A Roadmap to the Shore Releases[1]

The Shore Project Group
Computer Sciences Department
UW-Madison
Madison, WI
Version 1.1.1

October 27, 1997

[1]This research is sponsored by the Advanced Research Project Agency, ARPA order number 018 (formerly 8230), monitored by the U.S. Army Research Laboratory under contract DAAB07-92-C-Q508.

Contents

1 Introduction

This document briefly describes the contents of the directories in the various Shore releases. All pathnames are relative to the root of a shore installation, referred to as `SHROOT` in this document.

2 Binaries and Libraries (`include`, `lib`, and `bin`)

These three directories are at the root of the binary releases. They are also created when an installation is compiled from the source release. When compiled from a source release, they become sub-directories of `SHROOT/installed` unless you configure the source release to install them somewhere else (see the *Shore Software Installation Manual* for details).

`include` C++ header files.

`lib` runtime libraries.

`bin` executable programs.

3 Documentation and Examples (`docs.ps`, `docs.html`, and `examples`)

These three directories are at the root of a documentation release.

`docs.ps` document and manual pages as Postscript documents suitable for printing.

`docs.ps` document and manual pages as html (hypertext markup language) pages suitable for viewing with a web browser such as Mosaic or Netscape.

`examples` a variety of example programs, described more fully below. Users at your installation who wish to learn more about Shore may wish to copy one or more of the sub-directories of `SHROOT/installed/examples` to their own workspace to compile them. The examples are designed to be usable in conjunction with a binary release (a source release is not required).

4 Configuration management (`config, tools`)

These two directories are at the root of a source releases.

`tools` A variety of executable shell and Perl scripts uses for configuring and building Shore from sources. Particularly noteworthy is `makemake`, a Perl script for creating Makefiles from Imakefiles.

`config` files used by `tools/makemake` to generate Makefiles. Particularly noteworthy is `config.h`, which is used both to control the actions of `makemake` and as a header file included in virtually all C and C++ source files.

5 Sources (`src`)

The directory `SHROOT/src` is present only in a source release. It contains the source code for Shore. The subdirectories of `src` are presented here in the order of they are normally visited when Shore is compiled, which is more or less "bottom-up." The code in a later directory may depend on code in directories described earlier.

5.1 Foundation Classes (`src/fc`)

The `SHROOT/src/fc` directory contains what we call "foundation" classes. These C++ classes are meant to be easy to use in software systems other than Shore. They do not depend on any other Shore code. Included are classes for such things as lists, hash tables and error handling. Most things in this directory have the prefix `w_`. Nobody knows what "w" stands for.

5.2 Thread Package (src/sthread)

The SHROOT/sthread directory contains a thread package. It also contains source code for the diskrw program, which helps the Shore server do asynchronous I/O. The server forks a copy of diskrw for each volume it is managing. When it wants to do I/O, it notifies the diskrw process. The server and the diskrw process communicate data through shared memory and synchronize with TCP messages or signals.

5.3 Common Code (src/common)

The SHROOT/common directory contains code needed by most layers of Shore. In this directory are the interfaces and implementations for data types from which Shore's persistent data structures are built. Included are implementations of object identifiers, thread-safe hash tables and options.

5.4 Storage Manager (src/sm)

The SHROOT/sm directory contains code for the Shore Storage Manager (SSM). The interface for the SSM consists of the public interfaces of the classes ssm, pin_i and scan_*xxx*_i classes defined in sm.h, pin.h and scan.h, respectively.

5.5 RPC Package (src/rpc4.0)

The SHROOT/rpc4.0 directory contains a modified copy of Sun Microsystems' "Portable ONC/NFS", a public-domain implementation of the Sun RPC and XDR protocols.

We have made minor changes for its use in Shore. We use only a portion of the package. The complete package is included here, but we support only the portion that Shore uses (SHROOT/rpc4.0/rpc and SHROOT/rpc4.0/rpcgen), to the extent that the Shore Value-Added Server uses it. (We do not provide general support for it, and we have no manual pages for it.) On the Solaris platform, only SHROOT/rpc4.0/rpcgen is used.

5.6 Shore Value-Added Server (src/vas)

The SHROOT/vas directory contains code for the Shore Value-Added Server (SVAS). It has the following subdirectories.

include The include files needed by software that uses the SVAS.

client The SVAS client library, which contains class implementations meant to linked with application code to create a Shore client process. The client library caches some information for local processing, but most SVAS methods are remote operations that are performed by the server.

tests A simple program for testing the SVAS client library.

scripts A set of Tcl scripts for testing the SVAS. *There is no "support" for these scripts.* The testing mechanism for the SVAS is in transition. Not all of the scripts work; some use obsolete syntax for commands.

`common` Code that is used in both the server and the tester.

`server` The Shore Value-Added Server (SVAS). Running `make` in this directory creates a complete SVAS server in the form of an executable file named `shore`.

5.7 Language Independent Library (`src/lil`)

The `SHROOT/lil` directory is for libraries that provide programming-language-independent functionality above the SVAS. Currently, this directory contains only the subdirectory `oc` and an Imakefile with instructions for assembling the `libshore.a` and `libserver.a` libraries from object files created in other directories.

The `SHROOT/lil/oc` directory contains code for the object cache that is linked with applications. This code is responsible for managing a fixed-size LRU (least-recently used) cache of objects, for swizzling and unswizzling references, and for for flushing objects back to the server on transaction commit and on buffer overflow.

5.8 Shore Data Language Compiler (`src/sdl`)

The `SHROOT/sdl` directory contains source for the Shore Data Language (SDL). It creates the executable program `sdl`, which compiles SDL specifications into metatype objects stored in the database and C++ header files and implementation stubs to support support these types. This directory also contains a library of routines that support the implementation of generated types by communicating with the object cache layer.

5.9 Utilities (`src/util`)

The `util` directory holds miscellaneous Shore utilities.

The only such program in the current release is is the *mount* utility, for mounting the Shore file system as an NFS file system The directory `SHROOT/src/util/mount/generic` contains code for the `smount` and `sumount` utilities, which provide only the basic functionality of the standard `mount` and `umount` utilities, but with an extension required for Shore. See the `README` file in that directory for more details. These programs are not used on the Linux platform, where the standard `mount` untility already has the required extension.

6 Example Applications (`installed/examples`)

The `SHROOT/installed/examples` directory contains a few simple example application programs. It is only included in a documentation release. To compile and run these examples, you will need either a binary release, or the results of compiling a source release.

`shls` is a very simple program that lists the contents of a Shore directory. See the tutorial Scanning Directories in a Shore Application for more details.

`pscan` is a very simple program that lists the contents of a Shore pool. See the tutorial Scanning Pools in a Shore Application for more details.

`stree` contains source code for the examples used in the tutorial Getting Started with Shore.

`vas` contains two sub-directories with examples of value-added servers other than the Shore value-added server (SVAS). The directory `SHROOT/installed/examples/vas/hello` contains a "minimal" value-added server.

The directory `SHROOT/installed/examples/vas/grid` contains a more substantial value added server. It is described in the tutorial Writing Value-Added Servers with the Shore Storage Manager.

`unixfile` contains several programs for manipulating so-called "Unix files," which are Shore registered objects that mimic the behavior of Unix files.

`oo7` contains an implementation of the OO7 Benchmark.