Envoy XIPC

Version 3.4.0

Platform Notes - WIN32

Envoy Technologies Inc.

555 Route 1 South
Iselin, NJ 08830

http://www.envoytech.com

Printed in United States of America.

TABLE OF CONTENTS

1. PREFACE 2

1.1 Purpose 2

1.2 Contents 2

2. INSTALLATION 3

2.1 Selecting A Target Machine 3

2.2 Installing X•IPC On The Target Machine 3

2.3 Configuring the Platform Environment 3

2.3.1 XIPCROOT 3

2.3.2 Path 3

2.4 Configuring the Network Environment 4

2.5 The X•IPC Service Programs and Background Processes 4

3. USING X•IPC ON WINDOWS 5

3.1 Starting and Stopping the X•IPC Platform Environment 5

3.2 Application Development With X•IPC on Windows 5

3.2.1 Predefined Datatypes 5

3.2.2 Compiling 5

3.2.3 Linking 6

3.2.4 Monitoring 7

3.2.5 Sample Programs 7

3.2.6 Non Thread-Safe Functions 7

1. PREFACE

1.1 Purpose

The primary objective of these Platform Notes is to provide all platform-specific information necessary for working with X•IPC on Win32 platform.

The document is divided into three sections: a preface that outlines the purpose of the Platform Notes; detailed step-by-step instructions to be followed when installing X•IPC on Win32; and platform-specific information necessary for developing applications with the current release of X•IPC on the Win32 platform.

1.2 Contents

The Platform Notes consist of the following parts:

- ❑ *Installation:*
  - ❑ Selecting the machine to install on.
  - ❑ Reading X•IPC from the provided media onto the target machine.
  - ❑ Configuring the network databases for using X•IPC in a network environment.
- ❑ *Using X•IPC on Windows:*
  - ❑ X•IPC service programs: their function and how to use them.
  - ❑ Notes on compiling, linking and other information necessary for developing software applications with X•IPC on Windows.

2. INSTALLATION

2.1 Selecting A Target Machine

Rules for selecting a target machine for the installation revolve around the issues of accessibility and convenience. The product can be installed on any Windows platform covered by the developer license.

2.2 Installing X•IPC On The Target Machine

The X•IPC release media includes all the software, sample programs and other related files necessary for working with the product. Installing X•IPC from the provided media on the target machine should be performed in the following steps:

- □ Log into an account with administrator privileges.
- □ Insert the CD into the appropriate drive and run **d:\setup** (assuming d is the drive letter for your CD).
- □ Follow the installation instructions.

As a result of this process, the X•IPC directory structure will be created and X•IPC software and services will be installed.

The X•IPC installation directory contains the following directories and files:

lib \	a directory containing the X•IPC libraries.
bin \	a directory containing X•IPC executables and DLL's.
pdb\	a directory containing the platform database files used for data translation.
include\	a directory containing X•IPC header files.
samples \	a directory containing X•IPC sample programs.
log\	a directory for X•IPC log files.
xipc.env	a sample environment file for X•IPC.

2.3 Configuring the Platform Environment

2.3.1 XIPCROOT

XIPCROOT is a system environment variable which must be set to the path of the platform directory in which X•IPC was installed before invoking any X•IPC command or API call.

The installation process automatically sets up XIPCROOT as a system environment variable on the Windows platform.

2.3.2 PATH

It is a good idea to add the X•IPC programs directory to the path so that the programs can be easily invoked. The programs directory is the \bin directory within the X•IPC installation directory.

2.4 Configuring the Network Environment

Installing X•IPC requires that the network be notified of X•IPC's intention to use certain TCP/IP services. This is accomplished by adding three entries within the TCP `services` file. On Windows NT, this is typically found in the `c:\winnt\system32\drivers\etc` directory; on Windows 98, it is typically found in the `c:\windows` directory. The port numbers in these entries should be selected so that they relate to unused port numbers.

Example:

```
xipcetc      4000/udp
xipcetc      4000/tcp
xipcserv     4001/tcp
```

These entries *must* be added and *must* be identical on *each* of the platforms using X•IPC in a network environment. X•IPC will not work properly on platforms where these entries have not been added.

If some form of network directory service facility is being used for finding network services, the above indicated changes should be applied to that directory service instead of the `services` file.

Note: In certain situations, it is required by Win32 that an extra blank line be appended to the end of the `services` file.

2.5 The X•IPC Service Programs and Background Processes

X•IPC installs with its own service program, `xipcownd`. Additional background processes–`xipcisd, xipcicd, xipcidld` and `xipciad`–are also started by `xipcinit`.

Service Program	**Function**
`xipcownd`	X•IPC's Object service

Background Processes	**Function**
`xipcisd`	X•IPC's TCP/IP server
`xipciad`	X•IPC's TCP/IP asynchronous handler
`xipcicd`	X•IPC's TCP/IP catalog server
`xipcidld`	X•IPC's Idlewatch daemon

The `xipcownd` service program handles internal XIPC objects for instances on the machine. `xipcownd` reports any errors in the file `%XIPCROOT%\log\xipcownd.log`.

The `xipcisd` background process is the TCP/IP Server program that handles remote `XipcLogin()` and `XipcList()` requests. `xipcisd` reports any errors in the file `%XIPCROOT%\log\xipcisd.log`.

The `xipciad` background process is used by any platform employing X•IPC asynchronous functionality in a network environment; `XipcPing()`; `XipcAbort()`; and MomSys, in a local or network instance. `Xipciad` reports any errors in the file `%XIPCROOT%\log\xipciad.log`.

The `xipcicd` background process is used by MomSys in order to access the X•IPC catalog. `Xipcicd` reports any errors in the file `%XIPCROOT%\log\xipcicd.log`.

