2. If the MPI implementation raises an error related to process failure to the root process of MPI_COMM_SPAWN, no spawned processes should be able to communicate on the created intercommunicator.

Advice to users. As with communicator creation functions, it is possible that if a failure happens during dynamic process management operations, an error might be raised at some processes while others succeed and obtain a new communicator. (*End of advice to users.*)

17.2.4 One-Sided Communication

ket04172013.

One-sided communication operations must provide failure notification in their synchronization operations which may raise MPI_ERR_PROC_FAILED (see Section 17.2). If the implementation raises an error related to process failure from the synchronization function, the epoch behavior is unchanged from the definitions in Section 11.4. As with collective operations over MPI communicators, it is possible that some processes have detected a failure and raised MPI_ERR_PROC_FAILED, while others returned MPI_SUCCESS.

Unless specified below, the state of memory targeted by any process in an epoch in ket04172013. which operations raised an error related to process failure is undefined, with the exception of memory targeted by remote read operations (and operations which are semantically equivalent to read operations). All other window locations are valid.

1. If a failure is to be reported during active target communication functions MPI_WIN_COMPLETE or MPI_WIN_WAIT (or the non-blocking equivalent MPI_WIN_TEST), the epoch is considered completed and all operations not involving the failed processes must complete successfully.

2. If the target rank has failed, MPI_WIN_LOCK and MPI_WIN_UNLOCK operations ket04172013. raise an error of class MPI_ERR_PROC_FAILED. ket04172013. [If the owner of a lock has failed,]The lock cannot be acquired again, at any target in the window, and all subsequent operations on the lock must raise MPI_ERR_PROC_FAILED.

ket04172013.

After a process failure has been detected, MPI_WIN_FREE, as with all other collective operations, may not complete successfully on all ranks. For any rank which receives the return code MPI_SUCCESS, the behavior is defined as in Section 11.2.1. If a rank receives a return code related to process failure, the implementation makes no guarantee about the success or failure of the MPI_WIN_FREE operation remotely, though it should still attempt to clean up any local data used by the Window object. This will be signified by returning MPI_WIN_NULL when the object has been freed locally.

It is possible that request-based RMA operations complete successfully while the enclosing epoch completes by raising error due to process failure. In this scenario, the local buffer is valid but the remote targeted memory is undefined.

17.2.5 I/O

I/O error classes and their consequences are defined in Section 13.7. The following section defines the behavior of I/O operations when MPI process failures prevent their successful completion.