PRECISION RADIAL VELOCITY SPECTROMETER

Document Title	**PRVS Instrument Control System Software Description**
Document Number	PRVS-TRE-00006-0001
Issue	1.0
Date	15th September 2006

Document Prepared By:	H Yamada, AJ Vick	**Signature and Date**	15th September 2006 H Yamada, AJ Vick
Document Approved By:	David Lunney	**Signature and Date**	15th September 2006 David Lunney
Document Released By:	David Lunney	**Signature and Date**	15th September 2006 David Lunney

CHANGE RECORD

Issue	Date	Section affected	Change Description
0.1	12th September 06	All sections	First release
1.0	15th September 2006	All sections	Final review before release to Gemini by DWL

Table of Contents

List of Figures

List of Abbreviations

A+G	Gemini acquisition and guidance system
FPRD	Functional and Performance Requirements Document
FOV	Field of view
FV	Fibre viewer
GCAL	Gemini facility calibration unit
HR	High (spectral) resolution
OAP	Off-axis parabola
OCDD	Operation Concepts Definition Document
PRVS	Precision Radial Velocity Spectrometer
PSF	Point spread function
R	Spectrograph resolving power
RV	Radial velocity
SRF	Spectral Response Function
XD	Cross-dispersed
HET	Hobby-Eberly Telescope
DR	Data reduction
OLDP	Gemini On line data processing environment
IRAF	NOAO Image Reduction Facility

List of Definitions

TBD	To Be Defined : a requirement to be developed during the preliminary design stage of the instrument.
TBC	To Be Confirmed : a requirement that is correct with the current design information but requires confirmation during the preliminary design stage of the instrument.
TBR	To Be Reviewed : a requirement specified to meet the PRVS top-level requirements, but which might over-constrain the design.

1. INTRODUCTION

The PRVS control software is required to fulfil the following basic tasks;

- Monitoring of temperatures (and possibly pressures) of the instrument
- Control of one mechanical movement and several shutters
- Control of power to several sub-systems (lamps, vibration stage)
- Control, read-out and data handling for the two science arrays
- Control and read out of the acquisition camera, and provision of centroid data to the TCS
- Overall control and sequencing of the whole system, including error reporting, logging etc.
- Interfacing to the Gemini observatory control system (OCS).

