Welcome to SLX!

Dear SLX User:

We are pleased to welcome you to the growing community of SLX users. SLX users have tacked a wide variety of complex applications, utilizing the software in very creative ways. Their feedback has resulted in many improvements to SLX. SLX's capabilities for end-user customization, its scalable technology, and its tremendous model translation and execution speeds have proven it to be well suited for *difficult* applications. It has been used to develop simulation models of complex manufacturing, transportation, and telecommunications systems, and it has been used to simulate ports of entry into the U.S. to study infrastructure requirements for interdiction of drugs. In many of these applications, there are no off-the-shelf software packages available. Users have been able to extend and tailor SLX to meet their unique requirements.

The documentation for SLX provided by Wolverine at currently being updated and improved. In particular, Professor Thomas Schulze's **Simulation Needs SLX**, has been translated from German to English. At present, we are distributing draft copies of the Foreword, Chapter 1, and Chapter 2. Material from Chapter 2 is available in the form of online help. You can access this material in two ways. Clicking Help, Procedural SLX will bring up the table of contents for the on-line help system. Alternatively, you can right click on SLX keywords and click SLX Help in the popup menu that appears. Please note that at this time, only the procedural portions of SLX are included in the on-line help. In addition to this documentation, we are providing you with quite a bit of information and with a substantial collection of sample programs.

SLX is distributed on a "Products of Wolverine Software CD." If you simply insert the CD and follow the instructions, SLX will be installed on your system. Alternatively, you can download the Student/Demo version of SLX from the Downloads page of www.wolverinesoftware.com.

When you install SLX, a number of PDF documents will be created. To read these documents, you will need the Adobe Acrobat Reader (or a full version of Adobe Acrobat, if you have one.) You can download the Acrobat Reader free-of-charge from www.adobe.com.

The following files are included. You should read them in the order in which they are described below. Unless otherwise noted, filed are PDF documents.

File	Document Title	Contents
SLXFAQS	SLX FAQs	This document answers frequently asked questions about SLX.
TOUR30	30-Minute Guided Tour	This document takes you on a 30-minute introductory tour of SLX.
GSWSLX	Getting Started with SLX	This document presents a sequence of eight variations on a model of a networked laser printer. The variations of the problem are designed to reveal the fundamental architecture of SLX. You should this memo before you try to use the software.
SLXIDE	The SLX IDE	This document describes the toolbar and menus of the SLX interactive development for Windows.
SLXSAMP	Sample SLX Programs	This document gives thumbnail sketches of a large collection of sample SLX programs. These programs are in addition to the eight case studies included in "Getting Started with SLX."
SLXVSC	A Comparison of SLX and C	Since many of you are familiar with C, and much of SLX is based on C, we have provided a summary of some of the major similarities and differences between SLX and C.
SLXVSH	A Comparison of SLX and GPSS/H	Since some of you may be familiar with GPSS/H, we have included a document describing the similarities and differences between SLX and GPSS/H.
SLX\Doc\SLX2\ SLX2Overview	Introduction to SLX2	This document describes the object-oriented features added in SLX2.
SLX\Doc\SLX2\ SLX2Notes1	SLX2 Status & Discussion	This document contains notes and discussion about SLX2 features with SLX users.
SLX\Doc\SLX2\ SLX2Notes3	SLX 2.0 Keywords / Interpretations	This document provides concise descriptions of the keywords added to SLX2
SLX\Doc\SLX2\ Methods	Using SLX Methods	This document was written just before the development of SLX2 began. It illustrates the usage of SLX methods slightly prior to the

		actual advent of SLX2. It's a good starting point for users unfamiliar with object-oriented programming.
SLX\Doc\SLX3\ SLX3Notes	SLX3 Design Notes	This document describes the extensions made to SLX in SLX3.
STATS	SLX Statistical Features	This document describes SLX's built-in capabilities for generating random variates and for observing, collecting, and analyzing random outputs.
BATCH	Command-Line SLX	This document describes how to invoke SLX from a command line. Options are described for controlling output destination, whether messages are shown, etc. A "silent" option is available for suppressing default messages and embedding SLX in other applications.
SLXDLL	Calling DLL Functions from SLX	This document describes how to call DLL functions from SLX. DLLs are almost always generated using C/C++. Although in theory other languages such as Visual Basic could be used, to our knowledge no one has done so. The SLX compiler can automatically generate C/C++ ".h" files describing the SLX data to be passed to a C/C++ function. If you're using C/C++ functions to manipulate data stored in SLX objects, you must use an SLX-generated ".h" file, since SLX objects map into C/C++ structs in non-obvious ways. (SLX objects contain descriptive information that is hidden from C/C++ code.)
USERCPTS	Run-Time Rollback	The SLX debugger contains checkpoint/restore capabilities. Issuing a checkpoint command saves the complete status of an SLX program. Subsequently issuing a restore command restores the program's state. The primary use of checkpoint/restore is for locating obscure bugs in programs, using a divide-and-conquer approach. For example, if you know that your program has made an error at time 1000, you could rerun it to time 500, and if everything's OK at time 500, issue a checkpoint and run to time 750. If the error exists at time 750, you could rollback to time 500 without having to rerun the program, and run to time 600, etc. A number of years ago, we added run-time access to checkpoint/restore, making it possible for a program to save checkpoints and perform restores *under program control*. There are several potential uses of this capability. First, it makes it easy to conduct multiple experiments, each of which is the result of running forward from a saved state. Second (an extension of the previous use), it makes it easy to examine thousands of scenarios in the context of performing optimizations. Third, it makes it possible to incorporate SLX models into distributed simulations that employ local time management with rollback, e.g. time-warp.
SLX97	An Introduction to SLX	This paper was presented at the 1997 Winter Simulation Conference.
SLX98	An Introduction to SLX	This paper was presented at the 1998 Winter Simulation Conference.
SLX99	An Introduction to SLX	This paper was presented at the 1999 Winter Simulation Conference.

The easiest way to communicate with us is via e-mail (mail@wolverinesoftware.com). If you have questions or problems send us an e-mail, and attach the files necessary to demonstrate the question/problem. Please do not paste source code fragments into an e-mail. Line-wrapping and other formatting issues can introduce a whole host of problems with pasted input.

Updates to the commercial versions of SLX are available through the Updates page of www.wolverinesoftware.com.

The latest version of the Student/Demo version of SLX can be downloaded from the Downloads page of www.wolverinesoftware.com. (We do not post incremental updates for the Student/Demo version.)

Once again, welcome aboard!

Jim Henriksen