TIBCO SmartSockets™

Installation Guide

Software Release 6.8
July 2006

Important Information

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE *TIBCO SMARTSOCKETS INSTALLATION GUIDE*). USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

TIB, TIBCO, Information Bus, The Power of Now, TIBCO Adapter, RTclient, RTserver, RTworks, SmartSockets, and Talarian are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and / or other countries.

EJB, J2EE, JMS and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Please see the readme.txt file for the availability of this software version on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND / OR CHANGES IN THE PRODUCT(S) AND / OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

Contents

Preface

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Please see the readme.txt file for the availability of this software version on a specific operating system platform.

TIBCO SmartSockets™ is a message-oriented middleware product that enables programs to communicate quickly, reliably, and securely across:

- local area networks (LANs)
- wide area networks (WANs)
- the Internet

TIBCO SmartSockets takes care of network interfaces, guarantees delivery of messages, handles communications protocols, and directs recovery after system or network problems. This enables you to focus on higher-level requirements rather than the underlying complexities of the network.

This reference provides the detailed information you need to understand and use the TIBCO SmartSockets system. Before using this reference, it is helpful to read the *TIBCO SmartSockets Tutorial* and work through the lessons in that book. The *TIBCO SmartSockets Tutorial* begins at a more basic level and is an excellent introduction to SmartSockets.

For an overview of the new features, changes, and enhancements in this Software Release 6.8, see Chapter 1, Overview of TIBCO SmartSockets.

Topics

Related Documentation

This section lists documentation resources you may find useful.

TIBCO Product Documentation

The following documents form the TIBCO SmartSockets documentation set:

- *TIBCO SmartSockets API Quick Reference*
- *TIBCO SmartSockets Application Programming Interface*
- *TIBCO SmartSockets C++ User's Guide*
- *TIBCO SmartSockets cxxipc Class Library*
- *TIBCO SmartSockets Installation Guide*
- *TIBCO SmartSockets Java Library User's Guide and Tutorial*
- *TIBCO SmartSockets .NET User's Guide and Tutorial*
- *TIBCO SmartSockets Tutorial*
- *TIBCO SmartSockets User's Guide*
- *TIBCO SmartSockets Utilities*
- *TIBCO SmartSockets C++ and Java Class Libraries*

  C++ class library and Java application programming interface (API) reference materials are available in HTML format only. Access the references through the TIBCO HTML documentation interface.

Using the Online Documentation

The SmartSockets documentation files are available for you to download separately, or you can request a copy of the TIBCO Documentation CD.

Conventions Used in This Manual

This manual uses the following conventions.

Typeface Conventions

This manual uses the following typeface conventions

Example	Use
`monospace`	This monospace font is used for program output and code example listing and for file names, commands, configuration file parameters, and literal programming elements in running text.
`monospace bold`	This bold monospace font indicates characters in a command line that you must type exactly as shown. This font is also used for emphasis in code examples.
Italic	Italic text is used as follows: • In code examples, file names etc., for text that should be replaced with an actual value. For example: "Select *install-dir* / `runexample.bat`." • For document titles. • For emphasis.
Bold	Bold text indicates actions you take when using a GUI, for example, click **OK**, or choose **Edit** from the menu. It is intended to help you skim through procedures when you are familiar with them and just want a reminder. Submenus and options of a menu item are indicated with an angle bracket, for example, **Menu > Submenu.**
	Warning. The accompanying text describes a condition that severely affects the functioning of the software.
	Note. Be sure you read the accompanying text for important information.
	Tip. The accompanying text may be especially helpful.

Notational Conventions

The notational conventions in the table below are used for describing command syntax. When used in this context, do not type the brackets listed in the table as part of a command line.

Notation	Description	Use
[ ]	Brackets	Used to enclose an optional item in the command syntax.
< >	Angle Brackets	Used to enclose a name (usually in *Italic*) that represents an argument for which you substitute a value when you use the command. This convention is not used for XML or HTML examples or other situations where the angle brackets are part of the code.
{ }	Curly Brackets	Used to enclose two or more items among which you can choose only one at a time. Vertical bars (|) separate the choices within the curly brackets.
...	Ellipsis	Indicates that you can repeat an item any number of times in the command line.

Identifiers

The term identifier is used to refer to a valid character string that names entities created in a SmartSockets application. The string starts with an underscore (_) or alphabetic character and is followed by zero or more letters, digits, percent signs (%), or underscores. No other special characters are valid. The maximum length of the string is 63 characters. Identifiers are not case-sensitive.

These are examples of valid identifiers:

```
EPS
battery_11
K11
__
_all
```

These are invalid identifiers:

```
20
battery-11
@com
$amount
```

Case

Function names are case-sensitive, and must use the mixed-case format you see in the text. For example, TipcMsgCreate, TipcSrvStop, and TipcMonClientMsgTrafficPoll are SmartSockets functions and must use the case as shown.

Monitoring messages are also case-sensitive, and should be all upper case, such as T_MT_MON_SERVER_NAMES_POLL_CALL. This makes it easy to distinguish them from option or function names.

Although option names are not case-sensitive, they are usually presented in text with mixed case, to help distinguish them from commands or other items. For example, Server_Names, Unique_Subject, and Project are all SmartSockets options.

Identifiers used with the products in the SmartSockets family are not case-sensitive. For example, the identifiers `thermal` and `THERMAL` are equivalent in all processes.

In UNIX, shell commands and filenames are case-sensitive, though they might not be in other operating systems, such as Windows. To make it easier to port applications between operating systems, always specify filenames in lower case.

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, please contact TIBCO Support as follows.

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

  http://www.tibco.com/services/support

- If you already have a valid maintenance or support contract, visit this site:

  http://support.tibco.com

  Entry to this site requires a user name and password. If you do not have a user name, you can request one.

Chapter 1 Overview of TIBCO SmartSockets

This chapter covers the information specific to this Software Release 6.8 and history from previous releases.

Topics

Introduction to TIBCO SmartSockets

is an interprocess messaging software product that enables processes to communicate quickly, reliably, and securely across different operating system platforms. The communicating processes can reside on the same machine, on LANs, on WANs, or anywhere on the Internet.

A SmartSockets message is a structured packet of information that is transferred between two or more programs, which might or might not reside on the same machine. It is not unusual for a SmartSockets message to exist only in memory and never be written to disk. A message is a mechanism that enables program-to-program communication to occur in a manner easily understood by both you and the programs.

SmartSockets is an industrial-strength package that takes care of network interfaces, guarantees delivery of messages, handles communication protocols, and deals with recovery after system or network failures. The SmartSockets' programming model is built specifically to offer high-speed interprocess communication, scalability, reliability, and fault tolerance. It supports a variety of communication paradigms, including publish-subscribe, peer-to-peer, and request-reply. Included as part of the package are graphical tools for monitoring and debugging your distributed applications.

The SmartSockets product consists of a suite of programming interfaces and class libraries, ready-to-run programs, source code for sample programs, and extensive documentation. It is designed to get your network programs running as quickly as possible.

These components are part of a standard SmartSockets release, and are installed during the SmartSockets installation process:

- SmartSockets Application Programming Interface (API) provides a C-callable library of functions for communicating between programs and monitoring your distributed applications.
- SmartSockets Java class library provides classes, objects, and interfaces to Java applications, allowing them to leverage the functions of the SmartSockets API.
- Two SmartSockets C++ class libraries provide your choice of an object-oriented layer on top of the standard SmartSockets services.
- SmartSockets on the Windows platform provides an API for .NET.
- RTserver, a powerful software message router, empowers applications with the publish-subscribe communications model.

- RTmon, a powerful tool for monitoring and debugging your distributed project, is accessible through a graphical user interface (GUI) and through a command-line interface.
- Structured message types, messages with predefined field types, enable transparent data conversion. SmartSockets comes with many predefined message types to get you working quickly. You can easily extend these by defining your own types.
- Options and command language allow you to reconfigure your SmartSockets applications easily.
- Sample programs get you off to a fast start.
- Books and online documentation are available. The complete SmartSockets library is available for download and on CDROM, in HTML and PDF formats. You can order printed books through your sales representative or from TIBCO Product Support.

Extending TIBCO SmartSockets

In addition to the product, there are several other products that work with SmartSockets to expand its capabilities:

- RTgateway

  The TIBCO SmartSockets RTgateway is designed to perform intelligent message switching between diverse message sources, such as RTserver clouds, databases, HTTP requests, or other messaging systems. You can connect several RTserver clouds, increasing the scalability of your SmartSockets system, or you can use it to segment which RTservers receive which messages based on subject or message type, perhaps for security or performance reasons.

  For more information, see the *TIBCO SmartSockets RTgateway User's Guide.*

- TIBCO SmartSockets Cache

  TIBCO SmartSockets Cache acts as a standalone RTclient that can be configured to cache messages by subject. On your SmartSockets system, it is often necessary to have newly joining clients retrieve messages that have already been sent. These new clients can retrieve the messages from the cache by sending a request message to Cache and specifying the subject they are interested in. Cache sends the clients copies of the last few messages that were

sent with a particular subject. This allows the new clients to catch up on messages, based on subject, that are important to them.

For more information, see the *TIBCO SmartSockets Cache User's Guide.*

- TIBCO SmartSockets Rendezvous Adapter

  The TIBCO SmartSockets Rendezvous Adapter is used with the TIBCO SmartSockets RTgateway to perform intelligent message switching between TIBCO SmartSockets and TIBCO Rendezvous.

  For more information, see the *TIBCO SmartSockets Rendezvous Adapter User's Guide*.

For more information on what these SmartSockets products can do for you, contact your TIBCO sales representative. Check the TIBCO website, `http://www.tibco.com`, for information on features and support.

Advanced Features

SmartSockets includes advanced features that must be ordered separately, and enabled in your SmartSockets license. For more information about these features, contact your TIBCO sales representative.

These features were introduced previously, and are available with this release:

- TIBCO SmartSockets Multicast

  The TIBCO SmartSockets Multicast feature improves the features and performance of SmartSockets by implementing reliable multicast, taking advantage of its bandwidth optimization properties. SmartSockets Multicast is an efficient way of routing a message to multiple recipients. This feature enables messages to be multicast to RTclients. SmartSockets Multicast defines the protocol called PGM to route messages and uses RTgms to handle the message routing.

  In the Version 6.2 release, SmartSockets Multicast was improved and enhanced, adding congestion control, bandwidth management, Single Source Mode, and SNMP support. For more information, see the *TIBCO SmartSockets User's Guide.*

The standard SmartSockets license does not include these features. You must contact TIBCO Product Support to purchase any of these features and upgrade your license key.

Migration and Compatibility

This section describes the considerations for migrating to newer releases of SmartSockets or using a mixed environment where not all RTservers or RTclients are at the same version level. This section also covers the changes you need to make to existing SmartSockets applications to convert them to work with newer releases.

Upgrading Your TIBCO SmartSockets Installation

If you currently have only SmartSockets installed, and are upgrading to the newest version, you can simply follow the instructions in the installation chapters, Chapter 3, Installing on UNIX or Chapter 4, Installing on Windows. On UNIX, you must install into a new directory, with the default name being `ss68`. On Windows, you must un-install your current SmartSockets product before you install this new version. By default, the new version is installed into a program folder called `ss68`.

If you have other SmartSockets products also installed, such as SmartSockets RTgateway, and you want to use them with this new SmartSockets Version 6.8 installation, you must re-install them into the new `ss68` directory after you have completed your SmartSockets Version 6.8 install. First, check that the version and revision of your existing products are compatible with SmartSockets Version 6.8. Use the Compatibility Matrix provided with your online documentation or contact TIBCO Product Support to find out. In some cases, you might need to upgrade your other SmartSockets products.

TIBCO SmartSockets RTgateway

Normally, you can install SmartSockets Software Release 6.8 and then install RTgateway into your `ss68` directory, and everything should work correctly. However, if you are building your own MSLs with RTgateway, you must do an extra step.

Building MSLs requires the latest `spss.h` and `spsspr.h` header files included with SmartSockets Software Release 6.8. These are overlaid when you install RTgateway. To avoid this, follow these steps:

1. Install SmartSockets Software Release 6.8 to create the `ss68` directory, and enable you to install RTgateway.
2. Install RTgateway into that `ss68` directory.
3. Uninstall SmartSockets so that you can re-install it. Do not un-install RTgateway.
4. Reinstall SmartSockets Software Release 6.8 into that same `ss68` directory.