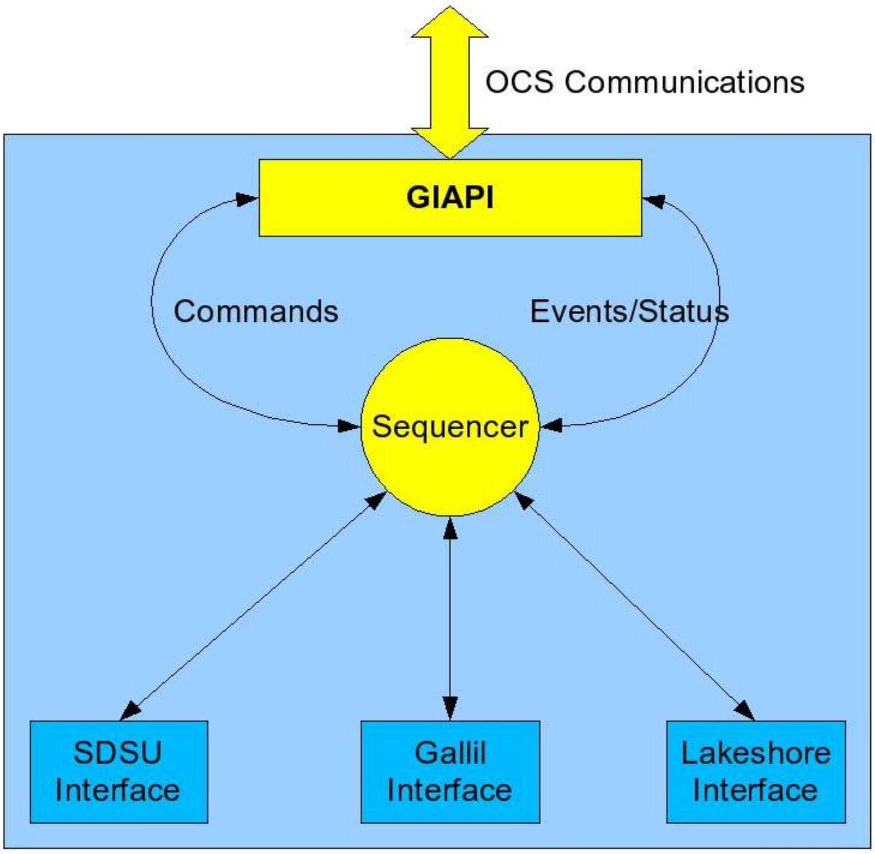

Figure 1: Top level sequence interfaces

The PRVS software will be based on the Gemini Aspen model. This specifies that a single top level interface be provided which links the Gemini Instrument API (GIAPI) with the instrument's sequencer; the GIAPI provides the instrument developer with the interface to the Observatory Control System (OCS). The PRVS software will provide a single instrument sequencer that uses the GIAPI as its external interface. This sequencer will then control all the subsystem interfaces that constitute the PRVS software system.

The design of the PRVS subsystem control is based on the use of "Ethernet appliances". The systems we have chosen to use, such as the Galil motion controllers and the Ultracam array control system, can all be viewed as intelligent controllers connected to the top level computer via ethernet. For each of these systems we have an API which allows them to be integrated simply into the sequencer; and they can in the first instance be run stand-alone (typically using provided GUI interfaces) which provides extra flexibility during integration stages.

There is one sub-system detailed in this report, the acquisition camera, which could be considered to be separate to the rest of the instrument control. Data derived from the output of this camera will be sent to the Gemini control system over a socket based interface, and the source of its commands may not be the GIAPI.

2. APPLICABLE AND REFERENCE DOCUMENTS

Reference	Document Title	Document Number	Issue & Date
AD01	SPIE meeting, Hawaii 2002 August 28, in "Advanced Telescope and Instrumentation Control Software II", Lewis, H.	SPIE 4848, "Astronomical Telescopes and Instrumentation"	Waikoloa 2002
AD02	Guidelines for Designing Gemini Aspen Instrument Software, Gillies, K.	Gemini Observatory Engineering Software Report	AspenSoft-03072004-6, May 13th 2004.

3. EXTERNAL COMMUNICATIONS

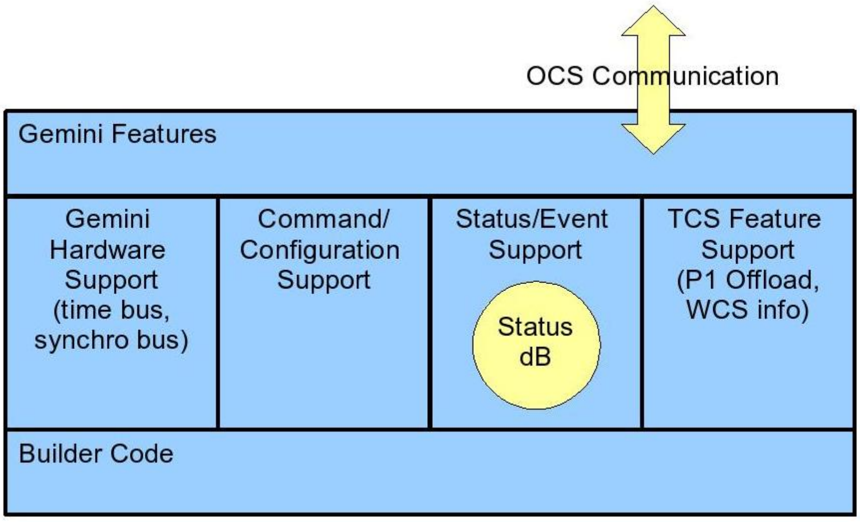

Figure 2: Gemini instrument API model

All communication with external (i.e. Gemini) systems will be via the GIAPI or, in the case of science data, via remote storage (e.g. NFS or Firewire-SAN). The GIAPI provides an Object Orientated interface to the Gemini Observatory Control System (OCS). The PRVS system will incorporate the GIAPI and provide hooks to support

- Command and configuration messages
- Status and event requests and updates
- TCS features; WCS info

It would also be possible for the acquisition camera system to receive commands (etc) from the GIAPI if this facility was available.

