CHAPTER 1

Getting Started

This module describes the platforms on which the SM C/C++ API can be used and how to install, compile, and start running the SM C/C++ API.

Platforms and Compilers

The SCMS SM C/C++ API was developed and tested on Windows, Solaris, and Linux platforms. It was compiled on Windows using Microsoft Visual C++ 6 compiler, on Solaris using the GCC 2.95.3 compiler, and on Linux using GCC 3.2.3 compiler.

Installation Package Contents

For brevity, the installation directory **sm-c-api-vvv.bb** is referred to as **<installdir>**, where **vvv** and **bb** stand for the SCMS SM C/C++ API version and build number.

The **<installdir>/winnt** folder contains the **smapi.dll** file, which is the Windows API Executable. It also contains additional DLL and LIB files necessary for the Windows API operation.

The **<installdir>/solaris** folder contains the **libsmapi.so** file, which is the Solaris API Executable.

The **<installdir>/linux** folder contains the **libsmapi.so** file, which is the Linux API Executable.

The **<installdir>/include** folder contains the API C/C++ header files.

The **<installdir>/include/system** folder contains the C++ API internal header files.

Table 1-1 Layout of Installation Directory

Folder	Subfolder (as applicable)	File Name
<installdir>		README.csmapi
	include	BasicTypes.h
		Common.h
		GeneralDefs.h

Table 1-1 Layout of Installation Directory (continued)

Folder	Subfolder (as applicable)	File Name
		Logger.h
		PrintLogger.h
		SmApiBlocking.h
		SmApiBlocking_c.h
		SmApiNonBlocking.h
		SmApiNonBlocking_c.h
	include/system	OperationHandleInterface.h
		OperationResultInterface.h
	linux	libsmapi.so
	solaris	libsmapi.so
	winnt	asn1ber.dll
		asn1ber.lib
		asn1rt.dll
		asn1rt.lib
		SmApi.dll
		SmApi.lib

How to Install the SCMS SM C/C++ API

The SCMS SM C/C++ API distribution is part of the SCMS SM-LEG distribution file and is located in the sm_api directory. The SCMS SM C/C++ API is packaged in a UNIX tar file. You can extract the SCMS SM C/C++ API using the UNIX tar utility or most Windows compression utilities.

- Installing the Distribution on a Unix Platform, page 1-2
- Installing the Distribution on a Windows Platform, page 1-3

Installing the Distribution on a Unix Platform

Step 1 Extract the SCMS SM-LEG distribution file.

Step 2 Locate the C/C++ SM API distribution file **sm-c-api-dist.tar**

Step 3 Extract the C/C++ API package tar.

```
#>tar -xvf sm-c-api-dist.tar
```

Installing the Distribution on a Windows Platform

Step 1 Extract the package using a zip extractor (such as WinZip)

How to Compile and Run the SCMS SM C/C++ API

The API connects to the PRPC server on the SM. For the API to work, the following conditions must be met:

- The SM must be up and running, and reachable from the machine that hosts the API.
- The PRPC server on the SM must be started.

The PRPC server is a proprietary RPC protocol designed by Cisco. For additional information, see the *Cisco Service Control Management Suite Subscriber Manager User Guide*.

- Compiling and Running a Program on Windows, page 1-3
- Compiling and Running a Program on Solaris, page 1-4
- Compiling and Running a Program on Linux, page 1-4

Note The C/C++ API should be compiled with 32-bit compiler. 64-bit API is currently not supported.

Compiling and Running a Program on Windows

Step 1 Ensure that **smapi.dll** and the other DLL files are in your path or in the directory of your executable.

Step 2 Ensure that the **include** folder is in your include path of the compilation.

Example using Microsoft Visual C++ 6:

Enter the project settings, click the **C++** tab, and then choose the **Preprocessor** category. Add the include directory path in the **Additional Include directories** line.

Step 3 Ensure that the smapi.lib file is in your linkage path.

Example using Microsoft Visual C++ 6:

Enter the project settings and click the **Link** tab. Add **smapi.lib** to the **Object\Library modules** line.

Step 4 Include the relevant API header file in your source code, and compile your code.

Compiling and Running a Program on Solaris

Step 1 Ensure that **libsmapi.so** is in your LD_LIBRARY_PATH.

For example, when using the **Bash** shell type, use the following command line:

```
bash-2.03$ export set LD_LIBRARY_PATH=$LD
LIBRARY_PATH:the-libsmapi.so-folder
```

Step 2 Ensure that the **include** folder is in your include path of the compilation.

For example, when using the **GCC,** add the include folder after the **-I** option flag, as follows:

```
>gcc -c -o TestSmApi.o -Ism-api-header-file-folder
-Ism-api-header-file-folder/system/ TestSmApi.cpp
```

Step 3 Ensure that the **libsmapi.so** file is in your linkage line or load it dynamically.

Link your object file to the *pthread* and socket library, as follows:

```
>gcc -o testSmApi TestSmApi.o -lsmapi -lpthread -lsocket
```

Compiling and Running a Program on Linux

Step 1 Ensure that **libsmapi.so** is in your LD_LIBRARY_PATH.

For example, when using the **Bash** shell type, use the following command line:

- Enter your password if prompted.

```
bash-2.03$ export set LD_LIBRARY_PATH=$LD
LIBRARY_PATH:the-libsmapi.so-folder
```

Step 2 Ensure that the **include** folder is in your include path of the compilation.

For example, when using the GCC, add the include folder after the -I option flag, as follows:

```
>gcc -c -o TestSmApi.o -Ism-api-header-file-folder
-Ism-api-header-file-folder/system/ TestSmApi.cpp
```

Step 3 Ensure that the **libsmapi.so** file is in your linkage line or load it dynamically.

Specify the location of **libsmapi.so** using the **-L** option flag. Link your object file to the pthread and stdc++ libraries, as follows:

```
>gcc -o testSmApi TestSmApi.o -lsmapi -lpthread -lstdc++ -L<lib path>
```