That should ensure the latest versions of the `spss.h` and `spsspr.h` header files are installed.

Upgrading C++ Applications

If you are upgrading from version 6.2.x or earlier of the `sscpp` SmartSockets C++ class library, you must recompile and link.

SmartSockets 6.8 includes new monitoring functions that break compatibility. However, the C++ class library remains source compatible. You do not need to make any changes to your application: simply recompile and link.

T_ENTRY Macro and Source Code Portability

For cross-platform portability and to correctly specify function calling conventions under Windows, a T_ENTRY declaration specifier is required in the definition of all callback and thread functions as well as in their prototypes. This is done using the T_ENTRY macro.

Here is a process callback example:

```
void T_ENTRY my_num_data_cb
(
  IPC_CONN conn,
  T_IPC_CONN_PROCESS_CB_DATA data,
  T_CB_ARG arg
)
```

Existing SmartSockets Version 5.0 application source code that does not use the T_ENTRY macro should be updated to do so.

Converting Version 5.x Applications

SmartSockets Versions 5.2 and 5.5 clients are source-compatible with SmartSockets Version 6.8 code. There are no API changes, so no conversion script is needed. However, clients below Version 6.1 cannot handle messages that use a delivery mode of ORDERED because that is a Version 6.1 feature and older clients do not recognize it.

The default behavior of RTclient was changed in Version 5.2 so that it no longer automatically starts RTserver. If existing applications depend on the behavior from Version 5.1 and earlier, the new default behavior can easily be changed by setting the start prefix of the Server_Names option.

The default behavior of RTserver was modified in Version 5.2 so that it automatically connects only to the other RTservers listed in the Server_Names option, not to the RTservers they are connected to. If existing applications depend on the behavior from Version 5.1 and earlier, the new default behavior can be changed by setting the connect prefix of the Server_Names option.

RTserver Compatibility

In SmartSockets Version 6.0 Revision 2, the way RTservers support GMD messages changed. These changes improve GMD reliability when all RTservers involved in a GMD transaction are SmartSockets Version 6.0 Revision 2 or higher. RTservers at a Version 6.0 Revision 2 level work with RTservers at a Version 5.5 level, but the GMD enhancements do not apply and you might experience missed GMD messages during warm disconnects. If you are using GMD, we recommend you upgrade all the RTservers involved in GMD to at least Version 6.0 Revision 2.

If you installed the SmartSockets Version 6.0 Revision 1 limited availability release, we recommend you upgrade all your RTservers to a generally available version, such as Version 6.0 Revision 2.

SmartSockets RTservers Version 6.0 and above are not compatible with RTservers below Version 5.5.

RTservers below Version 6.1 cannot handle messages that use the delivery mode ORDERED.

Chapter 2 System Requirements

This chapter describes the requirements your system must meet before you install TIBCO SmartSockets. Although the information in this chapter is as current and accurate as possible, be sure to check the README file shipped with SmartSockets for the latest corrections and changes. The README file also contains information for any supported operating systems and platforms not included in this reference.

Topics

Platform Requirements

You must install and run SmartSockets on a supported platform. This release of SmartSockets supports Solaris, Windows 2000, XP and 2003, Linux, HP-UX, AIX, IRIS, Tru64, HP Nonstop OSS and OpenVMS.

Additional platforms may be supported by TIBCO Professional Services Group. Contact TIBCO Product Support for more information on the platforms currently supported for any release of SmartSockets.

Sun SPARC Platforms

Any Sun Microsystems SPARC family computer, including all SPARCstations, SPARCservers, and Sun Ultras, is supported if it is running Sun Solaris Version 2.7, 2.8, 2.9 or 2.10.

Hardware must support 64-bit to run the 64-bit version of SmartSockets.

Hewlett-Packard Platforms

Any Hewlett-Packard 9000/7xx and binary-compatible 9000/8xx computer running HP-UX Version 11.11 (or later) is supported.

Hardware must support 64-bit to run the 64-bit version of SmartSockets.

x86 Platforms

Any IBM PC compatible computer equipped with an x86 processor is supported if it is running one of the following operating systems:

- Microsoft Windows XP
- Windows 2000 with Service Pack 1, the Professional or Server editions
- Windows Server 2003
- Sun Solaris 10
- Linux with these components:
  - kernel 2.4.21 and up, or 2.6.x
  - gcc 3.2 and later
  - glibc 2.3.2 and up

Hardware and operating system software must support 64-bit to run the 64-bit version of SmartSockets.

Intel Itanium (IA64) Platforms

Any IA64 platform with either of these operating environments:

- HP-UX 11.11 (or later)
- Linux with these components:
  - kernel 2.4.21 and up
  - gcc 3.2 and later
  - glibc 2.3.2 and up

Tru64 Platforms

Any Hewlett-Packard AlphaServer running Tru64 UNIX 4.0F or higher, or 5.0.

OpenVMS Platforms

Any Hewlett-Packard AlphaServer running OpenVMS 7.3-2 or higher.

AIX Platforms

Any IBM RS/6000 system running AIX 5.1 and up.

64-bit clients are supported only on 5.x. Hardware and operating system software must support 64-bit to run the 64-bit version of SmartSockets.

Hardware must support 64-bit to run the 64-bit version of SmartSockets.

IRIX Platforms

Any SGI IRIS running IRIX 6.5.

The 64-bit version of SmartSockets is not available on IRIX.

HP Nonstop Server

Tandem OSS G06.24 or higher.

Other Platforms

This list of supported platforms is accurate as of the date of the 6.8.0 TIBCO SmartSockets release. Contact TIBCO Product Support for an up-to-date list of supported platforms, or check the TIBCO web site at http://www.tibco.com.

If you purchased SmartSockets on a platform not listed here, look in the README file shipped with your product for information about that platform, its requirements, and any platform-specific installation instructions.

Memory Requirements

256MB of RAM is preferred, but the minimum is 128 MB. To run all the SmartSockets processes on a single system, 256 MB of RAM is the recommended minimum. You must provide adequate swap space for SmartSockets and all other applications. Refer to your operating system manuals for information on configuring swap space.

Java Requirements

To use the SmartSockets Java libraries and APIs, you must have the Java2 Software Developer Kit (J2SDK) version 1.4.2 or later installed on your system. You can download the J2SDK from the Sun Microsystems web site, `http://java.sun.com/j2se`.

UNIX Windows Requirements

To use the RTmon graphical user interface (GUI) on a UNIX system, such as Solaris or HP-UX, you must have the X Window System Version 11 Release 5 (X11R5 or later) and OSF / Motif Version 1.2 installed on your system.

The X Window System is often available in at least two forms. You can acquire and compile the X Consortium source code or use or the vendor implementation, such as Common Desktop Environment. The vendor implementation usually has some proprietary value added to it such as X server support for all graphics hardware, or extras like Display PostScript. Similarly, you can get the OSF / Motif source code from OSF and compile it yourself, or buy binaries from a vendor.

Platform-Specific Notes

This section describes special considerations for SmartSockets running on Hewlett-Packard and Sun SPARC Solaris.

Hewlett-Packard Platforms

When compiling with the Hewlett-Packard aCC compiler, warnings similar to this may occur:

```
Error (future) 229: "trcvr.cpp", line 73 # Ambiguous overloaded
function call; a function match was not found that was strictly
best for ALL arguments.
```

The aCC compiler detects ambiguities when resolving overloaded << and >> operators from the TipcMsg class. At this time, the future error does not prevent your code from compiling correctly. The error can be caused by these circumstances:

- When inserting and extracting fields from messages using the << and >> operators.

  The aCC compiler is strict when distinguishing between a void pointer, `void *`, and a type pointer, such as `char *`.

  There is one workaround to this problem: use the TipcMsg::append and TipcMsg::next methods to insert and extract fields from messages.

- When using a stream << operator with a TipcMsg object.

  For example:

  ```
  cout << msg.getDest() << endl;
  ```

  The compiler finds an ambiguity between the operator definition in the TipcMsg class and the definition in the C++ library.

  There is one workaround to this problem: do not include the using directive, `using namespace SmartSockets`, for the SmartSockets namespace in files using the << operator with a TipcMsg object. Instead, use the scope operator to access all SmartSockets classes and functions.

Sun SPARC Solaris

To ensure that SmartSockets works properly, you may need these Sun patches. Sun patches are available by anonymous FTP from the Sun web site, `http://sunsolve.sun.com`.

We recommend that the latest Sun patch cluster be installed on your system. Because there are patches to Solaris that may affect the correct operation of SmartSockets on Solaris, it is important you install the latest patches.

SmartSockets Software Release 6.8 has not been tested on all Sun patch revisions. Report any problems to TIBCO Product Support.

Sun Patch 106300-XX

On Solaris Version 7, install the Sun patch 106300-*XX*, where *XX* represents the current Sun patch revision number. This is a shared library patch for 64-bit C++, and is required to build and run 64-bit C++ RTclients on Solaris 7.

Sun Patch 106327-XX

On Solaris Version 7, install the Sun patch 106327-*XX*, where *XX* represents the current Sun patch revision number. This is a shared library patch for 32 bit C++, and is required to build and run 64-bit C++ RTclients on Solaris 7.

Sun Patch 106541-XX

On Solaris Version 7, install the Sun patch 106541-*XX*, where *XX* represents the current Sun patch revision number. This patch addresses a TCP packet corruption issue at the operating system level.

Sun Patch 107081-XX

On Solaris Version 7, install the Sun patch 107081-*XX libXm RunTime Kit Patch*, where *XX* represents the current Sun patch revision number.

Sun Patch 107311-XX

On Solaris Versions 7 and 8, install the Sun patch 107311-*XX*, where *XX* is at least number 18. this patch is required if you are using the Sun Workshop 5.0 C++ compiler.

Sun Patch 108528-XX

On Solaris Version 8, install the Sun patch 108528-*XX*, where *XX* represents the current Sun patch revision number. This patch addresses a TCP packet corruption issue at the operating system level.

Sun Patch 108940-XX

On Solaris Version 8, install the Sun patch 108940-*XX libXm RunTime Kit Patch*, where *XX* represents the current Sun patch revision number.

Chapter 3 Installing on UNIX

This chapter describes the installation process for most UNIX platforms. If information on your platform is not covered here, check the README file shipped with the product. The README file contains the most current information for the product, and may contain information on platforms not covered in this book.

Be sure to check the UNIX Platform Notes on page 30 section for any special notes that apply to the UNIX platform on which you are installing SmartSockets.

Topics

Upgrading From Previous Releases

If you are upgrading from a previous release of SmartSockets, you must install this release in a new directory. Usually, a new directory is created automatically. Installing this Software Release 6.8 release creates an `ss68` directory and installs SmartSockets into it. For additional upgrade considerations, see Upgrading Your TIBCO SmartSockets Installation on page 5.

If you are upgrading from a previous version, do not delete your existing license file or any other customized files. You may also want to make copies of your application files, such as `.cm` configuration files, before converting them to Software Release 6.8. The process of converting applications is described in detail in Migration and Compatibility on page 5.

Structure of the Product Release

The SmartSockets release is made up of several subsets. The environment variable `$RTARCH` refers to the hardware and operating system combination for which the files are made.

Subset	Contents
1	Text files and other platform-independent files, such as shell scripts, header files, standard text files, help text files, and the SmartSockets Java class library for RTclient only
2	Installation files
3	Debug versions of SmartSockets binary files (directories `lib/$RTARCH` and `bin/$RTARCH`)
4	Optimized versions of SmartSockets binary files (directories `lib/$RTARCH` and `bin/$RTARCH`)
5	Examples and source
6	Online Java API reference in JavaDoc format, code examples, and reference documentation for the `sscpp` class library

Installing TIBCO SmartSockets

The installation process consists of:

- loading the SmartSockets release onto your system, either from the FTP site or from the product CDROM
- printing the README file and the PDF version of this book
- running the installation scripts
- setting up user accounts to run SmartSockets by adding a line to either the `$HOME/.cshrc` or `$HOME/.profile` file for each user
- contacting TIBCO Product Support for your license key and customer ID
- updating the license file with your license key

Step 1 **Print the README file and the PDF version of this book**

You need to read the README file and this installation guide to understand the installation process.