The `xipcidld` background process is used whenever an X•IPC instance needs to be monitored against idle users. `Xipcidld` reports any errors in the file `%XIPCROOT%\log\xipcidld.log`. Refer to the X•IPC User Guide Appendix for the Technical Note on the Idle User Detection Mechanism for a complete description of this service.

3. USING X•IPC ON WINDOWS

The X•IPC paradigm is not specific to any particular operating system environment. From the programmer's perspective the model presented by X•IPC's API is almost entirely portable across environments. The underlying issues of how X•IPC relates to a particular platform and how it utilizes the native operating system resources are, however, important for understanding how to *best* use the product on that platform.

3.1 Starting and Stopping the X•IPC Platform Environment

The `xipcinit` and `xipcterm` commands must be run to, respectively, start and stop the X•IPC platform environment on the Windows platform. The `xipcownd` service is installed as an automatic service which is always running and is therefore not affected by `xipcinit` and `xipcterm`. On Windows 98, the `xipcownd` is automatically placed in the startup folder during the installation process.

Examples:

```
%XIPCROOT%\bin\xipcinit

%XIPCROOT%\bin\xipcterm
```

3.2 Application Development With X•IPC on Windows

Divergent methods of program development, specific to each operating system environment, affect how X•IPC is used in that environment, most notably in the areas of compiling and linking. These topics are examined in this section.

3.2.1 PREDEFINED DATATYPES

Much of X•IPC's documentation refers to predefined datatypes such as `XINT, CHAR,` etc. The mapping between these types and the underlying "C" language datatypes is machine dependent. For the Windows platform, type `XINT` is defined as a 32 bit signed integer and `CHAR` is defined as `char`.

It is recommended that programs making X•IPC function calls use these definitions for declaring parameter variables that are passed to the X•IPC functions. This will ensure portability across different hardware platforms. The definitions are in the file `mmcos.h` and are included automatically by any program that includes `xipc.h.`

3.2.2 COMPILING

X•IPC currently comes with a "C" language binding. Other languages may also be used to invoke X•IPC. This may require preparation of function prototypes and data type definitions for that language.

When compiling a "C" program using X•IPC, the header files directory should be made known to the compiler by specifying it in the `/I` compiler option.

Please refer to the sample makefile that is provided in `%XIPCROOT%\samples\src`.

3.2.2.1 An X•IPC Example

The following example is for the Microsoft Visual C++ compiler:

```
CL /c /MD /DWIN32 /I. /I c:\msvcnt\include /I %XIPCROOT%\include foo.c
```

3.2.3 LINKING

The X•IPC API library comes in three flavors, each of which addresses a specific class of application. The three libraries are:

- The X•IPC Stand Alone Library
- The X•IPC Network Library
- The X•IPC Combined Library

Refer to the "Using X•IPCLibraries" section of the X•IPC User Guide for a detailed discussion of when each library is appropriate.

This section presents technical instructions for using the X•IPC libraries to develop applications on the Windows platform.

3.2.3.1 The X•IPC Libraries

The following libraries are included in the `lib` directory:

sxipc.lib	The X•IPC Stand Alone Library

If you have installed the network version of the toolset, the following libraries are also included:

nxipc.lib	The X•IPC Network Library
xipc.lib	The X•IPC Combined Library

Programs that link with the Network or Combined API Library need to additionally be linked with the following protocol support library:

nxipctcp.lib	The X•IPC TCP/IP protocol support library

3.2.3.2 The X•IPC Dynamic Link Libraries (DLL)

The following dynamic link libraries are included in the `bin` directory:

xipc.dll	The X•IPC Combined DLL
sxipc.dll	The X•IPC Stand-Alone DLL
nxipc.dll	The X•IPC Network DLL
nxipctcp.dll	The X•IPC TCP/IP Protocol Support DLL

3.2.3.3 Linking With The X•IPC Stand-Alone Library

Linking programs that use the X•IPC Stand-Alone Library is performed as follows:

Example:

```
link foo.obj/out:foo.exe /LIBPATH:%XIPCROOT%\lib sxipc.lib advapi32.lib
     netapi32.lib
```

3.2.3.4 Linking With The X♦IPC Network Library

Linking programs that use the X♦IPC Network Library is performed as follows:

Example:

```
link foo.obj/out:foo.exe nxipc.lib nxipctcp.lib wsock32.lib
     advapi32.lib netapi32.lib
```

3.2.3.5 Linking With The X♦IPC Combined Library

Linking programs that use the X♦IPC Combined Library is performed as follows:

```
link foo.obj/out:foo.exe xipc.lib nxipctcp.lib wsock32.lib
     advapi32.lib netapi32.lib
```

3.2.4 MONITORING

All of X♦IPC's monitors (i.e., `momview`, `queview`, `memview` and `semview`) are supported by Windows NT/98. However, the trace step and trace flow update options are not implemented in this environment. (See the X♦IPC User Manual, Section 5.3.3, for general subsystem monitor information.)

3.2.5 SAMPLE PROGRAMS

A number of sample programs and makefiles are included with the X♦IPC product. They are installed under the `%XIPCROOT%\samples` directory.

3.2.6 Non Thread-Safe Functions

The following functions are not thread-safe: QueList(), SemList(), MemList() and MemSection().

Applications using multiple threads should use:

❑ Instead of QueList(), use QueListBuild() or QueListAdd().

❑ Instead of SemList(), use SemListBuild() or SemListAdd().

❑ Instead of MemList(), use MemListBuild() or MemListAdd().

❑ Instead of MemSection(), use MemSectionBuild().