MPI datatype	C datatype
MPI_CHAR	[ticket63.][signed]char (treated as printable character)
MPI_SHORT	signed short int
MPI_INT	signed int
MPI_LONG	signed long int
MPI_LONG_LONG_INT	signed long long int
MPI_LONG_LONG (as a synonym)	signed long long int
MPI_SIGNED_CHAR	signed char (treated as integral value)
MPI_UNSIGNED_CHAR	unsigned char (treated as integral value)
MPI_UNSIGNED_SHORT	unsigned short int
MPI_UNSIGNED	unsigned int
MPI_UNSIGNED_LONG	unsigned long int
MPI_UNSIGNED_LONG_LONG	unsigned long long int
MPI_FLOAT	float
MPI_DOUBLE	double
MPI_LONG_DOUBLE	long double
MPI_WCHAR	wchar_t (defined in <stddef.h>) (treated as printable character)
[ticket18.]MPI_C_BOOL	_Bool
[ticket18.]MPI_INT8_T	int8_t
[ticket18.]MPI_INT16_T	int16_t
[ticket18.]MPI_INT32_T	int32_t
[ticket18.]MPI_INT64_T	int64_t
[ticket18.]MPI_UINT8_T	uint8_t
[ticket18.]MPI_UINT16_T	uint16_t
[ticket18.]MPI_UINT32_T	uint32_t
[ticket18.]MPI_UINT64_T	uint64_t
[ticket18.]MPI_C_COMPLEX	float_Complex
[ticket18.]MPI_C_FLOAT_COMPLEX (as a synonym)	float_Complex
[ticket18.]MPI_C_DOUBLE_COMPLEX	double_Complex
[ticket18.]MPI_C_LONG_DOUBLE_COMPLEX	long double_Complex
MPI_BYTE	
MPI_PACKED	

Table 3.2: Predefined MPI datatypes corresponding to C datatypes

Rationale. The datatypes MPI_C_BOOL, MPI_INT8_T, MPI_INT16_T, MPI_INT32_T, MPI_UINT8_T, MPI_UINT16_T, MPI_UINT32_T, MPI_C_COMPLEX, MPI_C_FLOAT_COMPLEX, MPI_C_DOUBLE_COMPLEX, and MPI_C_LONG_DOUBLE_COMPLEX have no corresponding C++ bindings. This was intentionally done to avoid potential collisions with the C preprocessor and namespaced C++ names. C++ applications can use the C bindings with no loss of functionality. (*End of rationale.*)

