

Reconfiguring the Keck Instrument Desktop

***An Evaluation of Potential Replacements for
OpenWindows and the OLWM Window Manager***

**Al Conrad, James E. Lyke, & Gregory D. Wirth
*W. M. Keck Observatory***

Version 0.4
April 18, 2006

Table of Contents

1	Introduction	1
2	Evaluation Criteria.....	1
2.1	Assumptions	2
2.2	Requirements.....	2
2.3	Preferences.....	2
3	Systems Under Consideration	3
3.1	Windowing system choices	3
3.1.1	Window-manager-only choices.....	3
3.1.2	Desktop-plus-window-manager choices	4
3.2	Session Manager Choices	5
4	Analysis.....	6
4.1	Evaluation of options.....	6
4.1.1	Desktops and Window Managers	6
4.1.2	Session Managers	8
4.2	Competitive Analysis.....	9
5	Recommendations	9

1 Introduction

Since the introduction of Sun workstations at the W. M. Keck Observatory, observers have used the Sun default windowing system to launch instrument software and other tools such as xterms, text editors, IRAF and IDL. This windowing system, the Open Look Virtual Window manager (OLVWM), is provided by Sun with the purchase of the workstation as part of their standard operating system bundle.

Although this standard system has served CARA adequately for nearly a decade, Sun is discontinuing OLVWM in its next Solaris release. This document describes several possible paths for migrating to a new system.

First, we define several terms:

- A *window system* is software that enables the computer user to work with several programs or applications at the same time. The standard Unix/Linux window system is called *X windows*, while the typical window system on PCs is *Microsoft Windows*.
- A *window manager* is software that controls the placement and appearance of application windows. Under Microsoft Windows, the window manager functions are incorporated into the window system. Under Unix/Linux, a variety of window managers can be used in concert with the underlying window system. OLVWM is the one presently in widespread use at Keck.
- A *Desktop Environment* (a.k.a. *DE* or *Desktop*) provides a complete graphical user interface (GUI) solution to operate a computer. A DE provides icons, toolbars, applications, applets, and abilities like drag and drop. Microsoft Windows is a common example of a DE; typically, we do not run any DE on our Solaris workstations used for Keck observing, although we could do so.

The goal of this study is to identify the post-OpenWindows route that will provide observers with the best possible interface for running a Keck instrument. The remainder of this document describes the criteria we will use to evaluate the various options, lists the specific options under consideration, describes several tests we have run, evaluates the various options, and states our recommendations.

2 Evaluation Criteria

We will investigate several options for future instrument control beyond the OpenWindows interface. Beginning with a brainstorming session amongst the SA group, the OLWM-TNG subgroup has adapted and expanded a group of assumptions, requirements, and preferred attributes. In addition to the properties listed below, we will measure the speed with which the tested solutions login, logout, and start various software packages.

2.1 Assumptions

1. **No hardware or software changes to instrument computers.** Instrument computers are almost exclusively Sun, but they run various operating systems including SunOS and different versions of Solaris. The new method must work with them.
2. **No OpenWindows.** Since Sun has announced its discontinuation of OpenWindows support in Solaris 9, we do not consider its continued use an option.

2.2 Requirements

3. **Reliable.** We require a stable system which will not contribute to lost observing time due to software problems.
4. **Simple to use.** Observers must be able to easily figure out how to interact with the system, run instrument software, and launch other tools (editor, terminals, etc).
5. **Easy to administer.** No need to log into each account to change settings; no advanced expertise required to modify menus.
6. **Access from multiple seats.** Often, observers log into 2 or more workstations as the same user; this should not cause a conflict.
7. **Allow desktop sharing.** Provide built-in or EROS-like capability for eavesdropping on observers screens; this is essential for observing support.
8. **Permit access from remote sites.** Mainland observing must be supported.
9. **Support multiple instruments.** Provide the ability to access menus for multiple instruments on one screen or one login session. This will help daytime work by summit staff and for engineering.
10. **Compatible with existing tools.** Allow users to continue employing the software tools currently in use (xterm, tcsh, etc).
11. **Commonality/standardization.** Use the same interface for all instruments.

