RAA 2015 Vol. **15** No. **7**, 1077–1088 doi: 10.1088/1674–4527/15/7/015
http://www.raa-journal.org *http://www.iop.org/journals/raa*

Research in Astronomy and Astrophysics

A general observatory control software framework design for existing small and mid-size telescopes

Liang Ge[1,2], Xiao-Meng Lu[1] and Xiao-Jun Jiang[1]

[1] Key Laboratory of Optical Astronomy, National Astronomical Observatories, Chinese Academy of Sciences, Beijing 100012, China; *geliang@bao.ac.cn*
[2] University of Chinese Academy of Sciences, Beijing 100049, China

Received 2014 April 17; accepted 2014 July 9

Abstract A general framework for observatory control software would help to improve the efficiency of observation and operation of telescopes, and would also be advantageous for remote and joint observations. We describe a general framework for observatory control software, which considers principles of flexibility and inheritance to meet the expectations from observers and technical personnel. This framework includes observation scheduling, device control and data storage. The design is based on a finite state machine that controls the whole process.

Key words: telescopes — instrumentation: detectors — control software

1 INTRODUCTION

In recent years, astronomers have been pursuing large aperture telescopes to improve detection ability. However, compared to the huge costs associated with large telescopes, small and mid-size telescopes can provide a cost effective alternative, especially considering the scientific investigations that can be done with small and mid-size telescopes (Bailyn et al. 2007).

In China, there are more than 15 small and mid-size optical telescopes, whose apertures are between 50 cm and 200 cm, that are located in different observatories. These telescopes, which have different optical/mechanical systems and have been equipped with various instruments, are operated for various scientific studies. Because the operational modes for controlling these telescopes are different, observers have to learn different skills when using different telescopes. Troubleshooting problems in different telescope systems is also a challenge for technical personnel. A telescope that can be controlled in the same mode under a general framework is helpful for remote and joint observations as well.

Currently, there are two types of general frameworks used in this field. One is the Astronomy General Object Model (ASCOM) (Denny et al. 1998 based on Windows. The other is RTS2 (the second version of a remote telescope system) (Kubánek 2010; Castro-Tirado 2011) which is based on Linux. Each framework is supported by many devices (see *http://www.ascom-standards.org* and *http://rts2.org*). However, there are some obstacles for existing telescopes in China regarding how to implement ASCOM or RTS2:

(1) When the observers' attention is focused on variations of celestial objects, whether transient or long-term, they need the observatory control software (OCS) to observe their objects of interest

flexibly at any time during normal observations. As a result, the framework has to be flexible enough to meet the observers' expectations. Both frameworks mentioned above are too big and complicated to accommodate new ideas from observers when the telescope is operating.

(2) Most telescopes that currently exist were built in the last century. They do not support the standards of the above frameworks. Also, some devices that are part of one existing telescope may be controlled in the operating system of Windows, but others may be controlled by Linux. Hence, neither of these frameworks can cope with the entire system, because each of them only supports part of the system.

(3) In China, most of the devices on existing telescopes, including hardware and software, are developed and upgraded by technical personnel themselves at different observatories. Therefore, it is difficult to adopt a complete standard like ASCOM or RTS2 in these highly customized telescopes.

The users of OCS are observers and technical personnel. To meet their expectations, it is ideal for one to design a general framework for all existing telescopes. However, it is a huge challenge to make the general framework operational for all telescopes, considering the numerous differences that exist among these telescopes. As a result, we take an alternative approach by focusing on making the framework flexible.

Due to the efforts of our engineers in recent years, all of the existing telescopes are computer-controlled, making them 'modern' telescopes to some extent. Hence, another principle in this framework is inheritance, which means making the best use of the original resources incorporated in the telescopes.

The framework presented in this paper focuses more on how flexibility and inheritance are useful for different observers and various personalized telescopes. The detailed design is described in Section 2. The applications are shown in Section 3. In the last section, we draw our conclusions.

2 FRAMEWORK DESIGN

A typical observing sequence includes scheduling, device control and data storage, which are shown in Figure 1. Information about the environment is recorded to support data reduction, provide reference for scheduling, and alert the OCS to stop observation during poor weather conditions. A general framework should be able to accept schedules from different scientific research programs, control different telescopes in a uniform interface and output images according to a unified standard. Scheduling is initially provided by the observer, and should be able to be flexibly changed during observations by the state of the devices, data quality and environment. Therefore, device acquisition (Sect. 2.1) is described first in this paper. Then the design of scheduling is described in Section 2.2, and the data storage is shown in Section 2.3.

