Proposed API in support Fault Tolerance in MPI

MPI-3 Fault Tolerance Working Group

April 21, 2009

April 21, 2009

0.1 Introduction

Here is the list of assumptions used in creating the specification for Fault Tolerance Support in MPI:

- Existing MPI codes will run unmodified in the presence of MPI support for Fault tolerance
- Applications may choose to continue running if the failure does not impact MPI's ability to satisfy application MPI requests
- An application my choose not to know about failures that do not directly affect a given process. Here the dangers are similar to posting a pair of processes each posting a blocking send - a failure in a chained set of communications could cause a process to wait indefinitely on a message that may never arrive.
- Errors are specifically associated with specific call sites.
- An application may choose to be notified when an error occurs somewhere in the system.
- Applications not using collective operations do not require collective recovery.

0.2 Initializing Fault Tolerance Support in MPI

The communicator attributes mechanism will be used to specify predefined attributes that are used to manage the communicator recovery process. Specifying these at communicator construction would be preferred, but this would require changing the communicator construction API. MPI_COMM_SET_NAME() is used to attach a name to a communicator.

The list of supported parameters include:

MPI_COMMUNICATOR_RECOVER Defines if MPI should attempt to restart the failed communicator. Possible values:

- MPI_NO_RECOVER (default)
- MPI_RECOVER

MPI_CRITICAL_PROCS This includes the list of ranks that if failed, no recovery should be attempted. The first item in the array is the number of entries, followed by the list of ranks.

MPI_RECOVERY_THRESHOLD_PERCENT This parameter defines the minimum size of the recovered communicator, as a percent of the original communicator size. If MPI is unable to restore this minimal count, an error should be returned.

MPI_RECOVERY_THRESHOLD_COUNT This parameter defines the minimum size of the recovered communicator, as a count. If MPI is unable to restore this minimal count, an error should be returned.

MPI_PROC_RESTORATION_POLICY The policy to be used in restoring failed processes.

- MPI_RESTORE_ALL - restore all failed processes (default)
- MPI_RESTORE_SOME - restore as many failed processes as possible
- MPI_RESTORE_NONE - do not restore any failed processes

MPI_ERROR_REPORTING_FN User defined function to be called before returning from an MPI call used by the application to save error data.

MPI_RECOVERY_FN User defined function to be called by MPI right before the communicator recovery function returns. This provides the caller an opportunity to run user-defined code as part of the recovery process.

MPI_COLL_RECOVERY_FN User defined function to be called by MPI right before the communicator collective recovery function returns. This provides the caller an opportunity to run user-defined code as part of the recovery process.

MPI_GLOBAL_ERROR_NOTIFICATION Specifies if all processes in the communicator should be notified when any process in the communicator fails.

- MPI_LOCAL_NOTIFICATION - notify only processes directly impacted by the failure.
- MPI_GLOBAL_NOTIFICATION - notify all processes on failure.

MPI_DISCARD_PENDING_MESSAGES Specifies what to do with outstanding communication when process failure occurs.

- MPI_DISCARD_FAILED_PROCS - discard only traffic associated with the failed process
- MPI_DISCARD_ALL - discard all traffic associated with the communicator

0.3 Restoring MPI Processes

MPI_RESTORED_PROCESS(generation, return_code)

OUT	generation	Process generation (integer)
OUT	return_code	return error code (integer)

This function is used to figure out what generation the current process is for the local MPI process. Each MPI process starts at generation zero. The return value for generation is a local value, strictly with local meaning.

MPI_GET_LOST_COMMUNICATORS(comm_names, count, return_code)

OUT	comm_names	Array of communicators that may be restored (strings)
OUT	count	Number of Communicators that may be restored (integer)
OUT	return_code	return error code(integer)

This function returns a list of communicators that the application may choose to restore. The strings provided are those set on the communicator with MPI_COMM_SET_NAME. The library will, by default, restore MPI_COMM_WORLD within the local view as well as MPI_COMM_SELF and MPI_COMM_NULL. The (user defined) strings returned by the routine may be used to rejoin the failed communicators.

MPI_COMM_REJOIN(comm_names, comm, return_code)

IN	comm_names	Communicator name (string)
OUT	comm	communicator (handle)
OUT	return_code	return error code(integer)

This function rejoins the local rank to the specified communicator, with local recovery properties. When the call returns, the communicator may be used for point-to-point communications.