2.3 Preferences

12. **Easy to learn.** The interface should be similar to those with which observers are already familiar.
13. **Platform independent.** Future display computers may not be Sun/Solaris-based; compatibility with Linux is a plus.
14. **Provides launchers.** The solution should allow software to be launched easily using a taskbar, icons, or a pull-down menu.
15. **OLWM-like function/appearance.** We will more easily adapt to a system with a look and feel similar to OLWWM, than to a system with an entirely new look and feel.
16. **Centralized menus.** Account administrators should be able to modify all numbered accounts for a given instrument from a central file.
17. **Reload-on-demand.** The user should not be required to log out and back in again in order to effect changes in settings and menus.
18. **Low cost.** The solution should be available with minimal cost for licensing and maintenance fees; free, open-source software is preferred.

19. **Stable.** The software should be in a mature enough state of development that frequent upgrades will not be required.
20. **Well-supported.** The solution must not be a “flavor of the month”; we want a product that we can use for the next 5 years.

3 Systems Under Consideration

Traditionally, instrument control at Keck has been combined with the functions of the window manager; however, this need not be the case, and good reasons exist for not doing so. Accordingly, we have decoupled these considerations in our evaluation of the available options beyond the current OpenWindows setup. Below, we present the most promising options for desktop/window manager choices and also for the control of instrument software.

3.1 Windowing system choices

In considering the possibilities for selecting a window system, one may choose to run either a complete desktop environment that includes a window manager as a component, or to eschew the desktop and use only a window manager.

3.1.1 Window-manager-only choices

Table 1: Window Manager Options

Window Managers			
Name	License Type	Activity	Comments
FVWM	GPL	Hi	Virtual desktop
FVWM95	GPL	Med	Virtual desktop
TWM/VTWM	MIT	Lo	No
MWM	Open Group	Lo	No
CTWM	BSD	Lo	Multiple workspaces
OLWM/OLVWM	Custom	Lo	Virtual desktop
wm2/wmx	BSD	Lo	No
AfterStep	GPL	Hi	Virtual desktop
AmiWM	Custom	Lo	Multiple screens
Enlightenment	BSD	Hi	Virtual desktop
WindowMaker	GPL	Med	Multiple workspaces
SCWM	GPL	Lo	
IceWM	LGPL	Hi	Multiple workspaces
Sawfish	GPL	Hi	
Blackbox	BSD	Med	
Fluxbox	MIT	Hi	
Metacity	GPL	Hi	

As shown in Table 1, UNIX offers a plethora of window managers besides the venerable OLWM which is being discontinued. Many of the newest (those listed at the bottom of

the table) have been designed for use in a Desktop Environment, and hence are less fully featured and unsuitable for standalone use.

From the remaining choices, we have selected two – FVWM and SCWM – as representatives of the standalone breed of public domain window managers. Both are geared more toward experienced users of X-windows than to UNIX novices. Both have powerful features that allow programmers to tailor the window locations and sizes that are not really available with the Desktop choices covered in the next section.

■ **FVWM:** FVWM¹ is a no-frills window manager which has been on the X scene since 1993. The configuration files for creating original window placement are simple and straightforward and are well-suited for centrally managing multiple accounts.

■ **SCWM:** The acronym stands for Scheme Constraints Window Manager. SCWM is an offshoot of FVWM. Its main feature is the ability to specify a windows location, size, and front-back ordering, not only as fixed values as we do with OpenWindows and other window managers, but as ‘constraints’. For example: *Window A should always be the same width and directly below window B* is an example of a constraint.

3.1.2 Desktop-plus-window-manager choices

Table 2: Desktop Environment Options

Desktop Environments			
Name	License Type	Activity	Typical Window Manager(s)
GNOME	GPL	Hi	Sawfish, Enlightenment, Icewm, Metacity
KDE	GPL/LGPL/Various	Hi	kwm
CDE	Commercial	Lo	dtwm
XFce	GPL/BSD	Hi	XFwm

The available UNIX desktops shown in Table 2 and described in this section provide an extra set of features beyond what we now use with OpenWindows, and are more familiar in a PC or Mac interface. Specifically these are a taskbar and icons. The icons imply some mechanism that, for each file in the file system, associates an application to run when that file is double-clicked.

■ **GNOME:** The acronym stands for GNU Network Object Model Environment. This is the system being championed by Sun Microsystems as the successor to OpenWindows. It should be noted, however, that Sun does not always stay the course with these decisions. Three years ago CDE was the successor. Although it can be run on top of a variety of Window Managers, Sun currently packages it with the *Metacity* window manager. Account configuration under GNOME is

¹ There is disagreement on exactly what the F originally stood for. Sometimes “Famous Virtual Window Manager” is use. Typically, users just refer to it using the acronym.

carried out via a point-and-click interface, which is poorly adapted for the Keck model of centrally managing multiple accounts. On the plus side, GNOME utilizes CORBA; this is a factor we should consider in determining whether GNOME is the “desktop of the future,” or simply an unnecessarily complex product.

