Doc No: N1028/95-0210
Date: Nov 12, 1996
Project: Programming Language C++
Reply to: Jerry Schwarz
          jerry@intrinsa.com

Proposed Changes to Clause 27 (iostreams)

```
*** lib-iostreams Tue Nov 12 11:00:01 1996
--- lib-ios Wed Nov 13 00:07:58 1996

============================================================
============ Fix typedef of iostream ===================
============================================================

***************
*** 208,214 ****
    typedef basic_streambuf<char> streambuf;
    typedef basic_istream<char>   istream;
    typedef basic_ostream<char>   ostream;
!   typedef basic_ostream<char>   iostream;

  .Ce
  .Cb
--- 208,214 ----
    typedef basic_streambuf<char> streambuf;
    typedef basic_istream<char>   istream;
    typedef basic_ostream<char>   ostream;
!   typedef basic_iostream<char>  iostream;

  .Ce
  .Cb

============================================================
========== add typedef for wiostream ==============
============================================================

***************
*** 229,234 ****
--- 229,235 ----
    typedef basic_streambuf<wchar_t> wstreambuf;
    typedef basic_istream<wchar_t>   wistream;
    typedef basic_ostream<wchar_t>   wostream;
+   typedef basic_iostream<wchar_t>  wiostream;

  .Ce
  .Cb

============================================================
==============  Ensure initialization of predefined streams ==========
============================================================

***************
*** 409,422 ****
  follows the same semantics as mixing such operations on
  .CW FILE s,
  as specified in Amendment 1 of the ISO C standard.
! .eN
! These objects need to be constructed and associations established
! before dynamic initialization of file scope variables is begun.
! .nE
  The objects are constructed, and the associations are established at some
```

```
time prior to or
  during first time an object of class
  .CW basic_ios<charT,traits>::Init
! is constructed.
  The objects are
  .I not
  destroyed during program execution.\*f
--- 410,420 ----
  follows the same semantics as mixing such operations on
  .CW FILE s,
  as specified in Amendment 1 of the ISO C standard.
! .\" issue 27-108
  The objects are constructed, and the associations are established at some
time prior to or
  during first time an object of class
  .CW basic_ios<charT,traits>::Init
! is constructed, and in any case before the body of main begins execution.
  The objects are
  .I not
  destroyed during program execution.\*f

==========================================================
==============  eliminate requiement that predefined streams be unbufferd
====
==========================================================

***************
*** 439,445 ****
  .P
  The object
  .CW cin
! controls input from an unbuffered stream buffer
  associated with the object
  .CW stdin ,
  declared in
--- 437,443 ----
  .P
  The object
  .CW cin
! controls input from a stream buffer
  associated with the object
  .CW stdin ,
  declared in
***************
*** 460,466 ****
  .P
  The object
  .CW cout
! controls output to an unbuffered stream buffer
  associated with the object
  .CW stdout ,
  declared in
--- 458,464 ----
  .P
  The object
  .CW cout
! controls output to a stream buffer
  associated with the object
  .CW stdout ,
  declared in
***************
*** 475,481 ****
  .P
  The object
```

```
  .CW cerr
! controls output to an unbuffered stream buffer
  associated with the object
  .CW stderr ,
  declared in
--- 473,479 ----
  .P
  The object
  .CW cerr
! controls output to a stream buffer
  associated with the object
  .CW stderr ,
  declared in
***************
*** 517,523 ****
  .P
  The object
  .CW wcin
! controls input from an unbuffered stream buffer
