Soar 7.1

Randolph M. Jones, Chris Waterson, and Karen Coulter

June 28, 1997

Problems With Soar 7.0

- Not easy to install
- Even trickier if you do not create soar-specific versions of Tcl and Tk
- Different versions for different operating systems
- The most recent versions of Tcl and Tk are different enough (internally) that it would take significant effort to upgrade to them.

What Soar 7.1 Addresses

The Soar kernel does not change (except for bug fixes).

No changes will be required in your productions.

Move from Tcl 7.4 and Tk 4.0 to Tcl 7.6 and Tk 4.2 (at least).

The procedure for starting Soar will change slightly.

There will be a few changes to the command-line interface, mostly to do with manipulating multiple agents.

Existing simulation and graphical interfaces will likely require some modifications for communicating with Soar agent interpreters.

The General Idea

The most recent versions of Tcl and Tk support the use of "dynamically loaded extensions" and "packages".

- This allows us to delineate clearly which parts of Soar are Tcl and Tk, which parts are the Soar kernel, and which parts are the interface between the kernel and Tcl.
- This in turn allows users to incorporate the "Soar package" into whatever version of Tcl they wish (version 7.6 or later), or to load other Tcl packages with Soar (i.e., future Tcl upgrades will be easy).

The compilation of Tcl, Tk, and Soar can take place independently.

- If you already have Tcl 7.6 (or greater) and Tk 4.2 (or greater) on your system, you can use them to load the Soar package.
- Otherwise you can build Tcl and Tk separately before you build Soar.

Making Installation Easier

There are (at least) two reasons why installing Soar 7.0 is difficult.

- It is often difficult to figure out the correct configuration of compiler flags, etc., for each particular system.
- It is not obvious what all the pieces are and where they need to be.

Making Installation Easier

It turns out that the configuration and installation routines for Tcl and Tk are relatively automated. In addition, these installation procedures save their configuration parameters in specific files.

Because Soar 7.1 will assume independently built versions of Tcl and Tk, the installation procedure for Soar can use all of this configuration information, and the build will (likely) go smoothly.

- It took me less than two hours to install Tcl 7.6, Tk 4.2, and Soar 7.1 on the Ohio State Hewlett Packard machines, including some bug fixes in the installation scripts.

In addition, we can use the existing Tcl and Tk shells to create a friendly user interface for building Soar.

The Installation Dream

- Start the version of tclsh (or wish, or whatever) that you will be using with Soar.
- Load in the "build-soar.tcl" file.
- Click a few buttons on a graphical user interface, and change some configuration entries if necessary.
- Click the "build" button.
- Click the "install" button.
- OR, for some platforms (like Windows and Macintosh), simply download and run a self-extracting binary distribution.

Supporting Multiple Operating Systems

The biggest headache in adapting Soar 7.0 to different operating systems was working with different command and graphical interfaces.

The most recent versions of Tcl and Tk provide strong support for cross-platform compatibility for Unix, Windows, and Macintosh (including platform-independent file manipulation).

Building Soar 7.1 as an "optional Tcl package" takes full advantage of the cross-platform capabilities. Making Soar available on these different operating systems becomes trivial, because Tcl and Tk do all the work of talking to the operating system command interfaces.

The Tcl/Tk Soar Interface (next talk) will also make full use of the cross-platform capabilities, providing a uniform graphical interface across platforms (but with native "look and feel" on each platform).

How Things Work

We will build on the model used for Tcl and Tk

When you build Tcl (or use a Tcl binary distribution), you get three basic pieces:

- An "object library" that contains all of the compiled Tcl code.
- A stand-alone executable (`tclsh` on Unix) that provides a minimal command-line interface to the object library.
- A "Tcl library" containing specifications of various Tcl commands.

How Things Work

When you build Tk, you get:

- An "object library" that contains all of the compiled Tcl code.
  - This object library can be used as part of a "Tk package", to be loaded dynamically if your operating system supports dynamic linking.
- A stand-alone executable (`wish` on Unix), that is basically the static equivalent of running "tclsh" and then loading the "Tk package".
- A "Tk library" containing specifications of various Tk commands and widgets.

How Things Will Work

When you build Soar, you will get:

- An "object library" that contains all of the compiled Soar kernel code.
- Another "object library" that includes the Soar kernel code plus the code to interface the kernel with Tcl.
  - As with Tk, this object library can be used as part of the "Soar package", which can be loaded dynamically into any Tcl interpreter if your operating system supports dynamic linking.
- A "Soar library" containing the specifications of some Soar interface commands, the default rules, the Soar aliases, demo programs, the Tcl/Tk Soar Interface, etc.

How Things Will Work

You will also have the option of building standalone executables for:

- `soar`: The Tcl and Soar object libraries statically linked together
- `soartk`: The Tcl, Tk, and Soar object libraries statically linked together.

How Start-Up Will Change

Whether you start with `tclsh` or `soar` (or `wish` or `soartk`), things will initially be exactly the same. That is, `soar` starts you in a Tcl Shell with no Soar commands available. This is for the sake of uniformity with the "dynamically linked" versions of Soar.
This also means there will be no special Soar command-line options, and no special Soar start-up files, because as far as the system is concerned, you are not really "using Soar" yet.
To turn your Tcl interpreter into a "Soar interpreter", you must issue the command `package require Soar`.

- In the static version of Soar, this will automatically "load" the Soar package.
- In the dynamic version, the system must have compiled in the location of the dynamic object library (or use an environment variable).

How Start-Up Will Change

When you load the Soar package, the system will automatically load in the file `.soarrc`, but there will be no automatic loading of agent-specific start-up files.

Once you have loaded the Soar package, the system should act very much like Soar 7.0. . .except if you are using multiple agents!

Note: You will also be able to load any other Tcl packages you need or desire.

Multiple Agents in Soar 7.1

The facility for using multiple agents in Soar 7.0 is "home brewed", and works well enough, but contains a few bugs and problems.

The recent versions of Tcl have introduced built-in support for multiple interpreters, so it makes sense to move to this paradigm (doing so increases the independence between Tcl and Soar).

There are still some issues to be worked out, because Tcl allows arbitrary "trees" of interpreters, organized in a master-slave relationship (the primary motivation here appears to be software security).

This is not necessarily the most logical structure for Soar agents, although it does seem to make sense for Soar interpreters to be "slaves" of a simulation or graphical-interface interpreter (this is how the Tcl/Tk Soar Interface arranges things).

Changes in Commands for Multiple Agents

Because we are adopting Tcl's scheme for multiple interpreters, Soar 7.1 will do away with Soar 7.0's multiple-agent commands (`create-interpreter`, `select-interpreter`, `destroy-interpreter`).

Unfortunately, making use of multiple agents will require a relatively thorough understanding of the Tcl interpreter commands and interactions (but only for interface and simulation writers, not for the "vanilla Soar user").

Miscellaneous Information

Chris Waterson wrote the prototype version of Soar 7.1. This version is available at:

`http://ai.eecs.umich.edu/people/waterson/research/soar71/`

Randolph Jones has cleaned up the prototype a bit, and is working on creating the Friendly Soar Installation Interface.

Karen Coulter and Randolph Jones will incorporate the latest bug fixes from Soar 7.0, to come up with the final product.

The current prototype will be used at the AAAI Soar tutorial on machines running Unix, Windows, and the Macintosh operating system (see the demo tonight).

Hopefully, Soar 7.1.0 will be "released" by the end of the Summer.