- **KDE:** The acronym stands for The K Desktop Environment. Although it is not CORBA-based it is very similar in its niche to GNOME. As a result, there have been many GNOME-versus-KDE reviews written. Given that Sun chose GNOME over KDE, we did not look deeply into KDE since the choice will likely be between GNOME and one of the more lightweight window-manager-only choices listed in the previous section.

3.2 Session Manager Choices

The session manager is the tool which observers use to run instrument software. It enables the user to start and stop instrument control GUIs, run guider eavesdropping screens, launch scripts to acquire data, and similar tasks. Traditionally, Keck has used the window manager’s pulldown menu feature as the session manager; however, the change in desktop function provides an opportunity to re-evaluate whether this is the best available option.

We consider the following choices to fulfill the role of the session manager in our future Keck desktop:

- **Window manager pulldown menu:** this is the system currently employed at Keck. Using the mouse to select options from the window system’s pulldown menu accesses the functions related to instrument control.
- **Browser-based menu:** one can set up a web page with links which, when clicked on, launch various instrument control commands. Keck’s adaptive optics team once employed this approach to control the AO system.
- **GUI-based menu:** Instead of accessing menus through the window manager’s pulldown menu, one can put these options into a menu available in a GUI which is always present on the screen. This is the approach taken with the *InstrMenu* GUI, a Tcl-based widget currently employed at Keck on many of our instruments (primarily as an engineering aid).

4 Analysis

4.1 Evaluation of options

4.1.1 Desktops and Window Managers

The GNOME desktop environment now ships standard with Sun's Solaris 9 system, and was available to us to test using two machines in Remote Ops. We also installed the latest stable version of the FVWM window manager (version 2.4.19 of September 2004). To get a sense of their respective features and performance, we performed a series of basic operational tests on the two systems. Our findings are summarized in Table 3.

Table 3: GNOME vs. FVWM Feature Comparison

Criterion	GNOME	FVWM
Requirements		
Simple to use	Yes	Yes
Easy to administer	No	Yes
Access from multiple seats	Problems	Yes
Allow desktop sharing	Yes	Yes
Permit remote access	Yes	Yes
Multiple instruments	Yes	Yes
Compatible with existing tools	Yes	Yes
Commonality	Yes	Yes
Preferences		
Easy to learn	Yes	Yes
Platform independent	Yes	Yes
Launchers supported	Yes	Yes
Desktop environment	Yes	Partial
OLWM-like	Somewhat	Yes
Centralized menus	Problems	Yes
Reload-on-demand	Unknown	Yes
Lean and mean	Less so	Yes
Low cost	Yes	Yes
Stable/mature	No	Yes
Vendor-supported	Yes	No

Although neither GNOME nor FVWM satisfies all of the criteria we have listed, the drawbacks of FVWM are minor while those of GNOME are not. Specifically, we highlight the following key issues:

- **The GNOME desktop is not well suited to being shared.** The GNOME desktop environment is particularly well suited for a single-user application, much as a PC would be appropriate to use on one's desktop. However, the way we use our computers and accounts in the Keck observing realm requires that a single observing account be used simultaneously on several machines. This presents a significant hurdle for GNOME, as it complains when multiple users attempt to employ the same account at the same time. Lacking a desktop, FVWM has no problems with simultaneous use by the same account on several machines.
- **GNOME presents a substantial administrative burden.** Hours of online research have failed to identify any simple means for an administrator to globally edit the contents of the pulldown menus for a large number of accounts at once. Each Keck instrument involves over 20 individual accounts which must be kept current, and being able to perform global operations on the accounts is essential to lowering the administrative burden on the staff and ensuring that all accounts remain in good working order. FVWM provides a very straightforward capability for centralizing pulldown menu contents.
- **GNOME pulldown menus are cumbersome to modify.** While GNOME provides handy interactive tools for reconfiguring one's own desktop with the mouse, the structure of the resulting menu files is impenetrable. We maintain instrument accounts by editing directly the menu configuration files. This would not be practical for GNOME configuration files. In contrast, FVWM menus are defined in simple text files which, like the OLWM menus currently in use, are simple to edit.
- **GNOME is not a fully mature product.** Although it has developed rapidly over the past several years to become the dominant desktop technology for Linux, GNOME is still changing rapidly and can be expected to evolve steadily over the next five years. As a result, frequent releases, and the requisite changes to our application software, are likely if we select GNOME. While FVWM is also evolving, it has been fundamentally stable for well over five years and is a good bet to remain usable in its present form for at least another five years.
- **Observing tools perform better under FVWM than GNOME.** Our limited performance testing indicates that FVWM consumes fewer system resources (memory and CPU) than GNOME, and also that typical Keck observing tools (instrument control software, etc.) operate more efficiently under FVWM.