Step 2 **Access the computer from which you will install SmartSockets**

Log on to a computer that has access to the directory in which SmartSockets is to be installed. If you are installing from CDROM, the architecture of the computer must be compatible with the architecture of the product CDROM. For example, an HP-UX Version 11.x CDROM can be installed on an HP-UX Version 11.x workstation but not on a Solaris Version 2.x workstation.

Step 3 **Ensure the system meets all requirements**

Ensure the system on which you want to install SmartSockets meets all the hardware and software requirements described in Chapter 2, System Requirements.

Step 4 **Run the installation script**

1. Download the TIBCO SmartSockets installation package, or access the release package from the CDROM. The file name is:

   `TIB_ss_`*`num_platform`*`.tar`

   where *num* is the SmartSockets software release number and *platform* is the platform and operating system. For example, filename might be:

   `TIB_ss_6.8.0_s4_sol57_CC.tar`

2. Change to the directory in which SmartSockets is to be installed.

   For example:

   ```
   cd /usr/local
   ```

3. Untar the product distribution. For example:

   ```
   tar xvf TIB_ss_6.8.0_s4_sol57_CC.tar
   ```

4. Run the `/installSS` script. For example:

   ```
   $ ./installSS
   ```

   This creates the SmartSockets directory `ss68` and unpacks the SmartSockets subsets. The subsets are described in Structure of the Product Release on page 18.

   The `install` script guides you through the rest of the installation process. You can stop the installation process at anytime by sending an INTERRUPT signal, usually a Ctrl-c. However, stop the installation script only when `install` is prompting you for information. If `install` is stopped while it is in the middle of accessing the drive or modifying the software, the software can be corrupted and you must reinstall the software from the beginning.

5. Read the TIBCO SmartSockets license agreement.

   During the installation, you are prompted to read the TIBCO license agreement. You must agree to the TIBCO license agreement to install SmartSockets. Enter `y` to accept the license agreement. If you do not agree, enter `n` to exit the installation.

6. Select the debug or optimized installation for SmartSockets.

   SmartSockets offers two types of installation. The optimized only installation installs just the optimized version of the SmartSockets binary files. The debug and optimized installation installs both the optimized and debug binary files. During the SmartSockets installation, you are prompted to select a installation type:

   ```
   Would you like to install optimized only version? [y]:
   ```

   Entering `y` installs the optimized only version of the SmartSockets binary files. Enter `n` if you want to install both the optimized and debug versions. If you enter `n`, the next prompt is:

   ```
   Would you like to install both optimzed and debug version? [y]:
   ```

   Enter `y` to install both the optimized and debug versions of SmartSockets binary files.

   When the installation in complete, you have the option to view the SmartSockets README. Enter `y` to display the README, or `n` to continue.

7. Change to the `ss68` directory:

   ```
   cd ss68
   ```

8. Now use the `rtmove` script to set up the shell scripts.

   ```
   install/rtmove
   ```

Step 5 **Set up the user accounts**

On UNIX, root or super user privilege is not required to run an RTserver. RTserver is a normal user process.

To run SmartSockets, you must add a SmartSockets script to your command interpreter (also called a shell) startup script. This startup shell script is usually located in your home directory, `$HOME`. The process of adding the SmartSockets script is not automated because startup shell scripts vary.

These shells are supported by SmartSockets:

- C shell (`csh`)
- Bourne shell (`sh`)
- Korn shell (`ksh`)
- enhanced C shell (`tcsh`)
- GNU Bourne Again shell (`bash`)

Contact your system administrator if you are unsure of which shell a user account is using.

1. Add the `rtinit` script to the startup shell script.

   C Shell and Enhanced C Shell

   For C shell and enhanced C shell users, the script `rtinit.csh` should be used. The `rtinit.csh` script sets or modifies several environment variables.

   Add this line to the C shell startup script `$HOME/.cshrc`:

   ```
   source install_dir/ss68/bin/rtinit.csh
   ```

   Add the line towards the end of a user's `$HOME/.cshrc` file, so that the settings in `rtinit.csh` are applied last. For more information on `rtinit.csh`, see the information for `rtinit.csh` in the *TIBCO SmartSockets Utilities* reference.

Bourne Shell, Korn Shell, and Bourne Again Shell

For Bourne shell, Korn shell, and Bourne Again shell users, the script `rtinit.sh` should be used. The `rtinit.sh` script sets or modifies several environment variables. Add this line to the shell startup script `$HOME/.profile`:

```
. install_dir/ss68/bin/rtinit.sh
```

Add the line towards the end of your `$HOME/.profile` file, so that the settings in `rtinit.sh` are applied last. For more information on `rtinit.sh`, see the information for `rtinit.sh` in the *TIBCO SmartSockets Utilities* reference.

2. Verify that the SmartSockets environment is properly set up.

   Many users have a conditional exit in their `$HOME/.cshrc` or `$HOME/.profile` files that prevent certain commands from being executed if the invocation is non-interactive (such as when RTmon is being invoked from a script). The standard C shell test for an interactive login is:

   ```
   if (! $?prompt) then
      exit
   endif
   ```

   The line that invokes either the `rtinit.csh` or `rtinit.sh` script should appear before this conditional exit, or the SmartSockets environment is not properly set up, and the shell responds with a statement similar to that which follows a non-interactive login:

   ```
   rtmon: Command not found.
   ```

   Before you run the examples you must set up an account.

   To have the changes made to a `.cshrc` file take effect immediately, use this command:

   ```
   $ source $HOME/.cshrc
   ```

   When using Bourne shell and Korn shell, use this command:

   ```
   $ . $HOME/.profile
   ```

Step 6 **Update the license file**

Before you can use SmartSockets or start the RTserver, you must license it. See Licensing TIBCO SmartSockets on UNIX Systems on page 23.

Licensing TIBCO SmartSockets on UNIX Systems

You enter a license key and customer name into your SmartSockets system before certain processes are allowed to run. Formerly, these processes were branded with the license key and customer name or ID, using the `rtbrand` command. Now, there is a license file, stored in the `standard` directory, where you enter your product licenses and customer name.

You can still use the `rtbrand` command to brand your RTservers, instead of using the license file, but if you add your SmartSockets license to the license file, it overrides any branding. The license file takes priority over branding.

Your License Key and Customer ID

A license key is needed to license the RTservers and other processes for SmartSockets. These processes cannot be run until they have been licensed or branded. You also get a customer ID, which is encoded into your license. You need your customer ID to provide to TIBCO Product Support when you contact them with a question or support issue. Contact TIBCO Product Support to get both the license key and customer ID.

The license key is a 32 character string. Only hexadecimal characters are valid. It is case-insensitive. It can contain embedded hyphens as a readability aid.

Here is an example of a valid license key:

```
12345678-90ABCDEF-12345678-90ABCDEF
```

The customer ID is a numeric string, such as `4425`. When you receive your customer ID, note it someplace for future use when requesting product support. If you forget your customer ID, you can use this SmartSockets command to display your customer information:

```
rtserver -license
```

On platforms that support both 32- and 64-bit, use the `rtserver64` command to run the 64-bit version of RTserver.

Modifying the License File

An empty license file is created during your SmartSockets installation, and can be found in:

```
$RTHOME/standard/talarian.lic
```

Use an editor, such as `vi`, to edit the file and add your license information, including the product name and serial number and your company name. If your company name includes a space, it must be enclosed in double quotes. The product name and serial number must match the name and number sent to you in the Technical Support email that provides your license information. The best way to ensure a match is to copy the line in the email that includes the product name and serial number and paste it into the license file.

Here is a sample license file for a system that has several products, including SmartSockets:

```
#
# License file for TIBCO products.
#
# Licenses for TIBCO products may be added to this file,
# $RTHOME/standard/talarian.lic, or the TIBCO product may
# be branded. Any license found in this file will override a
# branded TIBCO product.
#

Product               Serial Number                          Company Name
--------------------- -------------------------------------  -----------------
SmartSockets          C586A216-12345678-0C2B6775-FEAD0AFF     "Acme Sockets"
SmartSockets/Gateway 4E85BE66-28D8437B-1F6017E5-12345678     "Acme Sockets"
SmartSockets/PGM      0384DB01-E0550592-56EB3ED5-12345678     "Acme Sockets"
SmartSockets/Cache    F5860E16-12345678-0C2B6775-FEA10A0F     "Acme Sockets"
```

Licenses you have added to this file override any branding you might do. The license file takes priority over use of the `rtbrand` command.

For this SmartSockets installation, add a line for the SmartSockets product name and serial number. If you licensed SmartSockets Multicast (a license for the RTgms process to support multicasting), also add a line for that product name and serial number to the file.

After you have edited the file and saved it, RTserver can now be started. Starting RTserver now helps verify that the SmartSockets installation was successful. To start RTserver, type this command at the prompt:

```
$ rtserver
```

A message appears saying that the RTserver was started successfully.

On platforms that support both 32- and 64-bit, use the `rtserver64` command to run the 64-bit version of RTserver.

If you have an existing SmartSockets license and want to upgrade or add other products, you also edit the license file. To add products add a new line in the license file. To upgrade an existing license, with additional RTservers or upgrading to multi-threaded RTservers, replace the existing SmartSockets serial number with the new serial number.

Displaying the License File

After you have installed SmartSockets, added the product license to the license file, and started the RTserver, you can use the `rtlic` shell script to display information about your license. At a SmartSockets command prompt, enter:

```
$ rtlic
```

This is a sample of what is displayed:

```
product              = SmartSockets
customer check sum   = <11>
server_count         = <20>
client_count         = <200>
customer_id          = <1234>
product_version      = <65>
reference_date       = <Tue Nov 23 13:11:42.000 2003>
customer_name        = <Superior Messaging, Inc.>
eval                 = <false>
capabilities         = <14>
                     = "Advanced Usage"
                     = "Multi-Threaded"
                     = "Group Connections"
```

For more information on using `rtlic`, see the *TIBCO SmartSockets Utilities* reference.

Branding a License

If you want to use branding instead of the license file, use the `rtbrand` command. You can also use this command to upgrade a standard SmartSockets Software Release 6.8 license to add the SmartSockets Multicast or SmartSockets MP features.

There are multiple `rtserver` executables installed during SmartSockets installation. If you have an existing SmartSockets license and want to upgrade by rebranding, you need to explicitly rebrand each configuration (each executable) of the RTserver that you use. For example, if you use the optimized version of the RTserver, you must rebrand that in addition to rebranding the default version.

Using your new license, modify your existing SmartSockets license, invoking `rtbrand` for each configuration of the RTserver that you use:

`rtbrand` *license_key customer_name rtserver_exec*

where:

license_key is the new license key you obtained from TIBCO Product Support.

customer_name is your customer or company name enclosed in quotes. It is helpful if this name is the same as the company or division name you provided to TIBCO Product Support when you were getting your license.

rtserver_exec is the name or pathname to the installed RTserver executable, where:

- `rtserver_d.x` is the explicit name of the executable for the default RTserver
- `rtserver_o.x` is the name of the executable for the optimized RTserver
- `rtserver.x` is a link to the optimized RTserver executable

This example brands the default RTserver with the new license and customer name:

```
$ rtbrand 12345678-90ABCDEF-12345678-90ABCDEF "Acme Sockets" rtserver.x
```

Installing and Sharing Between Multiple Architectures

The directory structure of SmartSockets puts architecture-dependent files, such as executables and object libraries, in their own `RTARCH` directories. To install and share SmartSockets for several architectures, Network File System (NFS) can be used to share file systems between several UNIX computers. Install SmartSockets on the first architecture, then run the `rtinstall` installation script on all subsequent architectures to unload the architecture-dependent subsets. A SmartSockets product CDROM is needed for each architecture. Subsets 1 (text files) and 3 (examples and source) from the SmartSockets CDROM can be shared between all architectures.

The shared architectures should be the same revision of SmartSockets. The revision is printed on the face of the CDROM and is also listed when a standard SmartSockets process is executed. If the shared architectures are not the same revision of SmartSockets, the shared files might not be compatible. Contact TIBCO Product Support if you have questions about a specific combination of revisions. UNIX and OpenVMS architectures cannot be shared in the same directory.

