3J−1

A Requirement Analysis for A Portable Window System on Top of Common Lisp

Masayuki Ida†1, Takashi Kosaka†2, Keisuke Tanaka†1, Katsuhiko Yuura†3, Eiji Shiota†4, Haruyuki Kawabe†5, Atsushi Atarashi†6, Keiji Hashimoto†7, Noritoshi Rokujo†8

Acknowledgement

The main framework of this research was born in the discussions under the 1988 Jeida Common Lisp Committee Secretariat meetings. The authors would like to express their gratitude to all the members of JCLC.

1 Background 1. – Needs for a window system in Common Lisp –

There are several Window Systems and UIMS for various Common Lisp implementations. But they have different functionalities. With this reason, though the portability of Common Lisp is proven, lots of interactive and/or graphical applications can not be ported to different implementations easily.

From 1984 on, there have been several Common Lisp based standardization efforts including window systems. In the beginning, the Common Windows of Intellicorp was one of the candidate for standard but was not adopted. At that time, Symbolics Lisp machine did provide the Dynamic Window which features the presentation system. Common Windows itself grew and several dialects of it were introduced.

We made surveys and got a sketch as follows:

1) Trial to integrate several existing window systems into one is required. It should be independent from any existing windows but should have a bridge from them.

2) Consultation with the current technology is required. At least, X-window and it's friends, Common Windows, Genera, and PC-based windows should be investigated.

3) Public acceptance of the superiority of Dynamic Windows feature should be checked.

4) Good environment needs large memory. Is it affordable ? Say, a full function programming environment may occupy 20 M bytes or more.

5) As a workstation, needs much more functionality on Japanese character handling in every text handling.

2 Background 2. – CLOS and object oriented approach –

Recently it is quite common to integrate a windowing stuffs using object oriented idea. This trend has a firm technical background. Smalltalk is one example. In the Lisp world, Flavors is used as a kernel technology for Symbolics Genera Window system.

Recent development of CLOS (Common Lisp Object System), and its adoption as a part of X3J13 Common Lisp, place Common Lisp among the family of object oriented languages. Since CLOS (chapter 1 and 2) was established in 1988, there are not so much experience reported.

Lots of existing window systems for Common Lisp lack of object oriented approach since they were born before CLOS was invented. To CLOSify a portable window system is the must.

3 Requirement Issues and their analysis

3.1 Issue 1. Single Process or Multi Process

▷ discussions: Related questions are whether each window is assigned an independent process or is assigned to a window of native window system or no.

▷ conclusions:

1) Place a root window at initialization. Each window is a child of it.

2) Each window can be assigned a process.

3) Keyboard and Mouse events should be correctly and promptly handled.

4) "Pure Multi Process" is not needed. (Each window is not needed to be independently executed)

3.2 Issue 2. Object-Oriented Style

▷ discussions: CLOS is the must from the background requirement.

▷ conclusion:

1) User interface and window handling should be CLOS based both.

2) Window can be dynamically defined.

3) Try to evaluate the CLOS power. Try to check whether Meta Object Protocol give us a solution or not.

4) Migration path from other object oriented windowing tools is needed.

3.3 Issue 3. What is the displayed output.

▷ discussions: We discussed about what is the major advantage of Genera window system. One thing to prove is whether presentation system is affordable. In Genera, output is recorded as much memory as exists. (user can clear the history freely). Is infinite recording really necessary considering the space consumption and redisplay speed. Should all the object

†1 Aoyama Gakuin University, †2 Aoyama Gakuin University/CSK Corp., †3 Hitachi Ltd., †4 Nihon Symbolics Corp. †5 Nihon Unisys Ltd. †6 NEC Corp. †7 CEC Ltd. †8 Fujitsu Ltd.

be mouse sensitive items? Context dependent mouse sensitivity is affordable or no.

▷ conclusions:

1) Must install output recording mechanism. Should provide a constant which has the maximum numbers of lines/items. (we all agree with the output recording feature is useful though it is heavy. need some purging mechanism)

2) Presentation type is very important. Prepare the facility to get user defined presentation type. A new definition using CLOS is introduced.