  associated with the object
  .CW stdin ,
  declared in
--- 515,521 ----
  .P
  The object
  .CW wcin
! controls input from a stream buffer
  associated with the object
  .CW stdin ,
  declared in
***************
*** 538,544 ****
  .P
  The object
  .CW wcout
! controls output to an unbuffered stream buffer
  associated with the object
  .CW stdout ,
  declared in
--- 536,542 ----
  .P
  The object
  .CW wcout
! controls output to a stream buffer
  associated with the object
  .CW stdout ,
  declared in
***************
*** 553,559 ****
  .P
  The object
  .CW wcerr
! controls output to an unbuffered stream buffer
  associated with the object
  .CW stderr ,
  declared in
--- 551,557 ----
  .P
  The object
  .CW wcerr
! controls output to a stream buffer
  associated with the object
  .CW stderr ,
  declared in
```

```
==========================================================
==========  add include <iosfwd> to synopsis of <ios> ============
==========================================================

***************
*** 589,594 ****
--- 587,595 ----
  .ix "[<ios>]"
  .DE
  .Cb
+ #include <iosfwd>
+ .Ce
+ .Cb
  namespace std {
    typedef \&\f4OFF_T\&\fP  streamoff;
    typedef \&\f4SZ_T\&\fP streamsize;
***************
*** 597,604 ****
    class ios_base;
    template <class charT, class traits = char_traits<charT> >
      class basic_ios;
-   typedef basic_ios<char>     ios;
-   typedef basic_ios<wchar_t> wios;
  .Ce
  .Cb
  \f6// _lib.std.ios.manip_, manipulators:\fP
--- 598,603 ----

==========================================================
=============  Describe failure of iword and pword  =================
==========================================================

***************
*** 1364,1376 ****
  member with a different index, after a call to its
  .CW copyfmt
  member,
! or when the object is destroyed.
  .Fs
  An implementation is free to implement both the integer
  array pointed at by \&\f6iarray\fP\& and the pointer array pointed at by
  \&\f6parray\fP\& as sparse data structures, possibly with a one-element
  cache for each.
  .Fe
  .La Returns:
  .CW \&\f6iarray\fP[\f6idx\fP] .
  .\"
--- 1363,1384 ----
  member with a different index, after a call to its
  .CW copyfmt
  member,
! or when the object is destroyed.\*f
  .Fs
  An implementation is free to implement both the integer
  array pointed at by \&\f6iarray\fP\& and the pointer array pointed at by
  \&\f6parray\fP\& as sparse data structures, possibly with a one-element
  cache for each.
  .Fe
+ .\" issue 27-109
+ .La ""
+ If the function fails\*f
+ .Fs
+ for example, because it cannot allocate space.
```

```
+ .Fe
+ it sets
+ .CW badbit ,
+ which may throw an exception.
  .La Returns:
  .CW \&\f6iarray\fP[\f6idx\fP] .
  .\"

==============================================================
================ fix definition of ios destructor ==============
==============================================================

***************
*** 1448,1454 ****
  Calls each registered callback pair
  .CW "(\f6fn\fP,\f6index\fP)"
  (_lib.ios.base.callback_) as
! .CW (*\f6fn\fP)(imbue_event,*this,\f6index\fP)
  at such time that any
  .CW ios_base
  member function called from within
--- 1456,1462 ----
  Calls each registered callback pair
  .CW "(\f6fn\fP,\f6index\fP)"
  (_lib.ios.base.callback_) as
! .CW (*\f6fn\fP)(erase_event,*this,\f6index\fP)
  at such time that any
  .CW ios_base
  member function called from within

==============================================================
====== Turn get with default into variant. Remove use of traits::newline
=====
==============================================================

***************
*** 3372,3381 ****
      streamsize gcount() const;
      int_type get();