At this time there is no expected need for time bus or synchro bus use in PRVS, but if these where found to be necessary they could be added at a later date.

4. INTERNAL COMMUNICATIONS

The major components of the instrument system will be connected via a 1000-base TX ethernet router with numerical-address translation (NAT) capability. The majority of the instrument will be hidden from the external network, and the router will be configured so that only the top-level control computer will be visible to the world.

Communication within the PRVS systems will be via standard 100 or 1000 Base T ethernet. This internal network will not be directly connected to the Gemini network, so two choice exist for it's deployment;

1. Provide dual ethernet interfaces in the top level control computer, one connected to the outside world (the Gemini network) and one to the PRVS network.

2. Connect the PRVS network to an ethernet router with numerical-address translation (NAT) capability, configured so that only the top level computer is visible to the outside world (or has access to it).

Of the two solutions we have opted for the second, use of an intelligent router, since it reduces the need for special network setup on the top level computer (and means that it doesn't have to run NAT or DNS services). Apart from this there are only minor differences between the two systems so the choice could easily be altered at a later date if Gemini preferred.

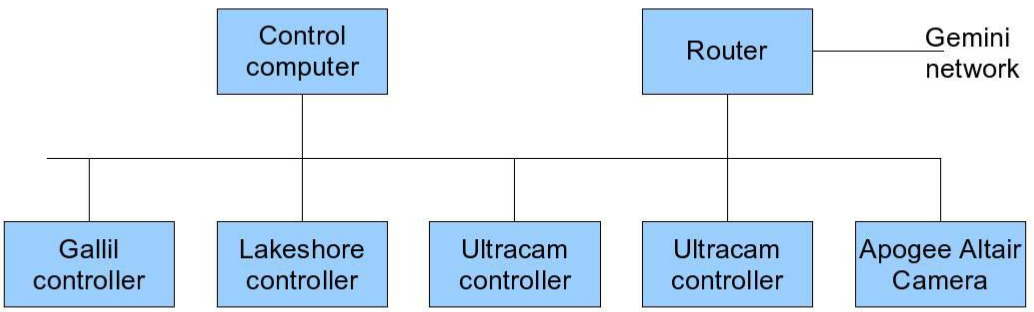

Figure 3: PRVS network architecture

5. THE TOP-LEVEL CONTROL COMPUTER

The top-level control computer will be an standard computer in an appropriate rack-mounted enclosure. It will use a standard Intel processor in the x86 processor family, with PCI bus, 1000-base TX ethernet interface, and serial-ATA disk drives. It will run an unmodified Linux kernel and current operating system release. It is expected that Gemini will recommend a specific operating system release, but, lacking such recommendation, Red Hat Enterprise 4.0 with the latest patches will be used. Most of the actual mechanism, detector, and subsystem control will be handled by appropriate subsystem processing units, so that the control computer will not require a real-time kernel or specialized hardware.

The top level computer will need to run a sequencer, a server program, to co-ordinate the various sub-systems. It is unclear at this time whether the GIAPI provides the server infrastructure, but we have considerable experience in writing systems like this. The Ultracam system is based on http: servers developed at ATC, and the server code (based around the ACE toolkit) is separable from the http: layer and can form the basis of the sequencer if required.

