Bull. Astr. Soc. India (2004) **32**, 75–??

IR-IMAC – A development approach

Vidhya Vaitheeswaran,[1] U. S. Kamath[1,2*] and Vidya Rangaswamy[1]
[1]*Indian Institute of Astrophysics, Bangalore 560 034*
[2]*Centre for Research and Education in Science and Technology, Hosakote 562 114*

Received 18 August 2003 ; accepted 12 December 2003

Abstract. IR-IMAC is a multi-utility software package developed to remotely control the Near-Infrared imager installed on the 2-metre Himalayan Chandra Telescope at Mt.Saraswati, Hanle. The core idea in the development of IR-IMAC was to effectively use the existing Open Source Code (OSC). We present here the features of IR-IMAC and its component-based development using Open Source Code approach.

Keywords : Instrumentation : Data acquisition software - IR-IMAC

1. Introduction

A Near-Infrared imager based on a 512 × 512 HAWAII array has been recently installed at the 2-metre Himalayan Chandra Telescope operated by Indian Institute of Astrophysics at Mt. Saraswati, Hanle. This instrument offers standard astronomical broad-band and narrow-band filters enabling users to obtain images at 1-pixel spatial resolutions of $0''.2$ (for the f/9 camera) or $0''.4$ (for the f/4.5 camera) covering a field of view of $1'.8 \times 1'.8$ or $3'.6 \times 3'.6$ respectively [1].

IR-IMAC (InfraRed IMage ACquisition) is a multi-utility package developed at IIA for use with this imager. The features of IR-IMAC and our development approach are described in this paper.

*e-mail: kamath@crest.ernet.in

[1] *See* http://www.iiap.ernet.in/iao/nir.html *and related pages for details.*

2. IR-IMAC

2.1 Design Philosophy

The design philosophy of IR-IMAC was to develop a comprehensiv, but very tightly integrated, product that would act like a Swiss Army Knife, with multiple capabilities built into a single interface. The guideline to develop such an interface was to follow an ideology of usefulness, simple interface and compliance to widely acknowledged GUI (Graphics User Interface) practices and, more importantly, hiding the complexities of the back-end process.

2.2 Component-Based Development

IR-IMAC is developed as a component-based product. Component-based development, a new paradigm in software engineering, enables a software architect to define a system that is made of simpler subsystems with well-defined interface between them. Component-based design is perceived as a key technology for developing advanced real-time systems in a cost- and time-effective manner. Already today, component-based design is seen to increase software productivity by reducing the amount of effort needed to update and maintain systems, by packaging solutions for re-use, and easing distribution. Building IR-IMAC on reliable well-tested components was a good starting point which led to a fast development life cycle. One of the main advantages of using component based-development lies in using Open Source Code (OSC) effectively. OSC is a concept that allows usage of existing, freely distributed source code and at the same time enables modifications and redistribution of the derived code. The device driver and DS9 screen interface are the two main base open source code components in IR-IMAC. The function of the device driver is to interact with the AstroCam controller, which houses a timing board with fibre optic connectivity to the host PC for image transfer. DS9, developed at the Smithsonian Astrophysical Observatory (SAO), provides a base platform to build the graphical user interface. These two components were readily available and very few modifications were required to fit them into the IR-IMAC design. Several development tools such as 'C', 'C++', Java, Tcl/Tk and Perl were analysed for their interoperability with the 'C' device driver code. Tcl/Tk emerged as a winner mainly due to the extension capabilities of Tcl. Tcl/Tk is actually a combination of two products – Tool Command Language and the Tool Kit – originally developed at the University of California, Berkeley . They are a versatile scripting language and a graphical interface development tool respectively. One of the strengths of Tcl/Tk is the concept of comprehensive, reusable, industrial-strength widgets. Examples include the Canvas and Text widgets. Because such widgets are complete and self-contained, they can be integrated into fully-featured applications. Tcl/Tk is also an open source code. It was also an easier choice for component based development methodology since DS9 is wholly written in Tcl/Tk.