0.4 Communicator Restoration

```
recover_rank {
    comm communicator
    int rank
}


recovery_result {
    comm communicator
    int rank
    int result
}
```

MPI_COMM_IRECOVER(ranks_to_restore, request, return_code)

IN	ranks_to_restore	array of ranks to restore (struct)
OUT	request	request object (handle)
OUT	return_code	return error code(integer)

This routine issues a non-blocking request to restored a list of processes. It is the responsibility of the MPI implementation to ensure that only a single instance of a given

process exists at a given point in time. It must ensure that requests to restart a healthy process or multiple requests to restart the same process do not result in the MPI implementation getting into an internally inconsistent state. This routine is called by a surviving process that detects process failure, and is strictly local in nature. It restores local communications (point-to-point, one-sided, data-type creation, etc), but not collective communications.

note !!! Need to update the status object, so it can be interrogated for the results of the process recovery operation.

MPI_COMM_RECOVER(ranks_to_restore, result, return_code)

IN	ranks_to_restore	array of ranks to restore (struct)
OUT	result	array of recovery results (struct)
OUT	return_code	return error code(integer)

This routine is the blocking version of the process recovery function.

MPI_COMM_IRECOVER_COLLECTIVE(ranks_to_restore, request, return_code)

IN	ranks_to_restore	array of ranks to restore (struct)
OUT	request	request object (handle)
OUT	return_code	return error code(integer)

This routine initiates asynchronous collective communicator recovery. All ranks (surviving and restored) in the recovered communicator must participate in this recovery by making a call to this function. If no process are to be restored, a single entry with rank MPI_COMM_NULL must be specified, with the union of the requests made by all ranks specifying the list of processes that will be restored.

MPI_COMM_RECOVER_COLLECTIVE(ranks_to_restore, request, return_code)

IN	ranks_to_restore	array of ranks to restore (struct)
OUT	request	request object (handle)
OUT	return_code	return error code(integer)

This routine initiates the synchronous collective communication recovery process. Since this is a blocking collective calls, callers must ensure correct call ordering in each rank to avoid deadlock.

0.5 Check Communictor State

MPI_COMM_VALIDATE(comm, failed_process_count, failed_ranks, return_code)

IN	comm	communicator (handle)
OUT	failed_process_count	number of failed ranks in communicator (integer)
OUT	failed_ranks	array of failed ranks (integer)
OUT	return_code	return error code(integer)

This blocking routine is used to check the state of the communicator, and is a collective call. The implementation must be cognizant of the fact due to process failure, not all ranks may be able to call this routine due to process failure, and still complete in a bounded amount of time. The MPI implementation must distinguish between process failure and late arrival of some ranks, due to caller timing issues.

MPI_COMM_IVALIDATE(comm, request, return_code)

IN	comm	communicator (handle)
OUT	request	request (handle)
OUT	return_code	return error code(integer)

Asynchronous version of he communicator validation routine.
Note !!! Need to update the status object for returned information.

0.6 Call back functions

void(*MPI_COMM_ERROR_REPORT_FN) (comm, error_code, data)

IN	comm	communicator (handle)
IN	parameter	error_code (integer)
IN	data	error description (void *)

For backward compatibility, the return code from MPI functions must remain an integer, rather than a structure containing the error description. This function will be called within the context of the caller, letting the caller manage the returned error description like it would manage data from a returned in an error data structure. The list of added error codes includes:

MPI_ERROR_RECOVERED As part of the recovery procedure, the library will invoke the local recovery function set by the MPI application at communicator creation. Only process local work, MPI and other, will be done within this user defined recovery function. In addition, the MPI library will discard any outstanding communication with the failed process, and reinitialize communications with the newly

restored ranks. `MPI_Wait()` and `MPI_Test()` calls made on `MPI_Request` objects associated with the restored process and that were initialized before recovery will return `MPI_ERROR_RECOVERED`, with the request object reset to `MPI_REQUEST_NULL`.

MPI_ERROR_PROC_FAILED This error code indicates that a process failure has been detected. The returned error information includes the communicator and rank information. If error notification is requested only at the affected call sites, this will return only the ranks associated with the communicator being used. With global error notification, information will be returned for all communicators in use by the given rank. Each failed process will be reported with all communicators in use, giving it's appropriate rank within each such communicator.

void(*MPI_COMM_RECOVERY_FN) (comm)

IN	comm	communicator (handle)

This callback function is invoked by theMPI library right before the library returns from the local repair function `MPI_COMM_RECOVER()` or `MPI_COMM_IRECOVER()`. This provides the application with a way to invoke communicator specific code on recovery. The intent is intended as support for layered library recovery.

void(*MPI_COMM_COLLECTIVE_RECOVERY_FN) (comm)

IN	comm	communicator (handle)

This callback function is invoked by theMPI library right before the library returns from the collective repair function `MPI_COMM_RECOVER_COLLECTIVE()` or `MPI_COMM_IRECOVER_COLLECTIVE()`. This provides the application with a way to invoke communicator specific code on recovery. The intent is intended as support for layered library recovery.

Index