      basic_istream<charT,traits>& get(char_type& \f6c\fP);
      basic_istream<charT,traits>& get(char_type* \f6s\fP, streamsize
\f6n\fP,
!                      char_type \f6delim\fP = traits::newline());
      basic_istream<charT,traits>& get(basic_streambuf<char_type,traits>&
\f6sb\fP,
!                      char_type \f6delim\fP = traits::newline());
  .Ce
  .Cb
      basic_istream<charT,traits>& getline(char_type* \f6s\fP, streamsize
\f6n\fP);
--- 3380,3391 ----
      streamsize gcount() const;
      int_type get();
      basic_istream<charT,traits>& get(char_type& \f6c\fP);
+     basic_istream<charT,traits>& get(char_type* \f6s\fP, streamsize
\f6n\fP);
      basic_istream<charT,traits>& get(char_type* \f6s\fP, streamsize
\f6n\fP,
!                      char_type \f6delim\fP);
!     basic_istream<charT,traits>& get(basic_streambuf<char_type,traits>&
\f6sb\fP);
      basic_istream<charT,traits>& get(basic_streambuf<char_type,traits>&
\f6sb\fP,
!                      char_type \f6delim\fP);
```

```
  .Ce
  .Cb
      basic_istream<charT,traits>& getline(char_type* \f6s\fP, streamsize
\f6n\fP);
***************
*** 3946,3952 ****
  .\"
  .Pb
  basic_istream<charT,traits>& get(char_type*  \f6s\fP, streamsize \f6n\fP,
!                   char_type \f6delim\fP = traits::newline());
  .Pe
  .La Effects:
  Extracts characters and stores them
--- 3956,3962 ----
  .\"
  .Pb
  basic_istream<charT,traits>& get(char_type*  \f6s\fP, streamsize \f6n\fP,
!                   char_type \f6delim\fP );
  .Pe
  .La Effects:
  Extracts characters and stores them
***************
*** 3982,3989 ****
  .CW *this .
  .\"
  .Pb
  basic_istream<charT,traits>& get(basic_streambuf<char_type,traits>&
\f6sb\fP,
!                   char_type \f6delim\fP = traits::newline());
  .Pe
  .La Effects:
  Extracts characters and inserts them
--- 3992,4008 ----
  .CW *this .
  .\"
  .Pb
+ basic_istream<charT,traits>& get(char_type*  \f6s\fP, streamsize \f6n\fP)
+ .Pe
+ .La "Effects:"
+ Calls
+ .CW "getline(s,n,widen('\en'))"
+ .La "Returns:"
+ Value returned by the call.
+ .\"
+ .Pb
  basic_istream<charT,traits>& get(basic_streambuf<char_type,traits>&
\f6sb\fP,
!                   char_type \f6delim\fP );
  .Pe
  .La Effects:
  Extracts characters and inserts them
***************
*** 4011,4016 ****
--- 4030,4045 ----
  .La Returns:
  .CW *this .
  .\"
+ .Pb
+ basic_istream<charT,traits>& get(basic_streambuf<char_type,traits>&
\f6sb\fP);
+ .Pe
+ .La "Effects:"
+ Calls
+ .CW "getline(s,n,widen('\en'))"
```

```
+ .La "Returns:"
+ Value returned by the call.
+ .\"
+ .\"
  .ix "[basic__istream] [getline]"
  .Pb
  basic_istream<charT,traits>& getline(char_type* \f6s\fP, streamsize
\f6n\fP,
***************
*** 4106,4112 ****
  basic_istream<charT,traits>& getline(char_type* \f6s\fP, streamsize
\f6n\fP);
  .Pe
  .La Returns:
! .CW "getline(p,n,widen('\en'))"
  .\"
  .ix "[basic__istream] [ignore]"
  .Pb
--- 4135,4141 ----
  basic_istream<charT,traits>& getline(char_type* \f6s\fP, streamsize
\f6n\fP);
  .Pe
  .La Returns:
! .CW "getline(s,n,widen('\en'))"
  .\"
  .ix "[basic__istream] [ignore]"
  .Pb

==========================================================
========= fix reference to failbit ============
==========================================================

***************
*** 4709,4715 ****
  the function endeavors
  to generate the requested output.
  If the generation fails, then the formatted output function does
! .CW setstate(ios::fail)
  which might throw and exception.
  If an exception is thrown during output then
  .CW ios::badbit
--- 4738,4744 ----
  the function endeavors
  to generate the requested output.
  If the generation fails, then the formatted output function does
! .CW setstate(ios::failbit)
  which might throw and exception.
