CLIM 1.0 Release Notes and Installation Guide

Overview of CLIM

CLIM, the Common Lisp Interface Manager, is a portable high-level object-oriented user-interface management system. It provides a full spectrum of capabilities for describing the user interface of an application and implementing that user interface on a variety of computers and window systems.

CLIM is portable in several respects. It is written in a standard dialect of Common Lisp and CLOS, and is therefore executable by numerous computers provided by various vendors. Also, user interfaces described by CLIM are independent of any particular window system or output device, so that CLIM applications are readily portable to different hardware and software platforms.

CLIM provides high level facilities for describing a user interface. Complex user interface behaviors such as incremental redisplay, formatting textual and graphical output, context sensitive documentation, direct manipulation, and mixed keyboard-mouse interaction are easily described with a minimum of specialized code.

CLIM is object-oriented in several respects. It is implemented in Common Lisp Object System (CLOS), and uses object-oriented techniques to great advantage internally. More importantly, CLIM provides an object-oriented model for integrating an application with its user interface. This *presentation model* associates user interface behavior directly with application objects, which allows an application to describe its user interface in terms of its own semantics.

CLIM provides the following facilities:

- *Windows* — Convenient facilities for creating, placing, and manipulating windows, including managing margins and scroll bars.

- *Streams* — Stream-oriented, device-independent input and output of arbitrarily mixed text, graphics, and presentations.

- *Graphics* — A rich graphics model which includes a variety of common geometric shapes (such as lines, rectangles, polygons, and ellipses), drawing options (such as line thickness and joint shape), a sophisticated inking model for describing patterns and color, and full affine coordinate transformations (translation, rotation, scaling).

- *Styled text output* — The appearance of textual output (font family, typeface, and size) is specified with an abstract, device independent mechanism called text style.

- *Output recording* — A facility for capturing all output done to a window, which provides the basis for arbitrarily scrollable and redisplayable windows.

- *Presentations* — Presentations associate user interface behavior with application objects, using object-oriented programming technniques. The user interface of an

application may be described in terms of its own semantics, using the high level language of presentation types, instead of the lower level language of keystrokes, mouse events, and widgets.

- *Menus and Dialogs* — Many types of menus and dialogs may be automatically generated, using the presentation type facility to describe the desired appearance and behavior.

- *Context-sensitive input* — An application accepts direction from the user by establishing a context in which certain classes of operations and operands are valid, using presentation types. The user interface system uses this context to provide appropriate documentation and feedback to assist the user, and assures the application that user-supplied values are appropriate.

- *Commands* — The user interface operations of an application are described by commands, which operate on presentation types. This uniform mechanism is used for all interaction styles, including direct manipulation, menus, dialogs, keystroke accelerators, and command lines.

- *Formatted output* — High-level macros allow applications to produce neatly formatted tabular and graphical displays with little additional programming.

- *Incremental Redisplay* — Recorded output may be changed and the display incrementally and efficiently updated, without extensive programming.

- *Application frames* — The screen layout and top level behavior of an application are described by application frames.

For detailed user and reference documentation about CLIM, see the document *Common Lisp Interface Manager (CLIM): Release 1.0.*

CLIM incorporates refined versions of many features and concepts from Symbolics Dynamic Windows. See the section "Comparing and Contrasting DW and CLIM" for a description of their similarities and differences. There is a facility to assist in the task of converting Dynamic Windows software to CLIM, see the section "Converting from DW to CLIM". Note that conversion of existing Genera software to CLIM is wholly optional, and that Dynamic Windows will continue to be supported.

The CLIM Standard

The Common Lisp Interface Manager was defined and developed by a consortium of cooperating Lisp vendors, including Franz, International Lisp Associates, Lucid, Symbolics, and Xerox PARC. We consider CLIM a *de facto* standard, though it may be proposed for official standardization after the community has gained experience using it.