Named Predefined Datatypes		C/C++ types
[ticket107.]C type: `MPI_Datatype`	C++ type: `MPI::Datatype`	
[ticket107.]Fortran type: `INTEGER`		
[ticket63.]MPI_CHAR	MPI::CHAR	`char` (treated as printable chara
MPI_SHORT	MPI::SHORT	`signed short int`
MPI_INT	MPI::INT	`signed int`
MPI_LONG	MPI::LONG	`signed long`
MPI_LONG_LONG_INT	MPI::LONG_LONG_INT	`signed long long`
MPI_LONG_LONG	MPI::LONG_LONG	`long long` (synonym)
MPI_SIGNED_CHAR	MPI::SIGNED_CHAR	`signed char` (treated as integral value)
MPI_UNSIGNED_CHAR	MPI::UNSIGNED_CHAR	`unsigned char` (treated as integral value)
MPI_UNSIGNED_SHORT	MPI::UNSIGNED_SHORT	`unsigned short`
MPI_UNSIGNED	MPI::UNSIGNED	`unsigned int`
MPI_UNSIGNED_LONG	MPI::UNSIGNED_LONG	`unsigned long`
MPI_UNSIGNED_LONG_LONG	MPI::UNSIGNED_LONG_LONG	`unsigned long long`
MPI_FLOAT	MPI::FLOAT	`float`
MPI_DOUBLE	MPI::DOUBLE	`double`
MPI_LONG_DOUBLE	MPI::LONG_DOUBLE	`long double`
MPI_WCHAR	MPI::WCHAR	`wchar_t` (defined in `<stddef.h>`) (treated as printable chara
[ticket18.]MPI_C_BOOL	[ticket18.](use C datatype handle)	[ticket18.]`_Bool`
[ticket18.]MPI_INT8_T	[ticket18.](use C datatype handle)	[ticket18.]`int8_t`
[ticket18.]MPI_INT16_T	[ticket18.](use C datatype handle)	[ticket18.]`int16_t`
[ticket18.]MPI_INT32_T	[ticket18.](use C datatype handle)	[ticket18.]`int32_t`
[ticket18.]MPI_INT64_T	[ticket18.](use C datatype handle)	[ticket18.]`int64_t`
[ticket18.]MPI_UINT8_T	[ticket18.](use C datatype handle)	[ticket18.]`uint8_t`
[ticket18.]MPI_UINT16_T	[ticket18.](use C datatype handle)	[ticket18.]`uint16_t`
[ticket18.]MPI_UINT32_T	[ticket18.](use C datatype handle)	[ticket18.]`uint32_t`
[ticket18.]MPI_UINT64_T	[ticket18.](use C datatype handle)	[ticket18.]`uint64_t`
[ticket18.]MPI_AINT	[ticket18.](use C datatype handle)	[ticket18.]`MPI_Aint`
[ticket18.]MPI_OFFSET	[ticket18.](use C datatype handle)	[ticket18.]`MPI_Offset`
[ticket18.]MPI_C_COMPLEX	[ticket18.](use C datatype handle)	[ticket18.]`float _Complex`
[ticket18.]MPI_C_FLOAT_COMPLEX	[ticket18.](use C datatype handle)	[ticket18.]`float _Complex`
[ticket18.]MPI_C_DOUBLE_COMPLEX	[ticket18.](use C datatype handle)	[ticket18.]`double _Comple`
[ticket18.]MPI_C_LONG_DOUBLE_COMPLEX	[ticket18.](use C datatype handle)	[ticket18.]`long double _C`
MPI_BYTE	MPI::BYTE	(any C/C++ type)
MPI_PACKED	MPI::PACKED	(any C/C++ type)

ticket4.

Annex B

Change-Log

This annex summarizes changes from the previous version of the MPI standard to the version presented by this document. [Only changes (i.e., clarifications and new features) are presented that may cause implementation effort in the MPI libraries.] Only significant changes (i.e., clarifications and new features) that might either require implementation effort in the MPI libraries or change the understanding of MPI from a user's perspective are presented. Editorial modifications, formatting, typo corrections and minor clarifications are not shown.

B.1 Changes from Version 2.1 to Version 2.2

1. Section 2.5.4 on page 14.
   It is now guaranteed that predefined named constant handles (as other constants) can be used in initialization expressions or assignments, i.e., also before the call to MPI_INIT.

2. Section 2.6 on page 16, Section 2.6.4 on page 19, and Section 16.1 on page 485.
   The C++ language bindings have been deprecated and may be removed in a future version of the MPI specification.

3. Section 3.2.2 on page 29.
   MPI_CHAR for printable characters is now defined for C type char (instead of signed char). This change should not have any impact on applications nor on MPI libraries (except some comment lines), because printable characters could and can be stored in any of the C types char, signed char, and unsigned char, and MPI_CHAR is not allowed for predefined reduction operations.

4. Section 3.2.2 on page 29.
   MPI_(U)INT{8,16,32,64}_T, MPI_AINT, MPI_OFFSET, MPI_C_BOOL, MPI_C_COMPLEX, MPI_C_FLOAT_COMPLEX, MPI_C_DOUBLE_COMPLEX, and MPI_C_LONG_DOUBLE_COMPLEX are now valid predefined MPI datatypes.

5. Section 3.4 on page 40, Section 3.7.2 on page 52, Section 3.9 on page 72, and Section 5.1 on page 135.
   The read access restriction on the send buffer for blocking, non blocking and collective API has been lifted. It is permitted to access for read the send buffer while the operation is in progress.

ticket99.
ticket99.
ticket100.
ticket65.
ticket150.
ticket63.
ticket18.
ticket45+98.
ticket50.