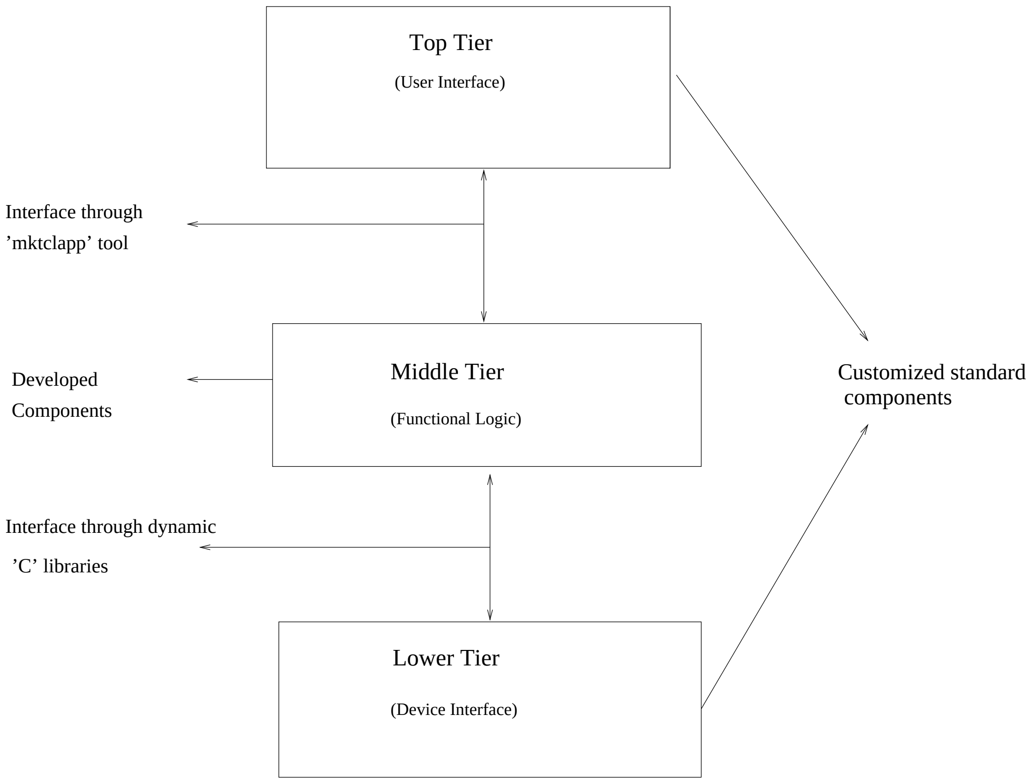

Figure 1. The 3-tier architecture of IR-IMAC

2.3 Software Architecture

IR-IMAC is a typical component-based system with n-tier structure. A tier or a layer is a part of the application which provides a specific functionality and has a well-defined interface to other layers. The application development in IR-IMAC followed a 'drop & glue' approach, picking components from standardised components, supporting all layers of system architecture. The lowest level consists of device driver components to interface with the near-IR camera. The middle level presents functional components and consists of programming logic and computational part of the application. The top tier presents a user interface which is responsible for getting user input data and displaying results. The functional logic tier manages the interface between the device and the top layer of user interface. This tier arrangement makes the development of the components as independent as possible. This in turn enables more flexibility for reusing standard components. Figure 1 shows the 3-tier architecture and standard components used.

Figure 2. The GUI of IR-IMAC

2.4 Graphical User Interface

The Himalayan Chandra Telescope and associated instruments are operated remotely from the CREST (Centre for Research and Education in Science and Technology) campus of IIA situated in Hosakote, about 40 kms from Bangalore. The Graphical User Interface was designed and tuned to fit the operational needs of remote observation using Virtual Network Connection (VNC). Screens have been designed with zero depth level which eliminates pop-up windows, listboxes and dialog boxes . This reduces refresh rates over the VNC thereby improving performance. Informational data about the image and the instrument are clustered and placed separately on the main screen. Seamless integration of the device driver and graphical user interface was made possible by powerful tools such as `mktclapp`. Figure 2 shows the GUI of IR-IMAC.

2.5 Features

IR-IMAC includes the look and feel of DS9 and functionality of IRAF-like features for image analysis. A single interface for device control (both AstroCam controller and motor controller), FITS file creation and management, housekeeping, device management, user input, archiving, monitoring of weather, analysis of image and image display are the attractive features of IR-IMAC.

3. Remarks : Future of IR-IMAC

At present the telescope, dome and various instruments are operated from individual servers / computers. It is envisaged that a master program called the observatory software would control these independent entities. Integration with the observatory software, an online data reduction system and more sophisticated data analysis tools are in pipeline for IR-IMAC.