Symbolics provides versions of CLIM for Symbolics computers running Genera and IBM-compatible personal computers running CLOE. For other platforms, compatible versions of CLIM are available from the vendors listed above among others.

Contact your Lisp vendor for availability information, or International Lisp Associates if your vendor has no current plans to offer CLIM.

Symbolics CLIM 1.0, and the CLIM implementations available in 1991 from other vendors, are based on a reference implementation of CLIM called CLIM Version 1. Throughout 1991, the CLIM consortium will be working on a new reference implementation which will incorporate the Silica technology developed at Xerox PARC. Implementations derived from CLIM Version 2 will be compatible with CLIM Version 1, but will provide additional functionality and better performance by integrating directly with popular window toolkits such as OSF/Motif, Open Look, and Microsoft Windows.

CLIM 1.0 Implementation Notes

This section summarizes some known limitations of the Symbolics implementation of CLIM 1.0, particularly with regard to rendering of graphic designs. Note that other implementations of CLIM 1.0 may have different limitations.

- The Genera debugger does not work on CLIM streams. If a CLIM application gets an error of any sort, the debugger will be invoked in a background window, and a notification issued to that effect. You may find it useful to tailor the behavior of these notifications, see the section "Pop-up notifications".

- **clim::make-contrasting-inks** provides up to 8 different colors or patterns of ink.

- **clim::make-contrasting-dash-patterns** provides up to 16 different dash patterns.

- CLIM 1.0 does not support composite designs. Also, CLIM 1.0 provides very limited support for opacity: opacity inks are supported, but are interpreted for rendering as either fully transparent or fully opaque.

- CLIM 1.0 contains limited support for patterned designs. The elements of a patterned design (whether a pattern, stencil, or tile) must be a color or opacity, and not a general design (a shape). Some previous versions of the documentation have referred to more stringent limitations; these limitations are no longer present.

- CLIM 1.0 does not support nonrectangular clipping regions. A clipping region is interpreted as the bounding rectangle of the region supplied.

- Some CLIM 1.0 implementations may not support tilted ellipses (ellipses not aligned with the X or Y axis). The Genera implementation does, but at some performance penalty when using a window system such as X or Macintosh which doesn't support tilted ellipses directly.

- PostScript streams do not yet create Encapsulated PostScript files. Their output can be printed on an Apple Laser Writer.

- For PostScript streams, line styles which specify **:line-unit :normal** and a **:line-thickness** other than 1 may not produce the desired effect.

- **filling-output** doesn't work very well in the face of nontextual output (presentations and graphics).

- In Genera, there is no global user interface for changing the foreground and background colors of a CLIM application, they are under program control only. Using the Window Attributes menu to change the drawing and erasing alu functions, or `FUNCTION C` to change to inverse video, will not affect the foreground and background colors of a CLIM window. (At present, you must refresh the CLIM window to restore proper appearance after such a change.)

Installing CLIM in Genera 8.0

For Genera 8.0, CLIM is distributed as a layered system on a single distribution tape, which contains loadable binaries for all 3600 and Ivory computers, all CLIM source code, online CLIM documentation, and all necessary Genera patches since the Genera 8.0.1 software ECO. CLIM may be installed on any Symbolics computer running Genera 8.0.1 or later, using the following procedure.

1. Confirm that Genera 8.0.1 or Genera 8.0.2 is installed.

   ```
   Show Herald
   ```

2. Restore the CLIM distribution tape, and enter a tape spec at the prompt.

   ```
   Restore Distribution
   Enter a tape spec [default Local: Cart]:
   ```

3. Load all Genera patches up to (and beyond) Genera 8.0.2.

   ```
   Load Patches
   ```

   At this point, you should be up to System patch level 425.140. Confirm this with `Show Herald`. Do not attempt to load CLIM into a system that is not up to patch level.

4. Load the CLIM system.

   ```
   Load System (a system [default System]) CLIM
   ```

5. Load the CLIM online documentation.

   ```
   Load System (a system [default CLIM]) CLIM-Doc
   ```