  If an exception is thrown during output then
  .CW ios::badbit

==========================================================
========== fix typo =============
==========================================================

***************
*** 4980,4986 ****
  .CW bool ,
  the function endeavors
  to generate the requested output.
! In any case, the formatted output function ends
  by destroying the
  .CW sentry
  object,
--- 5009,5015 ----
```

```
  .CW bool ,
  the function endeavors
  to generate the requested output.
! In any case, the unformatted output function ends
  by destroying the
  .CW sentry
  object,

===================================================
========== replace traits::newline with widen('\n') ==========
===================================================

***************
*** 5064,5070 ****
  .Pe
  .La Effects:
  Calls
! .CW \&\f6os\fP.put(traits::newline()) ,
  then
  .CW \&\f6os\fP.flush() .
  .La Returns:
--- 5093,5099 ----
  .Pe
  .La Effects:
  Calls
! .CW \&\f6os\fP.put(os.widen('\en') ) ,
  then
  .CW \&\f6os\fP.flush() .
  .La Returns:

===================================================
==============  fix argument of setfill (Stockholm motion) ===========
===================================================

***************
*** 5221,5227 ****
  .\"----
  .ix "[setfill]"
  .Pb
! \f4smanip\fP setfill(int \f6c\fP);
  .Pe
  .La Returns:
  An object \f6s\fP of implementation specified type such that if
--- 5250,5256 ----
  .\"----
  .ix "[setfill]"
  .Pb
! \f4smanip\fP setfill(char_type \f6c\fP);
  .Pe
  .La Returns:
  An object \f6s\fP of implementation specified type such that if

===================================================
============  add extra template parameters to use of basic_string ======
===================================================

***************
*** 5434,5440 ****
  The function allocates no array object.
  .\"
  .Pb
! explicit basic_stringbuf(const basic_string<char_type>& \f6str\fP,
                 ios_base::openmode \f6which\fP = ios_base::in |
ios_base::out);
```

```
  .Pe
  .La Effects:
--- 5463,5469 ----
  The function allocates no array object.
  .\"
  .Pb
! explicit basic_stringbuf(const basic_string<charT,traits,Allocator>&
\f6str\fP,
                 ios_base::openmode \f6which\fP = ios_base::in |
ios_base::out);
  .Pe
  .La Effects:
***************
*** 5464,5470 ****
  .\"
  .ix "[basic__stringbuf] [str]"
  .Pb
! basic_string<char_type> str() const;
  .Pe
  .La Returns:
  A
--- 5493,5499 ----
  .\"
  .ix "[basic__stringbuf] [str]"
  .Pb
! basic_string<charT,traits,Allocator> str() const;
  .Pe
  .La Returns:
  A
***************
*** 5477,5483 ****
  In case of an empty underlying character sequence, the function returns
  .CW basic_string<charT,traits,Allocator>() .
  .Pb
! void str(const basic_string<char_type>& \f6s\fP);
  .Pe
  .La Effects:
  If the
--- 5506,5512 ----
  In case of an empty underlying character sequence, the function returns
  .CW basic_string<charT,traits,Allocator>() .