For example, if you have two SmartSockets CDROM disks, one for the SPARC architecture and one for the HP 9000/7xx architecture, and you want to install both of them in the directory `/usr/local/ss68`, and `/usr/local` was shared using NFS on the SPARC and HP 9000/7xx computers, follow these steps:

1. Install the SPARC version of SmartSockets into `/usr/local`.
2. Insert the HP 9000/7xx SmartSockets CDROM disk, but do not unload any of it.
3. Log in to the HP 9000/7xx computer.
4. Change directory to the `ss68` directory on the CDROM disk.
5. Run `setup` on the HP 9000/7xx computer and unload the binaries you want to use.

Moving the Product Directory

To move SmartSockets from one directory to another after it has been installed, either:

- remove the existing directory `$RTHOME` and reinstall
- move `$RTHOME` from one directory to another and reconfigure SmartSockets for the new value of `$RTHOME`

These two methods are described here in more detail.

Removing the Existing Directory and Reinstalling

To remove the existing SmartSockets directory, first shut down all SmartSockets processes that are using the directory. You should warn anyone using those processes before stopping them.

1. You can stop RTservers and all processes attached to them by including the `-stop_all` argument with the `rtserver` command:

   ```
   $ rtserver -stop_all
   ```

If an RTserver is stopped without stopping the RTclients, those RTclients can be configured to automatically try to restart the RTserver.

2. Before removing the old directory, save any SmartSockets customized files. For example, the files in the `$RTHOME/standard` directory, which include your license file and the command (`.cm`) files, should be saved in another location.
3. Remove the directory using this command:

   ```
   $ rm -rf $RTHOME
   ```

4. Follow the instructions in Installing TIBCO SmartSockets on page 19 to re-install SmartSockets.

Moving and Reconfiguring the Directory Tree

To move SmartSockets from one directory to another on the same disk partition, use the UNIX `mv` command. For example, if `$RTHOME` was to be moved from the directory `/usr/local/ss68` to a new directory named `/usr/ss68`, use this command:

```
$ mv /usr/local/ss68 /usr/ss68
```

If SmartSockets is being moved between different disk partitions and the `mv` command failed, this error message is displayed:

```
mv: can't mv directories across file systems
```

To move SmartSockets between disk partitions, use the UNIX `tar` command. Do not use the `cp` command, as `cp` does not preserve symbolic links. The contents of the file that the link refers to are copied, rather than the link itself being copied. Using the `cp` command to move SmartSockets results in extra disk storage being used for the copies of many files.

The `tar` command can be used twice in a pipeline to create an archive and unload it in one step. If `$RTHOME` is to be moved from `/usr/local/ss68` to `/usr/ss68`, for example, use this command:

```
$ cd /usr
$ (chdir /usr/local; tar cf - ss68) | tar xvpf -
$ rm -rf /usr/local/ss68
```

Once SmartSockets has been moved to its new location, the shell script `rtmove` must be executed to change the value of `$RTHOME` in the SmartSockets text files. This script must be executed from the `ss68` directory:

```
$ cd /usr/ss68
$ install/rtmove
```

The `rtmove` script guides the user through the process. This text is displayed when running `rtmove`:

```
Welcome to rtmove, which reinitializes SmartSockets
once it has been moved to a new directory.

When this script asks you questions, a default answer
is shown in brackets next to the question. To use the
default answer, press <Return> in response to the question.
Otherwise, type one of the other suggested responses and
then press <Return>.

This script will ask all its questions first, and then
execute the actions you select.

Would you like to continue? [y]: (press return)
```

```
This script changes several SmartSockets files to reference this
directory. Therefore it needs to know the correct name of
this directory. If you will be accessing this directory via
NFS on several hosts, make sure the pathname is valid on all
those hosts. Do not use shell metacharacters (such as *
or ~) when answering this question.

Please enter the full pathname that you would like to use to refer
to this directory [/usr/ss68]: (press return)
Modifying SmartSockets text files to reference /usr/ss68.
Updating libraries.

Done moving SmartSockets to /usr/ss68.
```

UNIX Platform Notes

This section covers any special notes pertaining to certain UNIX platforms. Also check the README file shipped with the product. The README file contains the most current information for the product, and may contain information on platforms not covered in this book or information that is more current for your platform.

Sun Solaris

Solaris 64-Bit Shared Libraries

SmartSockets for Solaris includes a set of 64-bit shared libraries for building 64-bit RTclients. The `rtlink` shell script builds 64-bit versions of RTclients when you specify the `-64bit` flag. These libraries support the LP64 data model, using the compiler flag `-xarch=generic64`.

Build an RTclient application with the 64-bit client libraries by including the `-64bit` flag in the `rtlink` shell script. For example, this builds `send64.x` with the appropriate compiler flags and SmartSockets and operating system libraries for 64-bit applications:

```
rtlink -64bit -o send64.x send64.c
```

See the *Solaris 64-bit Developer's Guide* in the *Solaris Software Developer Collection* for further details, including the trade-offs involved in the use of 64-bit applications.

Hardware and Software Requirements

To use the 64-bit RTclient libraries you must have a 64-bit version of Solaris installed on hardware that supports a 64-bit instruction set (sparcv9, EM64T, or AMD64). Use `isainfo -b` to confirm that you have a suitable Solaris 64-bit environment. These patches are also recommended (*XX* represents the current Sun Microsystems patch revision number):

For Solaris Version 7 only

- 106327-*XX* — shared library patch for 32 bit C++
- 106300-*XX* — shared library patch for 64-bit C++

For Solaris Versions 7 and 8 only

- 107311-*XX*, where *XX* is at least number 18, if you are using the Sun Workshop 5.0 C++ compiler

See the *Solaris 64-bit Developer's Guide* in the *Solaris Software Developer Collection* for further details, including the trade-offs involved in the use of 64-bit applications.

GMD Restrictions

Solaris supports `tmpfs`, a memory-based file system used for increased performance, particularly with small, short-lived files. This file system type can be mounted on `/tmp`, which can severely disrupt file-based GMD functions. File-based GMD requires data to be written to an actual physical storage medium, and not simply cached in memory, so that GMD can continue to function after a possible reboot.

The `tmpfs` file system contents do not remain after restarting your system. In fact, they may never be written to a hard disk at all. Even when they are, it is to the swap filespace. Do not allow file-based GMD to be written to such partitions. Change the Ipc_Gmd_Directory RTclient option to a path on a different file system, or discontinue use of the `tmpfs` file system for `/tmp`.

High CPU Load and Scheduling Classes

Although it is not recommended that SmartSockets, especially RTservers, be used on machines with already overloaded CPUs, it is sometimes necessary. In these cases, it is important to recognize that lack of response from SmartSockets is generally caused by the overall system load and not a failure of the software. Understanding how your system allocates CPU time and setting process priorities and scheduling groups appropriately can make a large difference. For example, it might be necessary to do more than simply changing the priority with `nice`. In these cases, moving RTserver or other time-critical SmartSockets processes into a higher-priority, Real-Time scheduling class is useful. To change the scheduling class of a process, use the Solaris `priocntl` tool. For details of which classes are available and how to use `priocntl`, check your system's `priocntl` man page.

WorkShop Version 5.0 -xO4 Optimization Level

SmartSockets is compiled on SPARC Solaris using the -xO4 optimization option of the Sun Microsystems WorkShop 5.0 C and C++ compilers. The `rtlink` shell script builds optimized RTclients on Solaris using the -xO4 optimization option. This is Sun Microsystems' recommended optimization level with the WorkShop Version 5.0 compilers.

Hewlett-Packard HP-UX

HP-UX Version 11.x 64-Bit Shared Libraries

SmartSockets for HP-UX Version 11.x includes a set of 64-bit shared libraries for building 64-bit RTclients. The `rtlink` shell script builds 64-bit versions of RTclients when you specify the `-64bit` flag. These libraries support the HP-UX Version 11.x LP64 data model, using the compiler flag `+DD64 (c)` or `+DA2.0W` (cxx/cpp on HP 9000/7xx hardware).

For example:

```
rtlink -64bit -o send64.x send64.c
```

builds `send64.x` with the appropriate compiler flags and SmartSockets and operating system libraries for 64-bit applications.

See the Hewlett-Packard document *HP-UX 64-bit Porting and Transition Guide for HP9000 Computers* for more information on issues associated with 64-bit applications.

Hardware and Software Requirements

To use the HP-UX 64-bit RTclient libraries you must have a 64-bit version of HP-UX 11.x installed on Itanium hardware, or on PA_RISC 2.0 hardware that supports a 64-bit instruction set. Use `getconf KERNEL_BITS` to confirm that you have a suitable HP-UX Version 11.x 64-bit environment. Hewlett-Packard recommends configuring HP-UX Version 11.x for additional system resources to run 64-bit applications. Increase disk space and physical memory to provide more than 4 GB of swap and 4GB of memory, and increase these kernel parameters:

Parameter	Minimum Value
`maxdsiz`	`0x40000000`
`maxdsiz_64bit`	`0x400000000`
`maxswapchunks`	`10000`

See the Hewlett-Packard document *HP-UX 64-bit Porting and Transition Guide for HP9000 Computers* for more information on issues associated with 64-bit applications.

HP-UX Thread Support

The HP-UX Version 11.x release uses the POSIX FINAL 1003.1c thread API to implement the SmartSockets cross-platform thread API.

If a SmartSockets application mixes calls to the SmartSockets cross-platform thread API with calls to the native POSIX thread API they must be POSIX FINAL 1003.1c calls when using the SmartSockets HP-UX Version 11.x release. See `/usr/include/pthread.h` for more information on how API calls are handled on HP-UX.

Advanced Tuning

If you do advanced tuning with your HP-UX system, you might not want to use the defaults provided by SmartSockets and your HP system, especially the defaults for _M_ARENA_OPTS and _M_SBA_OPTS. We provide some tips here for tuning them. If you have any questions about tuning, contact your HP technical advisor.

_M_ARENA_OPTS can be used to to adjust the number of arenas and how many pages are used each time an arena expands itself (the expansion factor), assuming that the page size is 4096 bytes. In general, the more threads in an application, the more arenas are needed to improve performance. _M_ARENA_OPTS does not affect non-threaded applications.

Specify it using this syntax:

_M_ARENA_OPTS = *arena_num*:*expansion_factor*

where:

arena_num specifies the number of arenas. For non-threaded applications, only one arena is used.

For threaded applications, the valid range is 4 to 64, and the default value is 8. If *arena_num* is not set or you specify a number outside the valid range, the default of 8 is used.

expansion_factor specifies the expansion factor. The valid range is 1 to 4096. The default value is 32. If you specify a value outside the valid range, the default is used.

Here is an example:

```
$ export _M_ARENA_OPTS = 16:8
```

This sets the number of arenas to 16 and the expansion factor to 8 pages, each page being 4096 bytes. In general, the more arenas you use, the smaller the expansion factor should be, and the fewer the arenas, the larger the expansion factor.

_M_SBA_OPTS turns on the small block allocator, and sets up the *maxfast*, *grain*, and *numblks* parameters for the small block allocator. _M_SBA_OPTS affects both threaded and non-threaded applications. Applications usually run faster with the small block allocator turned on. The small block allocator can be turned on using `mallopt()`. However, that does not work well for C++ or Java applications. The environment variable turns it on before the application starts. The `mallopt()` call can still be used the same way. If the environment variable is set, and no small block allocator has been used, the subsequent `mallopt()`calls can still overwrite whatever is set using _M_SBA_OPTS. If the environment variable is set, and small block allocator has been used, then `mallopt()` has no effect.

Specify it using this syntax:

_M_SBA_OPTS = *maxfast*:*numblks*:*grain*

where:

maxfast specifies the maxfast size.

numblks specifies the number of small blocks.

grain specifies the grain size.

Here is an example:

```
$ export _M_SBA_OPTS = 512:100:16
```

This sets the maxfast size to 512, the number of small blocks to 100, and the grain size to 16. You must supply all three values, in the order shown. If you do not, the default values are used instead.