3) sensitivity control considering the mouse speed should be needed.

3.4 Issue 4. Font and Multi national character presentation

▷ discussions: Japanese characters should be handled. Can display several fonts in the window or no.

▷ conclusion:

1) Provide multi font ability, though font itself is implementation dependent. The mechanism for it can be standardized.

2) Must consider the provision for dynamic adjustment of displayed characters with various fonts.

3.5 Issue 5. Widgets or Accessories

▷ discussions: Lots of window systems have lots of widgets/gadgets and independent styles.

▷ conclusion:

1) Too much consideration for style guiding is not needed.

2) Coordination with Existing Styles or the styles of native window systems.

3.6 Issue 6. Input Editor

▷ discussions: We discussed about the needs for input fron-end processor (in a broader sense), Input Editing and Japanese character input mechanism as an example. Wnn (egg), or commercial fron-end package, can co-exist with our new window.

▷ conclusion: There must be a common way to access the stream internal buffer to alternate the character already inputted to that input stream.

3.7 Issue 7. Graphics Functions

▷ discussions: Rich functionality with proper Graphics display model. 3D model is needed ? Color handling is needed ?

▷ conclusion: The Lisp window system we are thinking doesn't provide high level graphic tools, but a typical level ones. Must provide color capability.

3.8 Issue 8. Implementation Model

▷ discussions: Is network oriented implementation needed ?

▷ conclusion: We don't need to specify the model details, since the target machines have much variety.

3.9 Issue 9. A new window system or a tool kit on top of other windows

▷ discussions: Creating a new window system is attractive solution for the above. On the other hand, if we start to develop a new window system from scratch, we might need to develop every codes which should cope with the same functionality as other window will have in every minutes.

We want to share the progress of window technology as a user of other window systems.

▷ conclusion: We will design and provide a window tool kit or user interface management system on top of various general purpose window systems.

4 Conclusion – *Yy Window Tool Kit* –

Yy Window Tool Kit is the goal of our analysis and is an output of our research works. As a summary, *Yy* is a CLOS based window tool kit with our scheme of output recording. We design the *Yy* as a three layered tool kit.

There are three layers: NWSI, YYWS, and YYAPI.

NWSI: Native Window System Interface Module

YYWS: *Yy* window system

YYAPI: *Yy* Application Program Interface

NWSI depends on a native window system. The NWSI for the *YyonX* is X-window dependent module. YYWS is a native window independent window system. The interface between YYWS and NWSI is defined to be independent from various underlying window systems as possible. YYAPI[1] is an Application Program Interface and is used by the users.

We already succeeded to implement a pilot of ***Yy*** on top of X-window, which is called ***YyonX*** [2]. Using *YyonX* , we continue to improve the *Yy* design.

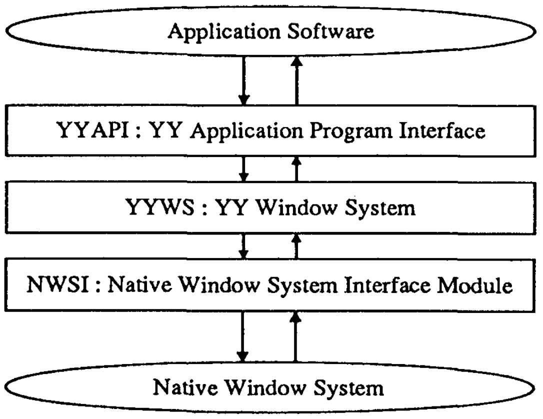
Application Software
YYAPI : YY Application Program Interface
YYWS : YY Window System
NWSI : Native Window System Interface Module
Native Window System

Figure 1. ***Yy*** Layered Model

References

1. M.Ida: YYAPI External Specification Manual v1.0 1989 Dec.

2. M.Ida, T.Kosaka, K.Tanaka: "Design of *YyonX* " IPSJ, march 1990