  .Pb
! void str(const basic_string<charT,traits,Allocator>& \f6s\fP);
  .Pe
  .La Effects:
  If the
***************
*** 5766,5782 ****
  .Cb
    \f6// _lib.istringstream.cons_ Constructors:\fP
      explicit basic_istringstream(ios_base::openmode \f6which\fP =
ios_base::in);
!     explicit basic_istringstream(const basic_string<charT>& \f6str\fP,
!                                  ios_base::openmode \f6which\fP =
ios_base::in);
  .Ce
  .Cb
    \f6// _lib.istringstream.members_ Members:\fP
      basic_stringbuf<charT,traits,Allocator>* rdbuf() const;
  .Ce
  .Cb
!     basic_string<charT> str() const;
!     void                str(const basic_string<charT>& \f6s\fP);
!   private:
```

```
  //  basic_stringbuf<charT,traits,Allocator> \f6sb\fP;     \f4exposition
only\fP
    };
  }
--- 5795,5812 ----
  .Cb
    \f6// _lib.istringstream.cons_ Constructors:\fP
      explicit basic_istringstream(ios_base::openmode \f6which\fP =
ios_base::in);
!     explicit basic_istringstream(
!                          const basic_string<charT,traits,Allocator>&
\f6str\fP,
!                          ios_base::openmode \f6which\fP = ios_base::in);
  .Ce
  .Cb
    \f6// _lib.istringstream.members_ Members:\fP
      basic_stringbuf<charT,traits,Allocator>* rdbuf() const;
  .Ce
  .Cb
!     basic_string<charT,traits,Allocator> str() const;
!     void str(const basic_string<charT,traits,Allocator>& \f6s\fP);
! private:
  //  basic_stringbuf<charT,traits,Allocator> \f6sb\fP;     \f4exposition
only\fP
    };
  }
***************
*** 5832,5838 ****
  .\"
  .ix "[basic__istringstream] [str]"
  .Pb
! basic_string<charT> str() const;
  .Pe
  .La Returns:
  .CW rdbuf()->str() .\*f
--- 5862,5868 ----
  .\"
  .ix "[basic__istringstream] [str]"
  .Pb
! basic_string<charT,traits,Allocator> str() const;
  .Pe
  .La Returns:
  .CW rdbuf()->str() .\*f
***************
*** 5843,5849 ****
  .Fe
  .\"
  .Pb
! void str(const basic_string<charT>& \f6s\fP);
  .Pe
  .La Effects:
  Calls
--- 5873,5879 ----
  .Fe
  .\"
  .Pb
! void str(const basic_string<charT,traits,Allocator>& \f6s\fP);
  .Pe
  .La Effects:
  Calls
***************
*** 5866,5873 ****
  .Cb
    \f6// _lib.ostringstream.cons_ Constructors/destructor:\fP
```

```
      explicit basic_ostringstream(ios_base::openmode \f6which\fP =
ios_base::out);
!     explicit basic_ostringstream(const basic_string<charT>& \f6str\fP,
!                                  ios_base::openmode \f6which\fP =
ios_base::out);
      virtual ~basic_ostringstream();
  .Ce
  .Cb
--- 5896,5904 ----
  .Cb
    \f6// _lib.ostringstream.cons_ Constructors/destructor:\fP
      explicit basic_ostringstream(ios_base::openmode \f6which\fP =
ios_base::out);
!     explicit basic_ostringstream(
!                          const basic_string<charT,traits,Allocator>&
\f6str\fP,
!                          ios_base::openmode \f6which\fP = ios_base::out);
      virtual ~basic_ostringstream();
  .Ce
  .Cb
***************
*** 5971,5984 ****
      explicit basic_stringstream(
          ios_base::openmode which = ios_base::out|ios_base::in);
      explicit basic_stringstream(
!         const basic_string<charT>& str,
          ios_base::openmode which = ios_base::out|ios_base::in);
  .Ce
  .Cb
      \f6// Members:
      basic_stringbuf<charT,traits,Allocator>* rdbuf() const;
!     basic_string<charT> str() const;
!     void str(const basic_string<charT>& str);
  .Ce
  .Cb
    private:
--- 6002,6015 ----
      explicit basic_stringstream(
          ios_base::openmode which = ios_base::out|ios_base::in);
      explicit basic_stringstream(
!         const basic_string<charT,traits,Allocator>& str,
          ios_base::openmode which = ios_base::out|ios_base::in);
  .Ce
  .Cb
      \f6// Members:
      basic_stringbuf<charT,traits,Allocator>* rdbuf() const;
!     basic_string<charT,traits,Allocator> str() const;
!     void str(const basic_string<charT,traits,Allocator>& str);
  .Ce
  .Cb
    private:
***************
*** 5989,5995 ****
  The template class
  .CW basic_stringstream<charT,traits>
  supports reading and writing from objects of class
! .CW basic_string<charT,traits> .