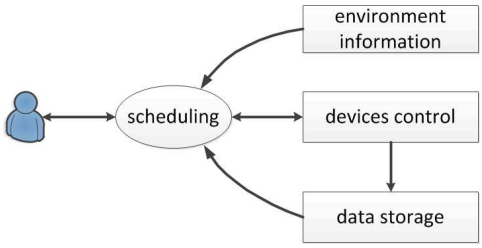

Fig. 1 A typical sequence of observation.

Meanwhile, a finite state machine (FSM) (Wright et al. 2005[1]) will be used in this paper to allow a continuous sequence of control to be administered. The FSM is a design model used in software systems, which identifies what state the system is in, which event triggers a transition, and how the system reacts at a given state. However, the real observation system, which involves different schedules and strategies, various mounts and detectors and so on, is so complicated that it is difficult to make a complete FSM model for it, especially when the system is malfunctioning. Therefore, we introduce an 'Error' state to complete the state set, which needs to be dealt with automatically or manually by different users.

2.1 Device Control

The main function of device control is to complete the given schedule and acquire the data as required. As shown in Table 1, the devices associated with a typical telescope are divided into enclosure, optical/mechanical structure and instruments. The enclosure includes the dome, slit, air brattice, etc. The optical/mechanical structure includes the mount, field de-rotator, focuser, mirror cover, etc. The instrument works mainly for photometry or spectroscopy. The core device of the instrument is the detector, such as a CCD and CMOS. The instruments used for different targets may also include filters, flip mirrors, calibration device, grating, etc. The various devices can be divided into the front-end system, including axes, switch unit and detector. Although the level of automation for telescopes continually improves, some devices, such as gratings with different resolutions, need to be manually changed. The objects in a telescope system that require manual adjustment are not discussed in this paper.

Table 1 Typical devices that are part of telescopes.

Devices of telescope		
Enclosure	Optical/mechanical structure	Instrument
dome	mount	filter
slit	focuser	rotator
air brattice	cover	grating
...	rotator	slit
	...	calibration devices
		...

The control software for all the devices in this framework is designed based on FSM. As mentioned above, we introduce the 'Error' state in device control. However, we will not define how to leave the 'Error' state so that flexibility and inheritance can be improved. The ways to return a normal state are different for different devices, and should be decided by the fundamental rules used in the original software, the experience of personnel and new ideas in the future. Developers can add to or revise them at any time.

2.1.1 Axis control

Most of the devices that are part of a telescope can be classified as 'axis.' These include the two axes in a telescope mount, focuser, tracking dome, filter wheel, etc. As stated above, the axis control is based on FSM. The set of states of one axis can be divided into two main types: running and finished. The states include:

(1) Not Homed: If the encoder of the axis is relative, it needs to find home first. The 'Not Homed' state means that the axis has stopped moving, but it cannot point by giving a position command

[1] *http://www4.ncsu.edu/ drwrigh3/docs/courses/csc216/fsm-notes.pdf.*

because the current real position of the axis is unknown. If the encoder of the axis is absolute, this state does not exist.
(2) Finding Home: this is a running state in the process of finding home. Obviously, there is no such state for an absolute encoder.
(3) Idle: a finished state. The axis already stopped. It can move when a command is received.
(4) Stopping: a running state. The axis is in the process of stopping, but has not stopped yet.
(5) Pointing: a running state. The axis is moving to a given position, but has not arrived.
(6) Wait Moving: a running state. The axis is going to move at a given speed, but the speed has not been achieved.
(7) Moving: a finished state. The axis is moving at a given speed, and the following error meets the necessary requirements.
(8) Error: the system enters the 'Error' state when any problem occurs. Table 2 shows some errors in different states.

The events of axis control include:

(1) Find Home: this event means the start of finding home, and it clears the zero point of the encoder.
(2) Point: the axis points to a given position after this event.
(3) Move: the speed of the axis will reach a given value when this event is triggered.
(4) Stop: this event will stop the axis.
(5) Home Found: this event is triggered when the zero point of the axis encoder is found. If the encoder of the axis is absolute, this event is true all the time.
(6) Home not Found: while finding home, this event would arise if the time is out or the axis is moving too far.
(7) Pointed: in the process of pointing, this event would arise when the axis has arrived at the desired position and the pointing error is acceptable.
(8) Moved: in the process of waiting while moving, this event arises when the given speed of the axis has been reached and the following error meets the necessary requirements.
(9) Stopped: this event means the axis has stopped from the running state.
(10) Error: this event is triggered when an error occurs. Some errors are listed in Table 2.

