and concentrate on bit-oriented concrete type.

Chuck Allison agreed to revise the proposal.

18.5.2 Array

The group discussed the array proposal only briefly. The topic of an abstract iterator was tabled, pending a more complete array proposal.

Uwe Steinmueller agreed to revise proposal.

Work Plan

The Library working group will continue to refine proposals for submission to the full X3J16/WG21 committee. By the March, 1992 meeting in London, several documents should present proposals in almost-final form.

Documents

18 Library Introduction

Mike Vilot agreed to develop the wording for the introduction to the Library chapter, and to develop a Rationale statement for the library.

18.1 Language Support

Mike Vilot agreed to revise the proposal. The Library working group plans to submit the proposal to X3J16/WG21 for a vote on including it into the working document at the July, 1992 meeting in Toronto.

18.2 Input/Output

Jerry Schwarz agreed to revise the proposal. Since the proposal is so large, it is unlikely that the group will propose a vote on it at any of the next few meetings.

18.3 ISO C Library

Since X3J16 and WG21 voted to incorporate Steve Clamage's proposal, the only remaining work is to check that the next version of the working paper accurately reflects the proposal.

18.4 String

Uwe Steinmueller agreed to revise the proposal.

18.5 Containers

Chuck Allison agreed to revise the Bits/BitString proposal.

Uwe Steinmueller agreed to revise the DynamicArray proposal.