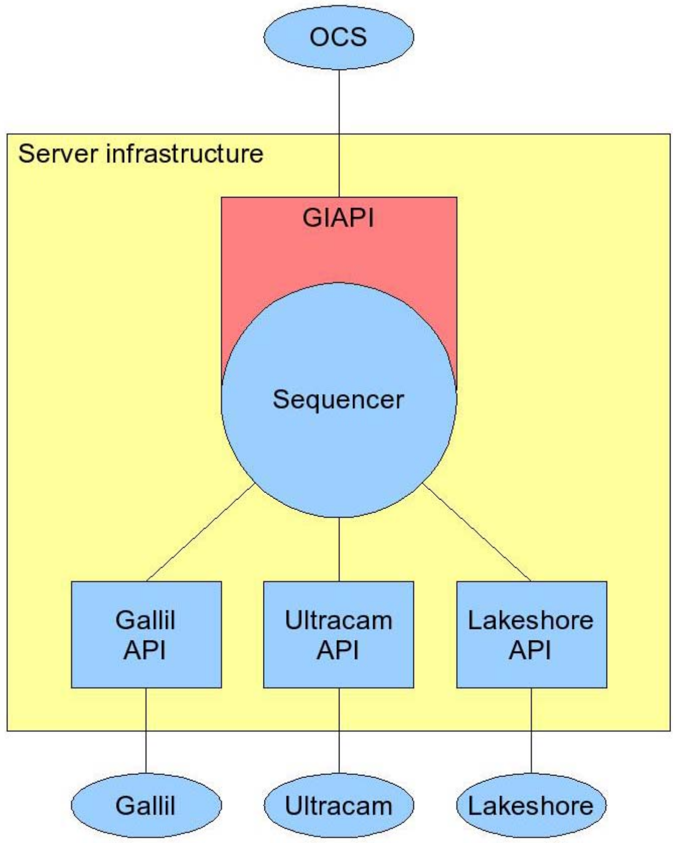

Figure 4: PRVS top level architecture

The sequencer will inherit the GIAPI as its interface to outside systems, and will use existing APIs to communicate with the subsystems.

5.1 ACCEPTANCE TEST AND ENGINEEERING USER INTERFACE (ATEUI)

The system requires an ATEUI during development and integration, and a development plan for an ATEUI is a requirement from the Aspen software document (seed AD02, section 10). The ATEUI is required to demonstrate proper operation of the instrument, and must at a minimum

- Allow control of all instrument mechanisms or devices
- Present all engineering status items and update these items in real-time
- Control data acquisition and observing modes completely

The PRVS team have experience in a number of areas which are appropriate to this user interface;

- Experience with Java/Channel access and Java/http: (Ultracam and WFCam)
- Experience with GTK2 and CORBA (MICO 2.3) (Pan-STARRS)

At this point in time we would choose to develop an ATEUI using the GTK2 and CORBA option, using CORBA to interface directly to the sequencer on the top level control computer, based on the experience of the University of Hawaii team. Engineering GUIs already exist for Ultracam (Java/http) and for the Galil motion controller (Java RMI to a supplied API) which can be used for early development work.

However, AD02 notes that the choice of technologies for the ATEUI is a part of the PDR review, and we intend to discuss this area with Gemini software representatives early in the PDR stage to ensure we minimise our own effort and provide a usefull engineering tool to the project.

6. MECHANISMS AND SUBSYSTEMS

6.1 MECHANISM CONTROLLERS

PRVS has no real-time hardware requirements. All mechanism control can take place at a low data rate, less than 1 Hz. Mechanism control will use a Galil intelligent motion controller in the 21x3 family. This motion controller provides an ethernet interface, digital input and output, and analog input capabilities. A control program running on the control computer will continually monitor the status of the Galil motion controller. It will use the ethernet interface using the standard Galil instruction set to send commands to the motion controller, using the Galil API on the top level computer. The mechanism control program must be able to;

- Start the homing operation
- Start a movement command
- Reject a motion command if the mechanism is not initialized
- Report errors, status, and position information
- Report completion information
- Interrupt operations

6.2 GUIDER

PRVS will use the peripheral wavefront-sensors to for guiding. A separate guide camera will monitor a guide star and will provide incremental corrections. These corrections will only need to handle errors that are not handled by the peripheral wavefront-sensors (such as instrument flexure), so will occur at a very low data rate (less than 1 Hz). The proposed camera system is the Apogee Alta (NB: selection of this camera will be subject to further review). This system comes with an ethernet interface and is provided with a Linux-compatible control library. The camera controller will handle the details of camera configuration and readout, so that a separate guide-control computer will not be required. The guider program will take data frames from the Apogee Alta, subtract the background and centroid the data, and compute a correction term. It will read the telescope azimuth, elevation, and instrument rotator angle, and use the information to convert the data into telescope offsets, which will then be passed on to the telescope control system. Since these corrections will be taking place at a low data rate, no special communications interface will be needed; a socket based interface can deliver the offsets to the Gemini control system. The guider program must:

- Provide a data frame for initialization
- Analyze the data frame to find the guide star
- Read out the appropriate region of the detector
- Compute a centroid
- Compute a correction factor

- Read telescope information
- Transform the correction factor into appropriate coordinates
- Provide the telescope with offset values

6.3 DETECTOR CONTROLLERS

The detector controller system will be based on two San Diego State University Generation III controllers, each using a 250MHz PCI interface to a control computers. Each detector controller requires its own computer because of bandwidth requirements. The operating system will be the same as for the control computer, but will require a real-time kernel. The detector controllers must:

- With the system in observing mode, move a suitable distance off target and take a sky frame.
    - Slew to a sky position
    - Take (some) frames
    - Slew back to original position
    - Produce an OHSKY data set
- Standard star frame
    - Exactly the same as a Science observation, but on a standard star
    - Produces a STDSTAR data set
- Observing
    - Starts after acquisition has occurred and any required calibrations are available
    - Download
    - Exposure time
    - No. of reads
    - No. of co-adds (if required)
    - OH line windows
    - Check guiding OK, image stabilised (i.e. PWFS loop has closed)
    - Start observation

- Save data (produces a SCIENCE dataset)

The detector control system for the two arrays will be based on the Ultracam system, using SDSU III controllers connected to standard PCs using SDSU fibre links to the PCI bus. This system has been used on several previous instruments (Ultracam, WFCam, Dazzle; see also AD01), and will require minimal work to support the additional features required by the PRVS system. This system provides a stable platform for array controller development; most applications will never need to change any part of the code except the DSP application.

The system provides a low-level engineering interface (A Java GUI) suitable for development in the lab, and since the sequencing of two arrays is achieved using hardware on the SDSU controllers it is trivial to run the two controllers separately, which allows flexibility in integration schedules.

The system has several APIs available (C/C++, Java and Python) which allow full control from external systems; PRVS will use the C/C++ API to enable the control computer to command and get status for the system, and will use NFS disk access to provide data to Gemini systems as FITS files.

6.4 TEMPERATURE CONTROLLERS

PRVS will use Lakeshore temperature monitors/controllers. The PRVS team have significant experience in using these devices, and they have been used extensively on Gemini instruments before. Of particular importance is their prior use in controlling detector array temperatures to 10mK.

PRVS will use the Lakeshore controllers in stand alone mode; no control computer will be required for their temperature control function to operate. Connection to the devices will be via ethernet to serial (RS232) converter, as such the Lakeshore will appear to be another internet connected appliance. We have used Lakeshores in this mode before, and a source code API exists to facilitate integration into the PRVS system. This will enable the central sequencing system to monitor temperatures and (if required) to adjust parameters in the controller. This mode of operation means that there is no "real-time" requirement on the control software (though there may be performance requirements). The Lakeshore will directly monitor and control the temperature using it's in-built functionality.

The control software will need to be able to

- Report the current temperature
- Change the set point and other related parameters
- Report the set point and other related parameters
- Generate an error (send a message) if the temperature is out of range

All of these functions are available within the existing API, and a scripting language interface is available to provide simple monitoring and control in the lab at the start of PRVS integration if required.

7. SOFTWARE INFRASTRUCTURE

Software for PRVS will be written in C++, using gcc 3.4. Software configuration and setup will be managed using standard Bourne shell scripts and makefiles. The engineering user interface will be developed using the gtk 2 toolkit. Internal version control will be handled using SVN. Problem reports will be handled using Trac.

Software for PRVS will be mainly written in C++, with some lower level C code being inherited as part of re-use. We are currently using gcc v3.4 as the compiler, and use the following existing source code and libraries;

- LibCURL v7.11.1
- libxml2 v2.6.8
- gtk2 v2.4.0

Internal to the project we will use Subversion for version control and Trak for issue tracking. Source code is documented (and the documents generated) using the Doxygen system; documentation can be supplied in PDF and/or HTML formats. Reports and project documentation will be produced using MS Word.

During the development we will also export milestone releases to the Gemini configuration control systems, and we expect to move over to using Gemini systems as the primary repository for source code and issues when we enter integration phase of the project.