The first four events are triggered by users, but others are triggered when monitoring parameters related to the system.

The state transition diagram (STD) of axis control is shown in Figure 2.

As stated above, most of the devices consist of one or more axes. The focuser, which is attached to mirrors or an instrument, usually uses a single axis control. However, the mount control is a multi-axis control, and finding home and checking the zero point need all the axes to be involved.

Table 2 Errors in Axis Control Which Could Occur in Each State

State	Errors
Not Homed	Encoder error, Servo error
Finding Home	Encoder error, Servo error, Moving Error
Idle	Encoder error, Servo error
Stopping	Encoder error, Servo error, Moving Error
Pointing	Encoder error, Servo error, Moving Error
Wait Moving	Encoder error, Servo error, Moving Error
Moving	Encoder error, Servo error, Moving Error

Notes: Encoder error includes no counts, no hop and so on. Servo error means that the voltage or current is above the normal level, or the communication to the servo control is disconnected. Moving error may be caused by a limit switch, high speed, slip, incorrect direction, etc.

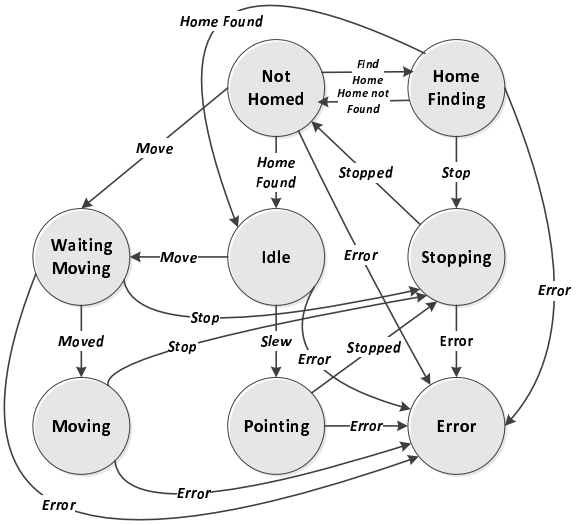

Fig. 2 The state transition diagram (STD) of axis control.

The main function of the mount is tracking objects, so two more states and three more events need to be included into the FSM of mount control. The additional states are:

(1) Tracking: a finished state, when the mount is tracking a given object.
(2) Waiting Tracking: a running state, when the mount is still in motion, but not ready for tracking a given object. It has neither reached the required speed nor the specific position.

The additional events are:

(1) Track: when an event is triggered by users, the mount starts moving for tracking.
(2) Tracked: tracking error under the state of 'Waiting Tracking' is being monitored. A 'Tracked' event is triggered if the tracking error is acceptable.
(3) Untracked: an 'Untracked' event would be triggered if the tracking error is not acceptable in the state of 'Tracking.'

The STD of mount control shown in Figure 3.

Another device, the tracking dome, most likely operates in coordination with the mount although it has only one axis. A filter wheel and a flip mirror are used to change the optical band or path. They use the same single axis control. What we need to pay attention to is that the position that the telescope needs to point toward is discrete.

2.1.2 Switch unit control

Switch units include mirror cover, shutter, rolling-roof enclosure calibration device, etc. The switch unit can be treated as a simple axis; in some cases, it has no encoders. Whether the switch unit is on or not is the only information required in such cases. The set of states can be defined simply as:

(1) Idle: a finished state. The unit is stopped. It transits to 'Full Open' directly if it is fully opened. It transits to 'Full Close' if it is fully closed.

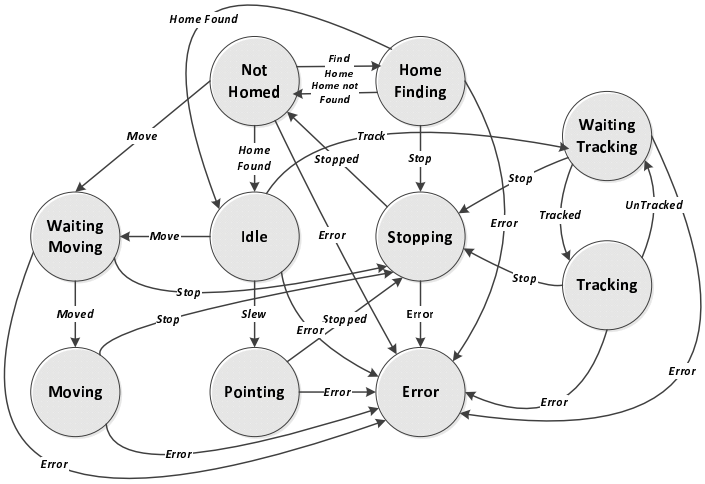

Fig. 3 The STD of mount control.

