```
1
               MPI_IRSEND(buf, count, datatype, dest, tag, comm, request)
          2
                           buf
                 IN
                                                        initial address of send buffer (choice)
          3
                 IN
                                                        number of elements in send buffer (non-negative inte-
                           count
          4
                                                        ger)
          5
          6
                 IN
                           datatype
                                                        datatype of each send buffer element (handle)
                 IN
                           dest
                                                        rank of destination (integer)
                 IN
                           tag
                                                        message tag (integer)
          9
         10
                 IN
                                                        communicator (handle)
                           comm
         11
                 OUT
                                                        communication request (handle)
                           request
         12
         13
               int MPI_Irsend(void* buf, int count, MPI_Datatype datatype, int dest,
         14
                               int tag, MPI_Comm comm, MPI_Request *request)
         15
         16
               MPI_IRSEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR)
         17
                    <type> BUF(*)
         18
                    INTEGER COUNT, DATATYPE, DEST, TAG, COMM, REQUEST, IERROR
ticket150. 19
               {MPI::Request MPI::Comm::Irsend(const void* buf, int count, const
ticket150. 21
                               MPI::Datatype& datatype, int dest, int tag) const (binding
                               deprecated, see Section 15.2) }
         23
                   Start a ready mode nonblocking send.
         24
         25
         26
               MPI_IRECV (buf, count, datatype, source, tag, comm, request)
         27
                 OUT
                           buf
                                                        initial address of receive buffer (choice)
         28
                 IN
                           count
                                                        number of elements in receive buffer (non-negative in-
         29
                                                        teger)
         30
         31
                 IN
                           datatype
                                                        datatype of each receive buffer element (handle)
 ticket51.
                 IN
                           source
                                                        rank of source or MPI_ANY_SOURCE (integer)
ticket
51. _{34}
                 IN
                                                        message tag or MPI_ANY_TAG (integer)
                           tag
                 IN
                           comm
                                                        communicator (handle)
         35
         36
                 OUT
                           request
                                                        communication request (handle)
         37
         38
               int MPI_Irecv(void* buf, int count, MPI_Datatype datatype, int source,
         39
                               int tag, MPI_Comm comm, MPI_Request *request)
         40
               MPI_IRECV(BUF, COUNT, DATATYPE, SOURCE, TAG, COMM, REQUEST, IERROR)
         41
         42
                    <type> BUF(*)
                    INTEGER COUNT, DATATYPE, SOURCE, TAG, COMM, REQUEST, IERROR
ticket150. 44
               {MPI::Request MPI::Comm::Irecv(void* buf, int count, const
ticket150.
                               MPI::Datatype& datatype, int source, int tag) const (binding
                               deprecated, see Section 15.2) }
         47
         48
                   Start a nonblocking receive.
```