Regarding the choice of window manager or desktop, the bottom line is that, despite the minor appeal of a desktop environment, GNOME and its brethren offers us very little in terms of important new functionality while placing a significant added burden on our administrative effort. This situation may change in the years to come as the technology matures and new tools for GNOME administration are developed, however, at this point it appears that a standalone window manager such as FVWM is the preferred solution.

The only apparent downside to FVWM is that it is not a vendor-supported product and hence we are on our own in supporting it. However, the observatory already makes significant use of other free software (e.g., Emacs, Tcl, Perl, etc.) and this experience has been very positive. Indeed, when tempted to adopt GNOME on the basis that it is championed by Sun, one needs only to recall that only a couple of years ago Sun was advocating the CDE desktop but has now abandoned that product. Had we followed Sun's advice all along, we would now be in the midst of our third major change of windowing environment in five years, having invested a substantial amount of work but deriving little real benefit. We should not be overly swayed by Sun's promotion of GNOME until this product matures and stabilizes.

Finally, we note that the FVWM vs. GNOME choice is to some extent a false dichotomy. The FVWM window manager is "GNOME compliant," meaning that it can be used in concert with GNOME. This provides the possibility that, should we in the future decide a desktop environment is needed, GNOME can be added and FVWM retained.

4.1.2 Session Managers

Table 4: Session Manager Options

Criterion	Window Manager Pulldown	Browser Menu	Menu Widget
Simple to use	Yes	Yes	Yes
Easy to configure/administer	Yes	No	Yes
Access from multiple seats	Yes	Yes	Yes
Allows remote access	No	Somewhat	Yes
Multiple instruments/one screen	No	Yes	Yes
Compatible with existing tools	Yes	Yes	Yes
Commonality	Yes	Yes	Yes
Small footprint (unobtrusive)	Yes	No	Yes

The window manager pulldown option is currently Keck's dominant paradigm for providing instrument control options to observers. The major drawbacks to this approach are that:

- a) the menu is not available unless the user is logged into a numbered instrument account on a Keck machine. This makes it cumbersome for support staff to utilize instrument menus from their office login sessions and it is incompatible with mainland instrument operation.
- b) the system does not allow more than one instrument to be controlled from a given login session.

These shortcomings are in fact what originally drove us to implement the *InstrMenu* Tcl-based widget menus.

The browser option for presenting menus presents no significant benefit over other options, plus it requires a large amount of screen real estate to display the menu. Given its abandonment by the Keck AO team, we do not regard this as a viable alternative.

Compared to the other options, the menu widget (*InstrMenu*-style) has no major drawbacks. Whatever language the menu widget is written in must be installed on the instrument computers, but most likely choices (e.g., Tcl) are already installed on these computers.

4.2 Competitive Analysis

It is worth noting what paths are being followed at Keck's peer observatories. Interviews with colleagues at other institutions paint the following picture:

Table 5: Tools in Use at Peer Observatories

Observatory	Platform	Window Manager or Desktop
CFHT	Linux	SCWM
Palomar	Linux	KDE
	Solaris	CDE
Lick	Linux	GNOME
Subaru	Solaris	CDE
	Linux	GNOME
Gemini	Solaris	CDE
VLT	HP-UNIX	??

With the exception of CFHT and Keck, all of these institutions are using the desktop approach. It is worth noting that the administrator of one observatory currently using GNOME expressed a strong opinion that the desktop approach was clearly inferior to employing a simple window manager. In this case, GNOME was apparently adopted due to a "path of least resistance" approach, as it was the default option supplied with new Linux hardware. In all, two administrators at these peer institutions discouraged us from using GNOME on the basis of cost/benefit analysis.

5 Recommendations

Based on our investigation of the options available and the advice we received from respected colleagues, the authors are unanimous in making the following recommendations to the group:

- As the principal GUI environment, a **window manager is preferred** over a desktop environment.
- **We recommend FVWM** as the most suitable choice for a window manager.
- A **widget menu** is the preferred solution for providing access to instrument functions.