(2) On: a finished state, the unit is fully opened.
(3) Off: a finished state, the unit is fully closed.
(4) Opening: a running state, the unit is in the process of opening.
(5) Closing: a running state, the unit is in the process of closing.
(6) Stopping: a running state, the unit is in the process of stopping
(7) Error: the 'Error' state of a fully open unit is the same as the axis control.

The events are:

(1) Open: the unit will open to the state of 'On.'
(2) Close: the unit will close to the state of 'Off.'
(3) Stop: the unit will stop moving.
(4) Full Opened: this event is triggered when the unit is fully opened for observations.
(5) Full Closed: this event is triggered when the unit is fully closed.
(6) Stopped: this event is triggered if the unit stopped in the state of 'Stopping.'
(7) Error: the same event with one axis control.

The STD of switch unit control is shown in Figure 4.

2.1.3 Detector control

Although there are various instruments on existing telescopes, most of them can be classified as a moving unit and a detector. The moving unit, e.g. rotators or switches, has one or more axes. For a detector, although its behavior is complicated, we can still use a two-step sequence to describe it:

(1) Exposure: this step includes actions such as cleaning residual charge, opening and closing a shutter (mechanical) and exposure.
(2) Reading Out: this step includes reading out and saving images (if needed).

Accordingly, a set of states for a CCD include:

(1) Idle: a finished state. The detector is idle and prepared for a new exposure.

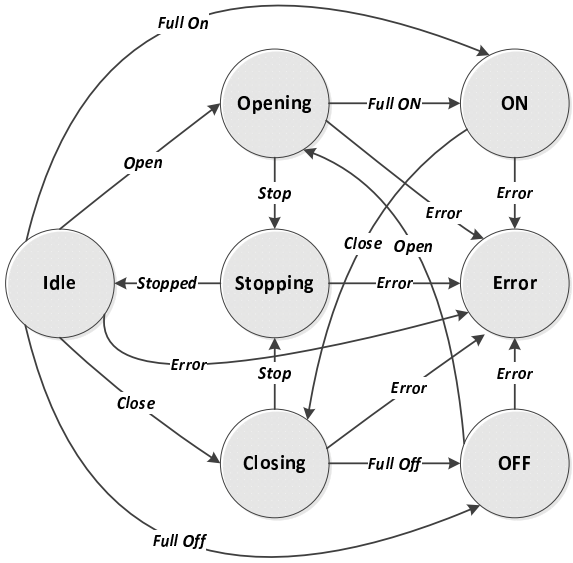

Fig. 4 The STD of a switch unit control.

Table 3 Errors for a Detector in Each State

State	Errors
Idle	Hardware error
Exposing	Hardware error, Shutter error, Timeout error
Reading Out	Hardware error, Timeout error, Storage error
Aborting	Hardware error, Timeout error

Notes: Hardware error includes high temperature, fan malfunction and so on. Timeout means that the time for finishing a given action is beyond an acceptable range. Storage error is caused by inadequate disk space, unmatched storing speed, etc. when data acquired from the detector are being recorded on storage devices.

(2) Exposing: a running state. The detector is in the step of 'Exposure' described above.
(3) Reading Out: a running state. The detector is in the step of 'Reading out' described above.
(4) Aborting: a running state. The detector is aborting from the running state.
(5) Error: if the detector is affected by any problems, the system goes into 'Error' state. Table 3 shows some errors under different states.

The events of axis control include:

(1) Expose: this event starts an exposure if the detector has been prepared.
(2) Stop: this event stops exposing under the 'Exposing' state, then reads out. An image can be obtained when it has stopped in the state of exposing or is reading out.
(3) Abort: this event aborts any running state, and no data are obtained.
(4) Exposed: this event is triggered when the exposure time is achieved or a stop event occurs.
(5) Read: this event is triggered when the data are read out, and stored if storage is required.