  It uses a
  .CW basic_stringbuf<charT,traits,Allocator>
  object to control the associated sequence.
--- 6020,6026 ----
  The template class
  .CW basic_stringstream<charT,traits>
  supports reading and writing from objects of class
```

```
! .CW basic_string<charT,traits,Allocator> .
  It uses a
  .CW basic_stringbuf<charT,traits,Allocator>
  object to control the associated sequence.
***************
*** 6013,6019 ****
  .CW basic_stringbuf<charT,traits,Allocator>(which).
  .Pb
  explicit basic_stringstream(
!     const basic_string<charT>& str,
      ios_base::openmode which = ios_base::out|iosbase::in);
  .Pe
  .La Effects:
--- 6044,6050 ----
  .CW basic_stringbuf<charT,traits,Allocator>(which).
  .Pb
  explicit basic_stringstream(
!     const basic_string<charT,traits,Allocator>& str,
      ios_base::openmode which = ios_base::out|iosbase::in);
  .Pe
  .La Effects:
***************
*** 6034,6040 ****
  .CW &sb
  .ix "[basic__stringstream] [str]"
  .Pb
! basic_string<charT> str() const;
  .Pe
  .La Returns:
  .CW rdbuf()->str() .\*f
--- 6065,6071 ----
  .CW &sb
  .ix "[basic__stringstream] [str]"
  .Pb
! basic_string<charT,traits,Allocator> str() const;
  .Pe
  .La Returns:
  .CW rdbuf()->str() .\*f
***************
*** 6044,6050 ****
  .CW object.
  .Fe
  .Pb
! void str(const basic_string<charT>& str);
  .Pe
  .La Effects:
  Calls
--- 6075,6081 ----
  .CW object.
  .Fe
  .Pb
! void str(const basic_string<charT,traits,Allocator>& str);
  .Pe
  .La Effects:
  Calls

==========================================================
=============  Add declaration of fstream and wfstream typedefs =======
==========================================================

***************
*** 6079,6084 ****
--- 6110,6120 ----
      class basic_ofstream;
```

```
    typedef basic_ofstream<char>    ofstream;
    typedef basic_ofstream<wchar_t> wofstream;
+
+   template <class charT, class traits = char_traits<charT> >
+     class basic_fstream;
+   typedef basic_fstream<char>     fstream;
+   typedef basic_fstream<wchar_t> wfstream;
  }
  .Ce
  .ix "[filebuf]"

==============================================================
================== Close should flush and should use unshift =========
==============================================================

***************
*** 6371,6377 ****
  If
  .CW "is_open() == false" ,
  returns a null pointer.
! Otherwise, closes the file
  (``as if'' by calling
  .CW ::fclose(\f6file\fP) ).\*f
  .ix "[fclose]"
--- 6407,6432 ----
  If
  .CW "is_open() == false" ,
  returns a null pointer.
! If a put area exists, calls
! .CW overflow(EOF)
! to flush characters.
! If the last virtual between
! .CW "underflow" ,
! .CW "overflow" ,
! .CW "seekoff" ,
! and
! .CW "seekpos"
! was
! .CW overflow
! then calls
! .Cb
! (use_facet<codecvt<charT, char, traits_type::state_type>
>getloc()).do_unshift
! .Ce
! to determine a termination sequence, inserts those characters and
! calls
! .CW overflow(EOF)
! again.
! Finally it closes the file
  (``as if'' by calling
  .CW ::fclose(\f6file\fP) ).\*f
  .ix "[fclose]"
***************
*** 6384,6389 ****
--- 6439,6451 ----
  .ix "[<cstdio>]"
  (_lib.c.files_).
  .Fe
+ If any of the calls to
+ .CW overflow
+ or
+ .CW "::fclose"
+ then
+ .CW close
```

```
+ fails
  .La Returns:
  .CW this
  on success, a null pointer otherwise.

