28. Section 16.1.6 on page 453.
    New C++ versions of the Fortran specified-length complex types must be defined and implemented.

29. Section 16.3.7 on page 486.
    !!!TODO!!! See Ticket – proposed text: The description was modified that it only describes how an MPI implementation behaves, but not how it must be implemented internally. The erroneous MPI-2.1 Example 16.17 was replaced with three new examples [**...insert reference to example numbers...**] on page [**...pageref...**] explicitly detailing cross-language attribute behavior. Implementations that matched the behavior of the old example will need to be updated.

30. Annex A.1.1 on page 490.
    Removed type MPI::Fint (compare MPI_Fint in Section A.1.2 on page 498).

31. Annex A.1.1 on page 490. Table *Named Predefined Datatypes*.
    Added MPI_(U)INT{8,16,32,64}_T, MPI_AINT, MPI_OFFSET, MPI_C_BOOL, MPI_C_FLOAT_COMPLEX, MPI_C_COMPLEX, MPI_C_DOUBLE_COMPLEX, and MPI_C_LONG_DOUBLE_COMPLEX are added as predefined datatypes.

2.2 Changes from Version 2.0 to Version 2.1

1. Section 3.2.2 on page 27, Section 16.1.6 on page 453, and Annex A.1 on page 490.
   In addition, the MPI_LONG_LONG should be added as an optional type; it is a synonym for MPI_LONG_LONG_INT.

2. Section 3.2.2 on page 27, Section 16.1.6 on page 453, and Annex A.1 on page 490.
   MPI_LONG_LONG_INT, MPI_LONG_LONG (as synonym), MPI_UNSIGNED_LONG_LONG, MPI_SIGNED_CHAR, and MPI_WCHAR are moved from optional to official and they are therefore defined for all three language bindings.

3. Section 3.2.5 on page 31.
   MPI_GET_COUNT with zero-length datatypes: The value returned as the count argument of MPI_GET_COUNT for a datatype of length zero where zero bytes have been transferred is zero. If the number of bytes transferred is greater than zero, MPI_UNDEFINED is returned.

4. Section 4.1 on page 77.
   General rule about derived datatypes: Most datatype constructors have replication count or block length arguments. Allowed values are non-negative integers. If the value is zero, no elements are generated in the type map and there is no effect on datatype bounds or extent.

5. Section 4.3 on page 127.
   MPI_BYTE should be used to send and receive data that is packed using MPI_PACK_EXTERNAL.

6. Section 5.9.6 on page 171.
   If comm is an intercommunicator in MPI_ALLREDUCE, then both groups should