Annex B

Change-Log

This annex summarizes changes from the previous version of the MPI standard to the version presented by this document. 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 2.2 to Version 3.0

1. Chapter 5 on page 133 and Section 5.12 on page 190.
   Added nonblocking interfaces to all collective operations. ticket109.

2. Section 2.5.8 on page 16, Section 3.2.2 on page 29, Section 3.3 on page 31, Section 5.9.2 on page 172, Section A.1.1 on page 545.
   Introduce MPI_Count, a type large enough to represent element counts in memory, file views, etc. ticket265.

3. Section 4.1.5 on page 97.
   A new function, MPI_TYPE_SIZE_X, returns the size of an MPI datatype as an MPI_Count value.

4. Section 4.1.7 on page 99.
   A new function, MPI_TYPE_GET_EXTENT_X, returns the lower bound and extent of an MPI datatype as MPI_Count values.

5. Section 4.1.8 on page 101.
   A new function, MPI_TYPE_GET_TRUE_EXTENT_X, returns the true lower bound and extent of an MPI datatype as MPI_Count values.

6. Section 4.1.11 on page 105.
   A new function, MPI_GET_ELEMENTS_X, returns the number of basic elements received as an MPI_Count value.

7. Section 12.3 on page 412.
   A new function, MPI_STATUS_SET_ELEMENTS_X, modifies the opaque part of MPI_STATUS so that a call to MPI_GET_ELEMENTS_X returns the provided MPI_Count value.

8. Section 3.2.5 on page 34, Section 4.1.5 on page 97, Section 4.1.11 on page 105, Section 4.2 on page 127.
   The functions MPI_GET_COUNT, MPI_TYPE_SIZE, and MPI_GET_ELEMENTS are now defined to set the count parameter to MPI_UNDEFINED when that parameter would overflows. The function MPI_PACK_SIZE is now defined to set the size parameter to MPI_UNDEFINED when that parameter would overflow.

B.2 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 499.
   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 41, Section 3.7.2 on page 52, Section 3.9 on page 71, and Section 5.1 on page 133.
   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.

6. Section 3.7 on page 50.
   The Advice to users for IBSEND and IRSEND was slightly changed.

7. Section 3.7.3 on page 55.
   The advice to free an active request was removed in the Advice to users for MPI_REQUEST_FREE.

8. Section 3.7.6 on page 66.
   MPI_REQUEST_GET_STATUS changed to permit inactive or null requests as input.

9. Section 5.8 on page 165.
   "In place" option is added to MPI_ALLTOALL, MPI_ALLTOALLV, and MPI_ALLTOALLW for intracommunicators.