Chapter2: MPI Terms and Conventions changes: MPI-3.1 → MPI-4.0-RC

- 11:23-15:19 **#96/PR116 Update to Semantic Terms Section (2nd vote Sep 2020)**
- 19:12 new compile-time constant in MPI-4.0: MPI_F_STATUS_SIZE (C only)
- 20:20-29, 21:6-30, 22:45-47 **Big Count (Adresses, Counts, Fortran mpi_08 only)**
- 20:44-45 new at Counts: sentence about timestamps (Tools)
- 22:1 2.6.1: Deprecated and Removed Interfaces (3.1: … Names and Functions)
- 23:26-30,34 deprecated functions since MPI-3.2 & MPI-4.0
- 25:2-9,30-38,41,46 **Error Handling**

- cleanup: whitespaces, macros --> not visible * DONE (see github diff)
- cleanup: remove blanks, no blanks around – --> visible * 16:1 Datatypes (no blank)
- cleanup: typos, capitalization, fontsize, macros --> visible * 18:18 datatype (no blank)

- 10:9-10 **MPI identifiers are limited to 30 characters --> no longer valid !**

  MPI identifiers are limited to 30 charactersMPI identifiers are limited to 30 characters (31 with the profiling interface). This is done to avoid exceeding the limit on some compilation systems.

  suggested change (Rolf):

  MPI identifiers are limited to 32 characters (35 with the profiling and large count interfaces, and Fortran specific names up to 41). This is done to avoid exceeding the limit on some compilation systems.

  why do we need this change:

  new constants:

  34 = MPI_T_CB_REQUIRE_ASYNC_SIGNAL_SAFE

  31 = MPI_T_CB_REQUIRE_MPI_RESTRICTED

  new (1 old) callback prototypes:

  31 = MPI_Session_errhandler_function

  31 = MPI_Datarep_conversion_function

  33 = MPI_Datarep_conversion_function_c

  needs to be truncated - 34 --> 31 = MPI_T_CB_REQUIRE_ASYNC_SIGNAL_SAFE

- 11:12-18 **MPI functions specification** **--> maybe mention Big Count here ?**

  All MPI functions are first specified in the language-independent notation. Immediately below this, language dependent bindings follow:

  * The ISO C version of the function (first using int as the type for byte displacement and count arguments and second using MPI_Aint and MPI_Count via separate "_c" suffixed procedures).
  * The Fortran version used with USE mpi_f08 (first using INTEGER as the type for byte displacement and count arguments and second using INTEGER(KIND=MPI_ADDRESS_KIND) and INTEGER(KIND=MPI_COUNT_KIND) via polymorphic interfaces, see Section 19.2 for a full explanation).
  * The Fortran version … with USE mpi or INCLUDE 'mpif.h' (only using INTEGER as the type for byte displacement and count arguments).

→ this needs to be fixed / word smithed by the Big Count WG !

Chapter2: MPI Terms and Conventions OPEN ISSUES – MAYBE – #.....

- 21:40 **Hard to read/understand by people not familiar with Fortran --> maybe include ?**
  With the mpi_f08 module, two new Fortran features, *assumed type* TYPE(*) and *assumed rank* DIMENSION(..), are also required, see Section 2.5.5.

- 22:13 **Misleading (was true only for MPI-3.0) --> write in a clearer way ?**
  Some of the previously deprecated constructs are ~~now~~ removed, as documented in Chapter 17.
  OR
  Some of the constructs that were deprecated in MPI-2.0 are removed since MPI-3.0, as …

- 23:26,29 **MPI-3.2** should be substituted by MPI-4.0 (throughout)

- 23:37 formatting: Table caption: why is Removed capitalized ?

- 23:30,34 **Deprecated MPI_SIZEOF in Table 2.1 --> maybe improve ?**

  MPI_SIZEOF MPI-4.0 (deprecated since) storage_size()[5] (Replacement)

  [5] Fortran intrinsic: It returns the size in bits instead of bytes.

  suggested change (Claudia):

  MPI_SIZEOF MPI-4.0 storage_size()[5] or c_sizeof()

  [5] Fortran intrinsic: storage_size() returns the size in bits instead of bytes, see Section 16.4.

  see later in 16.4 Deprecated…:

  773:44-47

  The following Fortran subroutines are deprecated because the Fortran language storage_size() and c_sizeof() intrinsic functions provide similar functionality. Note that while MPI_SIZEOF and c_sizeof() return the size in bytes, storage_size() provides the size in bits.

Chapter2: MPI Terms and Conventions OPEN ISSUES – formatting, …

- 12:6-7 **MPI Operations – stages:**

  Freeing returns control of the rest of the argument list (e.g., the data buffer address and array arguments) to the application.
- 12:8-9 formatting --> move left: MPI operations are available…
- 12:40 formatting --> move left: Additionally, an MPI operation can be collective or…
- 13:12 formatting --> longer dash: … - defined as follows: …
- 12:17/18 Figure caption: Do not capitalize all words, only first
- 12:30 Figure caption: Do not capitalize all words, only first
- 13:7/8 Figure caption: Do not capitalize all words, only first
- 15:19 For MPI datatypes, the following terms are defined:

Chapter2: MPI Terms and Conventions OPEN ISSUES – formatting, …

- 15:30 bad line break, seems to fit in line (maybe remove or move “\flushline”)
- 16:19 too much white space (vertical)
- 18/19 bad page break (forced)
- 19:20 formatting --> move left: The constants… (seems okay, but hard to read that way)
- 20:27 no blank (~) in INTEGER~(KIND=…
- 21:5 (x2) no blank (~) in INTEGER~(KIND=…
- 21:13/14 bad line break (forced ?)
- 21:16 no blank (~) in INTEGER~(KIND=…
- 21:21 no blank (~) in INTEGER~(KIND=…
- 22:8 & 39 do we really want to write: INTEGERs
- 22:33-34 different macros/sizes for: COMM_COPY_ATTR_FUNCTION and MPI_NULL_COPY_FN
- **#335** **https://github.com/mpi-forum/mpi-issues/issues/335**