Linux

SmartSockets Software Release 6.8 was built with Red Hat Enterprise Linux Version 3.

SmartSockets on Linux includes the runtime RTmon only with its Command Interface (CI) and does not support the RTmon graphical user interface (GUI).

To build C++ RTclients using the `g++` compiler set the CC environment variable to `g++` when using the `rtlink` shell script, as in this example:

```
env CC=g++ rtlink -cxx -o myclient.x myclient.cxx
```

SmartSockets Software Release 6.8 running on Red Hat Linux looks for a shared library `libg++.so`. Create a symbolic link to the latest version of `libg++` on your system. For example:

```
ln -s /usr/lib/libg++.so.2.7.2.8 /usr/lib/libg++.so
```

Linux 64-Bit Shared Libraries

SmartSockets for x86 Linux includes a set of 64-bit shared libraries for building 64-bit RTclients. The `rtlink` shell script builds 64-bit versions of RTclients when you specify the -64bit flag. These libraries support the LP64 data model, using the compiler flag -m64.

For example:

```
rtlink -64bit -o send64.x send64.c
```

builds send64.x with the appropriate compiler flags and SmartSockets and operating system libraries for 64-bit applications.

Hardware and Software Requirements

To use the x86 64-bit RTclient libraries you must have a 64-bit version of Linux installed on hardware that supports the AMD64 or EM64T instruction set. Use `uname -m` to confirm that you have a suitable Linux 64-bit environment.

IBM AIX

Hardware and operating system software must support 64-bit to run the 64-bit version of SmartSockets.

To build and run 64-bit SmartSockets applications, `LIBPATH` must include `$RTHOME/lib/$RTARCH/64bit` *before* `$RTHOME/lib/$RTARCH`. Because AIX is not able to fall back on additional paths when a library mismatch (64-bit vs. 32-bit) occurs, RTserver must be started with the command `rtserver64_o.x` (to run optimized) or `rtserver64_d.x` (to run debug), and will normally run in the foreground. Not all options will be available when starting RTserver in this manner.

To stop RTserver, you must use the `rtserver` command, and `$RTHOME/lib/$RTARCH/64bit` *must not* appear in your `$LIBPATH`.

HP Alpha Tru64 UNIX

There are no platform-specific notes for Tru64 at this time.

Silicon Graphics IRIX

SmartSockets IRIX releases only support the N32 object code format.

This section presents brief information about the IRIX object code formats. For more information, see the abi(5) man page.

Supported Object Code Format

An Application Binary Interface (ABI) defines a system interface for executing compiled programs, defining the supported instruction set architectures (ISA), and the object file formats and calling conventions. IRIX Version 6.2 supports these ABIs:

Option	Description
-32 (O32)	The old 32 bit ABI that was standard on IRIX Version 5 systems. The -32 option is the default when compiling on any non-R8000 based machine. This ABI is not supported by SmartSockets.
-n32 (N32)	This high performance 32 bit ABI was introduced on IRIX Version 6.2, and generates an -n32 object. All SGI platforms support the development and execution of -n32 applications, but the subsystems containing the appropriate libraries are not always installed by default. This is the only ABI supported by SmartSockets.
-n64 (N64)	The 64 bit ABI was introduced on IRIX Version 6.0 systems. This is the default when compiling on any R8000 based machine, if no default specification file is present. This ABI is not supported by SmartSockets.

Each of these ABIs defines unique interfaces that make it impossible to link objects files of one ABI with object files of another ABI.

Thread Support on IRIX 6.5

An SGI Indy running IRIX 6.5 was used to test and release the latest SmartSockets distribution for IRIX. See the README file for the SmartSockets IRIX release for any recommended SGI patches to provide thread support.

Chapter 4 Installing on Windows

This chapter describes the installation process for Windows platforms. If information on your platform is not covered here, check the README file shipped with the product. The README file contains the most current information for the product, and may contain information on platforms not covered in this book.

Check the Windows Platform Notes on page 50 section for any special notes that apply to the Windows platform on which you are installing SmartSockets.

Topics

Upgrading From Previous Releases

Installation of different SmartSockets versions on the same Windows machine is currently not supported. Before installing this release of SmartSockets, you should uninstall any previous releases on this machine. For additional upgrade considerations, see Upgrading Your TIBCO SmartSockets Installation on page 5.

If you are upgrading from a previous version, do not delete your existing license file or any other customized files.You might also want to make copies of your application files, such as `.cm` configuration files, before converting them to Software Release 6.8. The process of converting applications is described in detail in Migration and Compatibility on page 5.

Installing TIBCO SmartSockets

The installation process consists of:

- loading the SmartSockets release onto your system, either from the FTP site or from the product CDROM
- printing the README file and the PDF version of this book
- running Install Shield, and stepping through the installation with it
- verifying the installation of your selected components
- contacting TIBCO Product Support for your license key and customer ID
- updating the license file with your license key

Step 1 **Print the README file and the PDF version of this book**

You need to read the README file and this installation guide to understand the installation process.

Step 2 **Access the computer from which you will install SmartSockets**

Log onto an administrator account on the computer where SmartSockets is to be installed.

Step 3 **Ensure the system meets all requirements**

Ensure the system on which you want to install SmartSockets meets all the hardware and software requirements described in Chapter 2, System Requirements.

Step 4 **Start InstallShield**

Follow these steps:

1. Download the TIBCO SmartSockets installation package, or access the release package from the CDROM.
2. Run the SmartSockets `.exe` file, either by double-clicking it or by going to the Start menu, selecting Run, and entering the location and name of the file, and pressing the Enter key.

   The name of the file follows the format `TIB_ss_`*`num_platform`*`.exe`, where *num* is the SmartSockets software release number and *platform* is the platform, operating system and compiler. For example, for SmartSockets Software Release 6.8.0, the filename is:

   ```
   TIB_ss_6.8.0_w32_VC8.exe
   ```

InstallShield begins running.

Step 5 **The TIBCO SmartSockets Welcome dialogue box appears**

Click Next to continue the installation.

Step 6 **Read the End User License Agreement**

The License Agreement dialogue box appears. Read the license agreement carefully. If you agree with the terms, click Yes. If you do not agree, click No to exit InstallShield.

Step 7 **Choose the folder where SmartSockets should be installed**

The Choose Destination Location dialog box appears. Click Browse to choose a folder other than `C:\Program Files\TIBCO\ss68`. Choose a new folder by typing in the pathname or by browsing your hard drive. Click Next after you have selected a folder. The folder you choose here is referred to as the *install_dir* in the rest of this book.

Step 8 **Select the installation type**

The Setup Type dialog box appears next. Choosing the installation type determines which components are installed on your machine. As your cursor highlights a selection, a description of the type of installation appears.

There are seven installation types.

Type	Description
Custom Installation	Allows you to select individual components to install.
Developer Installation	Installs everything except deprecated APIs.
Runtime Installation (32-bit C/C++)	Installs only those files needed to run 32-bit SmartSockets C/C++ clients.
Runtime Installation (64-bit C/C++)	Installs only those files needed to run 64-bit SmartSockets C/C++ clients.
Runtime Installation (Java)	Installs only those files needed to run SmartSockets Java clients.
Runtime Installation (.NET)	Installs only those files needed to run SmartSockets .NET clients.
Network Server Installation	Installs the 32- and 64-bit versions of RTserver and necessary runtime.

Select a type in the dialog box and click Next.

Custom Installation:

If you selected Custom Installation, the Select Components dialog box appears and you see the tree of SmartSockets components. Selecting a component displays a description of the component.

To choose a component or subcomponent, select it and check the check box. When the check box is checked, the size of the component is displayed, so you can estimate the amount of space needed for your custom installation.

Some components have prerequisites that must be installed. When you select such a component, its prerequisite components will also be selected and greyed out. To unselect a component that is a prerequisite, you must first unselect the component that requires it. At minimum, you should install `Redistributables`, `Release Notes`, and `Runtime Libraries` from `Required Files`, and `Support Files`.

Click Next when you are finished.

Step 9 **Select a Program Folder**

The Select Program Folder dialog box appears. Unless you choose another folder, the Setup program adds the SmartSockets program icons to the SmartSockets Program Folder. If you prefer a different folder, type in a new folder name or select a folder from the Existing Folders list. Click Next to continue.

Step 10 **Check your installation information**

The Start Copying Files dialog box appears. Check the information about your installation in the display. To change information, click Back until you reach the dialog box where the information was entered.

If all settings are correct, click Next to continue. The file copying begins.

Step 11 **Choose whether to install RTserver as an NT Service**

The RTserver as NT Service dialog box appears. Choose whether you want to install the RTserver as a Windows NT service by clicking the Yes or No button.

Step 12 **Choose automatic or manual start for RTserver**

The Choose Automatic or Manual Start dialog box appears if you chose to install RTserver as a Windows NT service in Step 11. Select the start option and click Next.

Step 13 **Finish setup**

The Setup Complete dialog box appears. Uncheck the first box if you do not want to view the README file. If you would like to open the license file for editing at this time, check the second box.

Click Finish.

Step 14 **Update the license information for your installation**

You must update the license file or brand your RTserver with a license before you can run the RTserver. Edit the license file to add the license information. See Licensing TIBCO SmartSockets on Windows Systems on page 46.

Invoking TIBCO SmartSockets

A SmartSockets program folder is automatically created when you install SmartSockets. To invoke a SmartSockets process:

- go to the **Start > Programs** menu and find the program folder for SmartSockets
- select **Command Prompt** from the SmartSockets program menu type in the command needed to start the process you want to start. For example, to start the RTserver, type `rtserver`.

On platforms that support both 32- and 64-bit, use the `rtserver64` command to run the 64-bit version of RTserver.

You cannot invoke any SmartSockets processes until you licensed or branded SmartSockets. See Licensing TIBCO SmartSockets on Windows Systems on page 46.

Path Settings

If you do not want to start SmartSockets processes using **Command Prompt** from the SmartSockets program menu, you must add some settings to your PATH, INCLUDE, and LIB statements on your Windows machine. Add these to the appropriate statements:

PATH: `%RTHOME%\bin;`
`%RTHOME%\bin\%RTARCH%;`
`%RTHOME%\bin\%RTARCH%\amd64;`

INCLUDE: `%RTHOME%\include`

LIB: `%RTHOME%\lib\%RTARCH%`

To build 64-bit SmartSockets Applications without using rtlink, you should either set `LIB` to `%RTHOME%\lib\i86_w32\amd64;%LIB%` or add `/LIBPATH:%RTHOME%\lib\i86_w32\amd64` to your link options.

Licensing TIBCO SmartSockets on Windows Systems

You enter a license key and customer name into your SmartSockets system before certain processes are allowed to run. Formerly, these processes were branded with the license key and customer name or ID, using the `rtbrand` command. Now, there is a license file, stored in the `standard` directory, where you enter your product licenses and customer name.

You can still use the `rtbrand` command to brand your RTservers, instead of using the license file, but if you add your SmartSockets license to the license file, it overrides any branding. The license file takes priority over branding.

Your License Key and Customer ID

A license key is needed to license the RTservers and other processes for SmartSockets. These processes cannot be run until they have been licensed or branded. You also get a customer ID, which is encoded into your license. You need your customer ID to provide to TIBCO Product Support when you contact them with a question or support issue. Contact TIBCO Product Support to get both the license key and customer ID.

The license key is a 32 character string. Only hexadecimal characters are valid. It is case-insensitive. It can contain embedded hyphens as a readability aid.

Here is an example of a valid license:

```
12345678-90ABCDEF-12345678-90ABCDEF
```

The customer ID is a numeric string, such as `4425`. When you receive your customer ID, note it someplace for future use when requesting product support. If you forget your customer ID, you can use this SmartSockets command to display your customer information:

```
rtserver -license
```

Modifying the License File

An empty license file is created during your SmartSockets installation, and can be found in:

```
%RTHOME%\standard\talarian.lic
```

Use an editor, such as `notepad`, to edit the file to add your license information, including the product name, serial number, and your company name. If your company name includes a space, it must be enclosed in double quotes. The product name and serial number must match the name and number sent to you in the Technical Support email that provides your license information. The best way to ensure a match is to copy the line in the email that includes the product name and serial number and paste it into the license file.

