MAINTENANCE NOTES

X♦IPC RELEASE 3.3.0.a

ENHANCEMENTS AND BUG FIXES

QueSys

- **QueBurstSend()**

  In prior versions, an application doing a QueBurstSend() to a network instance would terminate abnormally if the instance was brought down. This has been corrected and the QueBurstSend() call now returns the error XIPC_ER_SYSERR.

- **Queue Spooling**

  In prior versions, if spooling had been enabled on multiple queues and there was heavy activity (multiple senders/receivers on multiple queues), the QueSys verbs - QueSend(), QueReceive(), QueGet(), QuePut() could cause, in rare occasions, the application to exit with a fatal error. This has been fixed in the current release.

- **Queview**

  In prior versions, if the MAX_NODES parameter in the [QUESYS] section in the configuration file was set to greater than 10000, the queview display would be garbled. This has been corrected.

MomSys

- **SINGLEWRITE_TEXT_SIZE**

  A new parameter has been added to the [MOMSYS] section of the configuration file. The SINGLE_WRITE_TEXT_SIZE_MRI and SINGLEWRITE_TEXT_SIZE_MRO parameters indicate that messages that are shorter than or equal to the specified size are written to the message repository along with the message header in a single write operation. When configured correctly, this feature considerable increases the message throughput of MomSys. Typically, you would want to set the parameters to a value that will allow most of the messages to be written in a single operation. The default value for this parameter is set to 128 bytes. This parameter should not be set larger than the PAGESIZE on the system.

X♦IPC

- **XipcLogin() API**

  In prior versions, all threads within a multithreaded application that were doing an XipcLogin() operation would block if one of the threads was blocked on a call to XipcLogin(). This behavior has been corrected.

- **XipcAbort() hangs on UNIX**

  In prior versions, the XipcAbort() call would hang and freeze the entire XIPC instance if the System V IPC Queue limit had been reached on the system. This has been corrected.

- **Asynchronous Completion on UNIX**

  In prior versions, if two or more threads within an application were doing Asynchronous XIPC operations then, in rare occasions, one of the threads would return with an XIPC_ER_SYSERR on an XIPC API call that resulted in an Asynchronous completion of an XIPC operation. The failing system call would be "*msgsnd*" with the error number set to *EINVAL*. This has been corrected.

Windows NT/2000

- **XipcMaxThreads**

  The XipcMaxThreads variable was not exported from the XIPC DLL's. This limited the Windows applications to having a maximum of 64 threads logged into XIPC at any given time. This has been corrected.

Compaq Tru64 UNIX

- **XIPCLAD daemons**

  In prior versions, the XIPCLAD daemon would not route AEB's correctly when an asynchronous operation completed in case of multithreaded applications. This has been corrected.