==========================================================
======= Fix semantics of filebuf::seekoff to understand codecvt =======
======= encoding, and fail on files with variable width encodings =====
==========================================================

***************
*** 6603,6634 ****
                    = ios_base::in | ios_base::out);
  .Pe
  .La Effects:
! The current state is determined as follows:
! If the the last operation was
! .CW overflow() ,
! the current state is obtained by combining the shiftstate contained in
! .CW st
! with the current
! position (
! .CW fpos_t )
! of the file.
! If the last operation was
! .CW underflow() ,
! the shiftstate and file position are
! determined (according to whatever means they were saved by
! .CW underflow() )
! as corresponding to
! .CW pptr() .
! .br
! Then, alters the stream position within the controlled
! sequences, if possible, as described below.
  .br
  If
  .CW "is_open() == false" ,
  the positioning operation fails.
! Otherwise, repositions within the associated file
  (``as if'' by calling
! .CW ::fseek(\f6file\fP,\f6off\fP,\f6whence\fP) .\*f
  .Fs
  The macros
  .CW SEEK_SET ,
--- 6665,6685 ----
                    = ios_base::in | ios_base::out);
  .Pe
  .La Effects:
! Does
! .Cb
! int e =
!   (use_facet<codecvt<charT, char,traits_type::state_type> >
!                    (getloc())).encoding()
! .Ce
  .br
  If
  .CW "is_open() == false" ,
  the positioning operation fails.
! Otherwise, if
! .CW "e>0"
! repositions within the associated file
  (``as if'' by calling
! .CW ::fseek(\f6file\fP,\f6e\fP*\f6off\fP,\f6whence\fP) .\*f
  .Fs
```

```
  The macros
  .CW SEEK_SET ,

==========================================================
==========  add call to unshift to semantics of seek when variable width
=====
==========  encodings are in use
=====
==========================================================

***************
*** 6642,6647 ****
--- 6693,6704 ----
  (_lib.c.files_).
  .ix "[<cstdio>]"
  .Fe
+ Otherwise if
+ .CW \f6off\fP==0
+ repositions within the associated file
+ (``as if'' by calling
+ .CW ::fseek(\f6file\fP,0,\f6whence\fP) .
+ Other the positioning operation fails.
  .La "Notes:"
  The function determines one of three values for the
  argument \&\f6whence\fP\&, of type
***************
*** 6690,6695 ****
--- 6747,6765 ----
                                = ios_base::in | ios_base::out);
  .Pe
  .La Effects:
+ If the last virtual between
+ .CW "underflow" ,
+ .CW "overflow" ,
+ .CW "seekoff" ,
+ and
+ .CW "seekpos"
+ was
+ .CW overflow
+ then calls
+ .Cb
+ (use_facet<codecvt<charT, char, traits_type::state_type>
>getloc()).do_unshift
+ .Ce
+ to determine a termination sequence and inserts those characters.
  Alters the file position, if possible, to correspond to the position
stored
  in \f6sp\fP (as described below).
  .LI

==========================================================
===========  Specify semantics of filebuf::sync       =====
===========  and add precondition to filebuf::imbue   =====
==========================================================

***************
*** 6714,6725 ****
--- 6784,6810 ----
  .Pb
  int sync();
  .Pe
+ .La "effects"
+ If a put area exists calls
+ .CW filebuf::overflow
```

```
+ to write the characters to the file.
+ If a get area exists the effect is implementation-defined.
+ .ix "implementation-defined [filebuf]"
+ .La
  .\"----
  .\" 95-0074/N0674
  .ix "[basic__filebuf] [imbue]"
  .Pb
  void imbue(const locale& \f6loc\fP);
  .Pe
+ .La "Precondition"
+ If the file is not positioned at its  beginning and the encoding of
\f6loc\fP
+ (as reported by
+ .Cb
+ use_facet<codecvt<charT,char,traits_type::state_type>
>(getloc()).encoding()
+ .Ce
+ then that facet is the same as the corresponding facet of
+ .CW getloc() .
  .\"----
  .H4 "Template class \&\f7basic_ifstream\fP\&" lib.ifstream
  .ix "[basic__ifstream]"
```