Here is a sample license file for a system that has several products, including SmartSockets:

```
#
# License file for TIBCO products.
#
# Licenses for TIBCO products may be added to this file,
# $RTHOME/standard/talarian.lic, or the TIBCO product may
# be branded. Any license found in this file will override a
# branded TIBCO product.
#

Product              Serial Number                          Company Name
-------------------- -------------------------------------  -----------------
SmartSockets         C586A216-12345678-0C2B6775-FEAD0AFF     "Acme Sockets"
SmartSockets/Gateway 4E85BE66-28D8437B-1F6017E5-12345678     "Acme Sockets"
SmartSockets/PGM     0384DB01-E0550592-56EB3ED5-12345678     "Acme Sockets"
SmartSockets/Cache   F5860E16-12345678-0C2B6775-FEA10A0F     "Acme Sockets"
```

Licenses you have added to this file override any branding you might do. The license file takes priority over use of the `rtbrand` command.

For this SmartSockets installation, add a line for the SmartSockets product name and serial number. If you licensed SmartSockets PGM (a license for the RTgms process to support multicasting), also add a line for that product name and serial number to the file.

After you have edited the file and saved it, RTserver can now be started. Starting RTserver now helps verify that the SmartSockets installation was successful. To start RTserver, type this command at the prompt:

```
$ rtserver
```

A message appears saying that the RTserver was started successfully.

Or go to the Start menu, select Programs, and select the SmartSockets program folder. Select RTserver (32-bit).

Select RTserver (64-bit) instead to run the 64-bit version, if your hardware and operating system support 64-bit

If RTserver started successfully, the window is automatically minimized. Check your bottom bar for the RTserver process and click on it to expand the window, which displays a message saying that the RTserver was started successfully.

If you have an existing SmartSockets license and want to upgrade or add other products, you also edit the license file. To add products, add a new line in the license file. To upgrade an existing license, with additional RTservers or upgrading to multi-threaded RTservers, replace the existing SmartSockets serial number with the new serial number.

Displaying the License File

After you have installed SmartSockets, added the RTserver license to the license file, and started the RTserver, you can use the `rtlic` shell script to display information about your license.

At a SmartSockets command prompt, enter:

```
$ rtlic
```

This is a sample of what is displayed:

```
product              = SmartSockets
customer check sum   = <11>
server_count         = <20>
client_count         = <200>
customer_id          = <1234>
product_version      = <65>
reference_date       = <Tue Nov 23 13:11:42.000 2003>
customer_name        = <Superior Messaging, Inc.>
eval                 = <false>
capabilities         = <14>
                     = "Advanced Usage"
                     = "Multi-Threaded"
                     = "Group Connections"
```

For more information on using `rtlic`, see the *TIBCO SmartSockets Utilities* reference.

Branding a License

If you want to use branding instead of the license file, use the `rtbrand` command. You can also use this command to upgrade a standard SmartSockets Version 6.8 license to add the SmartSockets Multicast or SmartSockets MP features.

There are multiple `rtserver` executables installed during SmartSockets installation. If you have an existing SmartSockets license and want to upgrade by rebranding, you need to explicitly rebrand each configuration (each executable) of the RTserver that you use. For example, if you use the optimized version of the RTserver, you must rebrand that in addition to rebranding the default version.

Using your new license, modify your existing SmartSockets license, invoking `rtbrand` for each configuration of the RTserver that you use.

Here is the syntax for the `rtbrand` script:

`rtbrand` *license_key customer_name rtserver_exec*

where:

license_key is the new license key you obtained from TIBCO Product Support (see Your License Key and Customer ID on page 46).

customer_name is your customer or company name enclosed in quotes. It is helpful if this name is the same as the company or division name you provided to TIBCO Product Support when you were getting your license.

rtserver_exec is the name or pathname to the installed RTserver executable, where:

- `rtserverd.exe` is the explicit name of the executable for the default RTserver
- `rtservero.exe` is the name of the executable for the optimized RTserver
- `rtserver.exe` is also the executable for the optimized RTserver

  Note that `rtserver.exe` and `rtservero.exe` are the same file in this release.

For example:

```
$ rtbrand 12345678-90ABCDEF-12345678-90ABCDEF "Acme Sockets" rtserver.exe
```

This brands the default RTserver with the new license and customer name.

Uninstalling TIBCO SmartSockets

Ensure that no running programs have their current directory set inside the SmartSockets installation directories while uninstalling. Any files in the installation directories that were not part of the original install process, such as `.dbg` files, are not removed. You must remove them manually before running the uninstall process.

To uninstall SmartSockets, you must be logged on as administrator. Go to the Start menu and select Programs. Select the SmartSockets program folder, and select Uninstall SmartSockets. You are prompted through the uninstall process.

Windows Platform Notes

This section covers any special notes pertaining to Windows platforms. Also check the README file shipped with the product. The README file contains the most current information for the product, and may contain information on platforms not covered in this book or information that is more current for your platform.

TIBCO SmartSockets Visual C++ Libraries

Library Naming Convention

The SmartSockets 32-bit Windows (Win32) and 64-bit Windows (Win64) libraries use this naming convention:

name`x.lib`

where:

name indicates the library module name.

x is either 'd' to indicate a debug library, or omitted to indicate and optimized library

Libraries with debug information use the debug CRT provided with Microsoft Visual C++.

The DLLs incorporate a release compatibility version number as part of the name. The SmartSockets Software Release 6.8 and all binary-compatible releases have DLLs named *name*`50x.dll` (for 32-bit DLLs) or *name*`64x.dll` (for 64-bit DLLs).

The Libraries Directory

The subdirectory `lib\i86_w32` contains the SmartSockets Win32 libraries:

File	Link Type	Description	C Runtime
`tipc.lib`	Dynamic	IPC Library	Multithread DLL
`tipcd.lib`	Dynamic	IPC Library	Debug Multithread DLL
`tipcx.lib`	Dynamic	Legacy C++ Library	Multithread DLL
`tipcxd.lib`	Dynamic	Legacy C++ Library	Debug Multithread DLL
`tsscpp.lib`	Dynamic	New C++ Library	Multithread DLL
`tsscppd.lib`	Dynamic	New C++ Library	Debug Multithread DLL
`tutil.lib`	Dynamic	Utility Library	Multithread DLL
`tutild.lib`	Dynamic	Utility Library	Debug Multithread DLL

The new C++ libraries, `tsscpp`, support multiple RTserver connections, and include many other changes. They are not source-compatible and do require code changes. Choose which C++ libraries you want to link to, either the current set or the legacy set.

Windows-Specific Features

Function Calling Conventions under Win32

SmartSockets Version 5.x and higher Win32 DLLs use the `__stdcall` calling convention. This is a change from Version 4.0 DLLs, which used the `__cdecl` calling convention. The switch was made because, as can be inferred from the name, `__stdcall` is the standard calling convention for DLLs under Win32. Using the standard calling convention confers the advantage that functions in SmartSockets Version 6.8 DLLs can be called from languages other than C, for example, Visual Basic or PowerBuilder.

Recompiling existing SmartSockets code for the new release may result in compile-time errors if the T_ENTRY declaration specifier was omitted from callback functions and thread entry routines. The correct way to fix this error is to add T_ENTRY to the declaration specifiers of the function. Do not simply cast the offending function pointer reference to the correct type. This eliminates the compile-time error but the generated code is not correct, and your program can fail in mysterious and hard-to-diagnose ways. If T_ENTRY was used, no source-level changes should be required.

Omitting T_ENTRY was tolerated with the SmartSockets Version 4.0 libraries because T_ENTRY was defined to `__cdecl`, which was the compiler default. With SmartSockets Version 5.x and higher, T_ENTRY is defined to `__stdcall`, and omitting it will cause a compile-time error.

Windows Registry

SmartSockets requires that certain configuration information be made available to the runtime libraries through the process environment. On UNIX platforms, this is typically accomplished by requiring SmartSockets users to manually add the information in their account profiles. While this same approach also works with the Win32 version of SmartSockets, this version has been extended to also allow the information to be extracted from the Windows registry.

The SmartSockets installation program creates this registry key when SmartSockets is installed on a Windows machine:

`HKLM\SOFTWARE\TIBCO Software Inc.\SmartSockets\6.8.0\Environment`

`HKLM` is a commonly used acronym for `HKEY_LOCAL_MACHINE`.

Registry values added to this key are seen by the SmartSockets runtime libraries just as if they had been added to the environment settings on Windows NT, 2000, or XP. Real environment variables can override the fallback settings provided in the registry.

The advantage of using the registry for this information is that it does not require editing individual account profiles or modifying the system profile using administrator privileges.

Command Prompt

The SmartSockets Command Prompt is provided as a convenience, in case you want to interact with SmartSockets using a console prompt. When selected, it starts a command interpreter with its current directory set to the SmartSockets installation directory and it automatically executes `ssvars32.bat` to set the interpreter's `RTHOME`, `RTARCH`, `PATH`, `INCLUDE` and `LIB` environment variables.

When the tutorial lessons tell you to complete a sequence of steps by typing commands into a command prompt, this is the simplest way to get a properly configured command interpreter. Advanced users who find themselves accessing the command prompt frequently can modify their environment settings to incorporate the environment variables from `ssvars32.bat`. For more information, see Path Settings on page 45.

You can invoke the SmartSockets Command Prompt at any time by selecting **Programs > SmartSockets > Command Prompt** from the Start menu.

Redirecting Program Output

SmartSockets provides a number of text output functions such as TutOut and TutWarning. When your program needs to output text, you can use one of these SmartSockets functions. This makes your text output portable to all the SmartSockets platforms.

These functions are designed for simple `printf`-style console output. However, most Windows applications have no `stdout` or `stderr` for use with `printf`. The SmartSockets Windows platform provides an alternative using TutWinSetOutputListBox. This function is passed the `HWND` of a list box control. It manages all text output by placing it in the provided list box. In addition, you can specify the number of window buffer lines to be maintained.

Creating a Project With Visual C++ Developer Studio 6.0

The first time you use SmartSockets with Visual C++, add the SmartSockets directories to the default search directories. This allows the SmartSockets headers and libraries to be found without requiring full path names. Follow these steps from within Developer Studio to add the SmartSockets directories to the default search directories:

1. Select **Tools > Options**.
2. Select the Directories Tab.
3. Choose Show directories for: Include files.
4. Add your SmartSockets include directory to the list:
   `C:\Program Files\Tibco\ss68\include`
5. Choose Show directories for: Library files.
6. Add your SmartSockets library directory to the list:
   `C:\Program Files\Tibco\ss68\lib\i86_w32`
7. Click OK.

Project Settings

Create a new project using **File > New Project**. Choose the project name, type and directory. Once the project is created, display the Project Settings dialog box. For more information, see the Visual C++ documentation.

Note the presence of multiple targets in the Settings For portion of the Project Settings dialog box.

Highlight only the Debug target and make these setting changes:

Dialog Tab	Category	Section	Setting	Notes
C/C++	Preprocessor	Preprocessor Definitions	• `WIN32` • `_DEBUG` • `_WINDOWS`	Use additional appropriate defines for your project type.
	Code Generation	Use Run-Time Library	• Debug Multithreaded DLL, or • Debug Multithreaded	Use the appropriate setting for your choice of libraries.
Link	General	Object/Library Modules	• `tutild.lib` • `tipcd.lib`	

Go back and highlight only the Release target and make these setting changes:

Dialog Tab	Category	Section	Setting	Notes
C/C++	Preprocessor	Preprocessor Definitions	• `WIN32` • `NDEBUG` • `_WINDOWS`	Use additional appropriate defines for your project type.
	Code Generation	Use Run-Time Library	• Multithreaded DLL, or • Multithreaded	Use the appropriate setting for your choice of libraries.
Link	General	Object/Library Modules	• `tutil.lib` • `tipc.lib`	

TIBCO Software Inc. End User License Agreement

READ THIS END USER LICENSE AGREEMENT CAREFULLY. BY DOWNLOADING OR INSTALLING THE SOFTWARE, YOU AGREE TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT AGREE TO THESE TERMS, DO NOT DOWNLOAD OR INSTALL THE SOFTWARE AND RETURN IT TO THE VENDOR FROM WHICH IT WAS PURCHASED.

Upon your acceptance as indicated above, the following shall govern your use of the Software except to the extent all or any portion of the Software (a) is subject to a separate written agreement, or (b) is provided by a third party under the terms set forth in an Addenda at the end of this Agreement, in which case the terms of such addenda shall control over inconsistent terms with regard to such portion(s).

License Grant. The Software is the property of TIBCO or its licensors and is protected by copyright and other laws. While TIBCO continues to own the Software, TIBCO hereby grants to Customer a limited, non-transferable, non-exclusive, license to use the number of Permitted Instances set forth in the Ordering Document, in machine-readable, object code form and solely for Customer's internal business use.

Restrictions. Customer agrees not to (a) make more copies than the number of Permitted Instances plus a reasonable number of backups; (b) provide access to the Software to anyone other than employees, contractors, or consultants of Customer; (c) sublicense, transfer, assign, distribute to any third party, pledge, lease, rent, or commercially share the Software or any of Customer's rights under this Agreement (for the purposes of the foregoing a change in control of Licensee is deemed to be an assignment); (d) use the Software for purposes of providing a service bureau, including, without limitation, providing third-party hosting, or third-party application integration or application service provider-type services, or any similar services; (e) use the Software in connection with ultrahazardous activities, or any activity for which failure of the Software might result in death or serious bodily injury to Customer or a third party; or (f) directly or indirectly, in whole or in part, modify, translate, reverse engineer, decrypt, decompile, disassemble, make error corrections to, create derivative works based on, or otherwise attempt to discover the source code or underlying ideas or algorithms of the Software.

Beta and Evaluation Licenses. Notwithstanding the foregoing, if the Software is being provided for demonstration, beta testing, or evaluation purposes, then Customer agrees (a) to use the Software solely for such purposes, (b) that the Software will not be used or deployed in a production environment, and (c) that such use shall automatically terminate upon the earlier of thirty days from the date Customer receives the right to install the Software, or Customer's receipt of notice of termination from TIBCO.

Technical Support. Provided Customer has paid applicable support fees (not included with Software fees unless separately listed), TIBCO shall provide support for generally available TIBCO Software on an annual basis commencing on the Purchase Date, as follows ("Support"): Customer shall designate at TIBCO's support website https://support.tibco.com/eSupport/newuser.html, the number of technical support contacts permitted under the level of Support purchased (contacts are changeable upon 48-hours prior written notice to TIBCO). Each contact may contact TIBCO for problem resolution during TIBCO's published support hours corresponding to the level of Support fees paid.

Upon notice from a contact of a Software problem which can be reproduced at a TIBCO support facility or via remote access to Customer's facility, TIBCO shall use reasonable efforts to correct or circumvent the problem according to its published support objectives. TIBCO reserves the right to make changes only to the most currently available version. TIBCO will use reasonable efforts to support the previously released version of the Software for a maximum of six months.

TIBCO shall have no obligation to support the Software (i) for use on any computer system running other than the operating system software for which the Software is approved (as set forth in the Software documentation) and licensed hereunder, or (ii) if Customer has modified or authorized a third party to modify the Software. TIBCO shall have no obligation to modify any version of the Software to run with any new versions of any operating system, or any other third party software or hardware. If Customer purchases Support for any Software, Customer must purchase the same level of Support for all copies of the Software for which it is licensed.

Support may be extended for one-year periods on the anniversary of each Purchase Date at the standard amounts set forth in its price list, for as long as TIBCO offers Support. Customer may reinstate lapsed support for any then currently supported Software by paying all Support fees in arrears and any applicable reinstatement fee. Upgrades, patches, enhancements, bug fixes, new versions and/or new releases of the Software provided from time to time under Support shall be used only as replacements to existing Permitted Instances, and shall not be deemed to increase that number, and use thereof shall be governed by the terms of this Agreement, except for the first paragraph of the Limited Warranty and any right of return or refund.

Consulting Services. Customer may request additional services ("Services") either in an Ordering Document, or by a separate mutually executed work order, statement of work or other work-request document incorporating this Agreement (each, a "Work Order"). Unless otherwise expressly agreed to in a Work Order, all Services and any work product therefrom shall be (a) performed on a time and materials basis, plus meals, lodging, travel, and other expenses reasonably incurred in connection therewith, (b) deemed accepted upon delivery, and (c) exclusively owned by TIBCO (except for confidential information of Customer identified to TIBCO in the Ordering Document), including all right, title and intellectual property or other right or interest therein. Each Work Order is intended to constitute an independent and distinct agreement of the parties, notwithstanding that each shall be construed to incorporate all applicable provisions of this Agreement. Specific to TIBCO training services, additional information regarding courses, registration, restrictions or limitation can be found at TIBCO's website at http://www.tibco.com/services/educational under Education Programs. Fees for Services shall be due and payable in United States dollars net 30 from the date of TIBCO's invoice.

Limited Warranty. If Customer obtained the Software directly from TIBCO, then TIBCO warrants that for a period of thirty (30) days from the Purchase Date: (i) the media on which the Software is furnished will be free of defects in materials and workmanship under normal use; and (ii) the Software will substantially conform to its published specifications. This limited warranty extends only to the original Customer hereunder. Customer's sole and exclusive remedy and the entire liability of TIBCO and its licensors under this limited warranty will be, at TIBCO's option, repair, replacement, or refund of the Software and applicable Support fees, in which event this Agreement shall terminate upon payment thereof.

This warranty does not apply to any Software which (a) is licensed for beta, evaluation, testing or demonstration purposes for which TIBCO does not receive a license fee, (b) has been altered or modified, except by TIBCO, (c) has not been installed, operated, repaired, or maintained in accordance with instructions supplied by TIBCO, (d) has been subjected to abnormal physical or electrical stress, misuse, negligence, or accident, or (e) is used in violation of any other term of this Agreement. Customer agrees to pay TIBCO for any Support or Services provided by TIBCO related to a breach of the foregoing on a time, materials, travel, lodging and other reasonable expenses basis. If Customer obtained the Software from a TIBCO reseller or distributor, the terms of any warranty shall be as provided by such reseller or distributor, and TIBCO provides Customer no warranty with respect to such Software.

EXCEPT AS SPECIFIED IN THIS LIMITED WARRANTY, THE SOFTWARE, SUPPORT AND SERVICES ARE PROVIDED "AS IS", ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS, AND WARRANTIES INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTY OR CONDITION OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, SATISFACTORY QUALITY OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE, ARE HEREBY EXCLUDED TO THE EXTENT ALLOWED BY APPLICABLE LAW. NO WARRANTY IS MADE REGARDING THE RESULTS OF ANY SOFTWARE, SUPPORT OR SERVICES OR THAT THE SOFTWARE WILL OPERATE WITHOUT ERRORS, PROBLEMS OR INTERRUPTIONS, OR THAT ERRORS OR BUGS IN THE SOFTWARE WILL BE CORRECTED, OR THAT THE SOFTWARE'S FUNCTIONALITY OR SERVICES WILL MEET CUSTOMER'S REQUIREMENTS. NO TIBCO DEALER, DISTRIBUTOR, AGENT OR EMPLOYEE IS AUTHORIZED TO MAKE ANY MODIFICATIONS, EXTENSIONS OR ADDITIONS TO THIS WARRANTY.

Indemnity. If Customer obtained the Software from TIBCO directly, then TIBCO shall indemnify Licensee from and against any final judgment by a court of competent jurisdiction, including reasonable attorneys' fees, that the unmodified TIBCO Software infringes any patent issued by the United States, Canada, Australia, Japan, or any member of the European Union, or any copyright, or any trade secret of a third party; provided that TIBCO is promptly notified in writing of such claim, TIBCO has the exclusive right to control such defense and/or settlement, and Licensee shall provide reasonable assistance (at TIBCO's expense) in the defense thereof. In no event shall Licensee settle any claim, action or proceeding without TIBCO's prior written approval. In the event of any such claim, litigation or threat thereof, TIBCO, at its sole option and expense, shall (a) procure for Licensee the right to continue to use the TIBCO Software or (b) replace or modify the TIBCO Software with functionally equivalent software. If such settlement or modification is not commercially reasonable (in the reasonable opinion of TIBCO), TIBCO may cancel this Agreement upon sixty days prior written notice to Licensee, and refund to Licensee the unamortized portion of the license fees paid to TIBCO by Licensee based on a five-year straight-line depreciation. This Section states the entire liability of TIBCO with respect to the infringement of any Intellectual Property rights, and Licensee hereby expressly waives any other liabilities or obligations of TIBCO with respect thereto. The foregoing indemnity shall not apply to the extent any infringement could have been avoided by use of the then-current release.

Limitation of Liability. EXCEPT AS PROVIDED UNDER INDEMNITY OR RESULTING FROM A BREACH OF CONFIDENTIALITY (THE "EXCLUDED MATTERS"), IN NO EVENT WILL EITHER PARTY OR TIBCO'S LICENSORS BE LIABLE FOR ANY LOST DATA, LOST REVENUE, LOST PROFITS, DAMAGE TO REPUTATION, BUSINESS INTERRUPTION, OR ANY OTHER INDIRECT, INCIDENTAL, CONSEQUENTIAL, SPECIAL, PUNITIVE, EXEMPLARY OR ANY SIMILAR TYPE DAMAGES ARISING OUT OF THIS AGREEMENT, THE USE OR THE INABILITY TO USE THE SOFTWARE, OR THE PROVISION OF ANY SUPPORT OR SERVICES, EVEN IF A PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. EXCEPT FOR THE EXCLUDED MATTERS, IN NO EVENT SHALL A PARTY BE LIABLE TO THE OTHER, WHETHER IN CONTRACT, TORT (INCLUDING ACTIVE OR PASSIVE NEGLIGENCE), BREACH OF WARRANTY, CLAIMS BY THIRD PARTIES OR OTHERWISE, EXCEED THE PRICE PAID BY CUSTOMER UNDER THE APPLICABLE ORDERING DOCUMENT.

THE FOREGOING LIMITATIONS SHALL APPLY EVEN IF THE ABOVE-STATED REMEDY OR LIMITED WARRANTY FAILS OF ITS ESSENTIAL PURPOSE. BECAUSE SOME STATES OR JURISDICTIONS DO NOT ALLOW LIMITATION OR EXCLUSION OF CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION MAY NOT APPLY TO CUSTOMER.

Confidentiality. "Confidential Information" means the terms of this Agreement; all information marked by the disclosing party as proprietary or confidential; any provided software, related documentation or related performance test results derived by Licensee; and any methods, concepts or processes utilized in provided software or related documentation. Confidential Information shall remain the sole property of the disclosing party and shall not be disclosed to any non-Authorized User without the prior written consent of the disclosing party. If Confidential Information is communicated orally, such communication shall be confirmed as "Confidential" in writing within thirty days of such disclosure. The parties agree to protect the Confidential Information of the other in the same manner it protects the confidentiality of similar information and data of its own (and at all times exercising at least a reasonable degree of care). Except with respect to the Software, items will not be deemed Confidential Information if (i) available to the public other than by a breach of an agreement with TIBCO, (ii) rightfully received from a third party not in breach of any obligation of confidentiality, (iii) independently developed by one party without use of the Confidential Information of the other; (iv) known to the recipient at the time of disclosure (other than under a separate confidentiality obligation); or (v) produced in compliance with applicable law or court order, provided the other party is given reasonable notice of the same. Both parties agree to indemnify the other for any damages the other may sustain resulting from their unauthorized use and/or disclosure of the other's Confidential Information. Such damages shall include reasonable expenses incurred in seeking both legal and equitable remedies. To the extent required by law, at Customer's request, TIBCO shall provide Customer with the interface information needed to achieve interoperability between the Software and another independently created program, on payment of TIBCO's applicable fee. Customer agrees to observe obligations of confidentiality with respect to such information.

Export. Software, including technical data, is subject to U.S. export control laws, including the U.S. Export Administration Act and its associated regulations, and may be subject to export or import regulations in other countries. Customer agrees to comply strictly with all such regulations and agrees to obtain all necessary licenses to export, re-export, or import Software.

Government Use. If the Customer is an agency, department, or other entity of the United States Government ("Government"), the use, duplication, reproduction, release, modification, disclosure or transfer of the Software, or any related documentation of any kind, including technical data or manuals, is restricted in accordance with Federal Acquisition Regulation ("FAR") 12.212 for civilian agencies and

Defense Federal Acquisition Regulation Supplement ("DFARS") 227.7202 for military agencies. The Software is commercial computer software and commercial computer software documentation. Use of the Software and related documentation by the Government is further restricted in accordance with the terms of this Agreement, and any modification thereto.

Orders. An Ordering Document shall be deemed accepted only by issuance of a TIBCO invoice and solely for purposes of administrative convenience. None of the terms of the Ordering Document (other than the Software product name, number of Permitted Instances, level of Support, description of Services, and fees due in connection therewith) shall apply for any reason or purpose whatsoever, regardless of any statement on any Ordering Document to the contrary, unless countersigned by an officer of TIBCO. This Agreement constitutes the entire agreement between the parties with respect to the use of the Software, Support and Services, and supersedes all proposals, oral or written, and all other representations, statements, negotiations and undertakings relating to the subject matter hereof. All orders of Software, Support or Services by Customer from TIBCO shall be deemed to occur under the terms of this Agreement (with or without reference to this Agreement), unless expressly superseded by a signed written Agreement between the parties. Software shall be delivered electronically, and such delivery shall occur when the TIBCO Software is made available for download by Customer. Physical deliveries (as applicable) of Software and documentation which typically accompanies the Software on delivery shall be on CD-ROM, FOB Palo Alto, and delivery shall occur by depositing the CD-ROM with TIBCO's overnight carrier (at no charge to Customer).

Term and Termination. Support or Services may be terminated: (a) by either party upon a default of the other, such default remaining uncured for fifteen days from written notice from the non-defaulting party; (b) upon the filing for bankruptcy or insolvency of the other party, (c) by either party upon prior written notice at least sixty days prior to the end of any annual Maintenance period; or (d) by Licensee (for Services), upon ten days prior written notice. Termination of Support or Services shall not terminate this Agreement. Customer may terminate this Agreement in its entirety at any time by destroying all copies of the Software. Upon termination of this Agreement in its entirety, for any reason, Customer must cease using and return or destroy all copies of the Software. Customer's obligation to pay accrued charges and any fees due as of the date of termination, as well as the sections entitled "Confidentiality", "Limited Warranty" and "Limitation of Liability" shall survive any such termination.

Authority. You hereby represent and warrant that you have full power and authority to accept the terms of this Agreement on behalf of Customer, and that Customer agrees to be bound by this Agreement.

General. Fees on the Ordering Document (all to be paid on the latter of thirty days from Invoice by TIBCO or the date set forth in the Ordering Document) do not include sales, use, withholding, value-added or similar taxes, and Customer agrees to pay the same, excluding therefrom taxes related to TIBCO's income and corporate franchise tax. Customer agree to pay all reasonable costs incurred (including reasonable attorneys' fees) in collecting past due amounts under this Agreement. Except as set forth in the Section entitled Limited "Warranty" all fees paid under or in connection with this Agreement are non-refundable and no right of set-off exists. All payments of fees due shall be made in U.S. dollars, net 30 from Purchase Date, or, for any other amounts coming due hereafter, net 30 from TIBCO's invoice. A service charge of one and one-half percent per month will be applied to all invoices that are not paid on time. Licensee agrees to pay all sales, use, value-added, withholding, excise and any other similar taxes or government charges, exclusive of TIBCO's income taxes. No delay in the performance of any obligation by either party, excepting all obligations to make payment, shall constitute a breach of this Agreement to the extent caused by force majeure. Customer hereby grants TIBCO and its independent auditors the right to audit Customer's compliance with this Agreement. If any portion of this Agreement is found to be void or unenforceable, the remaining provisions shall remain in full force and effect. This Agreement shall be governed by and construed in accordance with the laws of the State of California, United States of America, as if performed wholly within the state and without giving effect to the principles of conflict of law. The state and/or federal courts in San Francisco, California, shall have exclusive jurisdiction of any action arising out of or relating to this Agreement. The United Nations Convention on Contracts for the International Sale of Goods is excluded from application hereto. If any portion hereof is found to be void or unenforceable, the remaining provisions of this Agreement shall remain in full force and effect.

Definitions. In connection with this Agreement, the following capitalized terms shall have the following meaning: "Agreement" means this End User License Agreement; "Case Start" means the initiation of a single instance of a defined business process; "Connection" for the following TIBCO Software products shall mean: for TIBCO Enterprise Message Service, a TIBCO Enterprise Message Service client connection to the TIBCO Enterprise Message Service server for the purpose of sending or receiving messages, for TIBCO SmartSockets and TIBCO SmartMQ, any network protocol link established with such TIBCO Software (directly or indirectly) to any other entity, including but not limited to software, firmware or hardware, for TIBCO Enterprise RTView - Standard Monitor System, the number of monitored server instances to TIBCO Rendezvous daemons or TIBCO Hawk agents; for TIBCO Enterprise RTView- EMS Monitor System, a monitored TIBCO Enterprise Message Service Connection (as defined above for that product); for TIBCO General Interface, an electronic data interface to a CPU on a server (which excludes CPUs on devices such as routers, switches, proxies, or HTTP or application servers configured to substantially pass-through information or messages to TIBCO General Interface) that produces information or messages consumed by TIBCO General Interface; "Customer" means the original purchaser or licensee of the Software and any permitted successors and assigns; "Developer" means one user/developer of a TIBCO Software product for use in Development; "Development" means used for software development purposes only; "Enterprise" means an unlimited number of Permitted Instances for a period of one year from the Purchase Date (unless otherwise set forth in the Ordering Document), at which time existing licenses convert to perpetual and Customer may not thereafter deploy additional Permitted Instances, and in any event, shall (during the one-year unlimited deployment period) exclude any entity which acquires, is acquired by, merged into, or otherwise combined with Customer. Customer hereby agrees to provide TIBCO with notice of the number of Permitted Instances deployed at the end of such one-year period within thirty days thereafter; "Fab" means unlimited use for shop-floor manufacturing applications at a Site; "Workstation" shall mean a single end-user computer that is generally intended to be accessed by one person at a time; "Ordering Document" means any purchase order or similar document or agreement requesting Software, Support or Services; "Permitted Instance(s)" means the number of copies of Software running on a Server Instance, Workstation, User, or Development basis, on a designated Platform, as set forth in an Ordering Document, including, without limitation, Enterprise, Site and Fab licensing; "Platform" means the operating system set forth in an Ordering Document; "Purchase Date" means the date of the Ordering Document is accepted by TIBCO; "Server Instance" means a computer with 1 CPU (unless otherwise set forth in the Ordering Document) performing common services for multiple machines; "Site" means an unlimited number of Permitted Instances at a specific

physical address set forth in the Ordering Document (or, in the absence of any address, at Customer's corporate headquarters); "Software" means the software products listed in an Ordering Document (except as provided in the second paragraph hereof), in whole and in part, along with their associated documentation; "TIBCO" means TIBCO Software Inc.; and "Named User" means the number of named users with access to the Software.

Special Product Provisions. TIBCO BusinessPartner: Customer may sublicense to third parties ("Partners") up to the total Number of Copies of TIBCO BusinessPartner, provided that for every such sublicense, the Number of Copies Customer is licensed to use shall be reduced by the same number, and provided further that prior to delivery of TIBCO BusinessPartner to a Partner, such Partner agrees in writing (a) to be bound by terms and conditions at least as protective of TIBCO as the terms of this Agreement, (b) that TIBCO BusinessPartner be used solely to communicate with Customer's implementation of TIBCO BusinessConnect, and (c) for such Partner to direct all technical support and Maintenance questions directly to Customer. Customer agrees to keep records of the Partners to which it distributes TIBCO BusinessPartner, and to provide TIBCO the names thereof (with an address and contact name) within sixty days of the end of each quarter. Third Party Software: Use of any other third-party software identified by its company and/or product name or otherwise designated in Licensee's Ordering Document (collectively "Third Party Software") is subject solely to the terms and conditions of the click-wrap or shrink-wrap license agreement included with the Third Party Software products, and for which TIBCO shall be an intended third-party beneficiary of same. TIBCO shall have no obligation whatsoever in connection with the Third Party Software (including, without limitation, any obligation to provide maintenance or support) and the provision of Third Party Software is accomplished solely as an accommodation and in lieu of Customer purchasing a license to Third Party Software directly from the third party vendor. Embedded/Bundled Products. Some TIBCO Software embeds or bundles other TIBCO Software (e.g., TIBCO InConcert bundles TIBCO Rendezvous). Use of such embedded or bundled TIBCO Software is solely to enable the functionality of the TIBCO Software licensed on the Cover Page, and may not be used or accessed by any other TIBCO Software, or for any other purpose. Open Source Software: If Licensee uses Open Source software in conjunction with the TIBCO Software, Licensee must ensure that its use does not: (i) create, or purport to create, obligations of use with respect to the TIBCO Software; or (ii) grant, or purport to grant, to any third party any rights to or immunities under TIBCO's intellectual property or proprietary rights in the TIBCO Software. You also may not combine the TIBCO Software with programs licensed under the GNU General Public License ("GPL") in any manner that could cause, or could be interpreted or asserted to cause, the TIBCO Software or any modifications thereto to become subject to the terms of the GPL.

Version 5.2, 3/05

Third-Party Software Notices

SNMP 4.2.3

This product includes SNMP software which is copyrighted (c) 1989, 1991, 1992 by Carnegie Mellon University; Derivative Work - 1996, 1998-2000 copyright 1996, 1998-2000 The Regents of the University of California.

zlib 1.2.3

This product includes Zlib software, Copyright 1995-2003 Jean-loup Gailly and Mark Adler.

ADDENDA: Third-Party License Agreements

Third-Party Software License Agreements

The following are the software licenses for the Third-Party Software provided in connection with the software.

The Hoard License, Version 2.1.0

The license for Hoard is a modification of the GNU Lesser General Public license, given below, with the following additional clause:

****

As a special exception, if other files instantiate generics from this library, or you link this library with other files to produce an executable, this library does not by itself cause the resulting executable to be covered by the GNU Lesser General Public License. This exception does not however invalidate any other reasons why the executable file might be covered by the GNU Lesser General Public License.

****

GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

(The master copy of this license lives on the GNU website.)

Copyright (C) 1991, 1999 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

[2]TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

* a) The modified work must itself be a software library.

* b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

* c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

* d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.) These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library. In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

* a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

* b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

* c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

* d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

* e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

* a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

* b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

The SNMP License, Version 4.2.3

Various copyrights apply to this package, listed in 3 separate parts below. Please make sure that you read all the parts. Up until 2001, the project was based at UC Davis, and the first part covers all code written during this time. From 2001 onwards, the project has been based at SourceForge, and Networks Associates Technology, Inc hold the copyright on behalf of the wider Net-SNMP community, covering all derivative work done since then. An additional copyright section has been added as Part 3 below also under a BSD license for the work contributed by Cambridge Broadband Ltd. to the project since 2001.

Code has been contributed to this project by many people over the years it has been in development, and a full list of contributors can be found in the README file under the THANKS section.

---- Part 1: CMU/UCD copyright notice: (BSD like) -----

Copyright 1989, 1991, 1992 by Carnegie Mellon University
Derivative Work - 1996, 1998-2000
Copyright 1996, 1998-2000 The Regents of the University of California
All Rights Reserved

Permission to use, copy, modify and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of CMU and The Regents of the University of California not be used in advertising or publicity pertaining to distribution of the software without specific written permission.

CMU AND THE REGENTS OF THE UNIVERSITY OF CALIFORNIA DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL CMU OR THE REGENTS OF THE UNIVERSITY OF CALIFORNIA BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM THE LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

---- Part 2: Networks Associates Technology, Inc copyright notice (BSD) -----

Copyright (c) 2001-2003, Networks Associates Technology, Inc
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of the Networks Associates Technology, Inc nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---- Part 3: Cambridge Broadband Ltd. copyright notice (BSD) -----

Portions of this code are copyright (c) 2001-2003, Cambridge Broadband Ltd.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* The name of Cambridge Broadband Ltd. may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDER ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Index

Numerics

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W