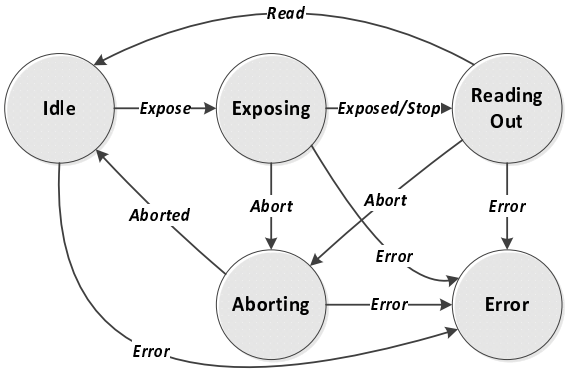

Fig. 5 The STD of detector control.

(6) Aborted: this is triggered when the data have been abandoned and the detector is preparing for a new exposure.

The first three events are triggered by users, while the others are triggered by monitoring parameters related to the system. The STD of the detector control is shown in Figure 5.

2.2 Scheduling

Traditionally, the function of scheduling helps the observer/devices to select which object to observe. The selection can be decided by time, zenith angle and so on. It has been examined in detail in different software packages, such as the ACP Scheduler (Denny 2004). For different targets, there is one or even a group of different references accordingly. It is almost impossible to have a general method which can cover all requirements from observers. However, in our approach, selection is not part of scheduling; instead, it is the task for observers. Our approach to scheduling in this paper is to form a direct link between the observer's requirement and OCS. The function of scheduling is to decide when and where the mount is pointing and tracking, how to set exposure parameters, etc. For existing telescopes, the scheduling is usually a new module, and is not closely related to the devices. Hence inheritance is not so important here, and flexibility is what we consider more.

Input keywords for a sequence of observations include: object position (RA, Dec, Epoch) and exposure parameters. For photometric instruments, the parameters usually include filter band (such as *UBVRI*/u' r' g' i' z'...) and the detector's parameters. For spectroscopic instruments, the parameters usually include central wavelength, band coverage, resolution, slit width, etc. The detector's parameters include exposure time, gain, readout speed, etc. As for observers, what they need to provide are just some keywords, or probably a couple of lines of text. The role of OCS is to input these keywords into the observation.

The set of states in scheduling can also be divided into two main types, running and finished. These states include:

(1) Idle: a finished state, all the devices have stopped, a new sequence can be started.
(2) Stopping: a running state, all the devices have been sent a command to stop, but not all are in the 'Idle' state.
(3) Waiting Tracking: a running state, the OCS sent commands to the mount, filter wheel, dome, focuser (if the telescope supports auto-focus), rotator unit, switch unit, etc. However, not all these devices have reached their given state. For example, the mount is in the 'Waiting Tracking' state while the filter wheel is still rotating.

(4) Tracking: a finished state, where all the devices except the detector are prepared for exposure. It is the last opportunity to configure the parameters for the instruments.
(5) Exposing: a running state, when the detector is being exposed. The other devices related to the exposure process cannot change their state at this moment.
(6) Reading Out: a running state, when the detector is in the process of reading out or saving data, which may take from some milliseconds to dozens of seconds. In this state, other devices can advance to the next item in the sequence to improve efficiency or change nothing to decrease the possibility of interference in the circuitry of the detector. This selection is flexible.
(7) Error: a malfunction can occur on any device in the system during observation. An error state in scheduling therefore is needed, like in device control. By following the same rules, specific methods on how to leave this state are not defined. They can be decided by technical personnel themselves to improve flexibility and inheritance.

The events that are part of scheduling include:

(1) Start: this event starts observation when the system is idle.
(2) Stop: this event stops a sequence. If the detector is at the state of 'Exposing,' it would stop exposure, then read out. If it is reading out, it would wait until this step finishes. An image can be acquired when a sequence stops in the state of 'Exposing' or 'Reading Out.'
(3) Abort: this event aborts a sequence, no matter which state the system is in when abort is triggered. No image can be obtained if this event arises.
(4) Tracked: this event is triggered only if all the conditions below are met.
(5) The mount is in the state of 'Tracking,' and the tracking error is permitted.
(6) If the dome is in tracking mode, it is tracking with the mount and the tracking error is acceptable. If it is in full-open mode, it has been fully opened.
(7) If there is a shutter, the shutter has been opened.
(8) If the focuser needs to move, it has completed this operation.
(9) If the filter wheel needs to move, it has completed this operation.
(10) If there is any other switch unit, such as a mirror cover, it has moved into the given position.
(11) If there is any other rotator unit, such as a tertiary mirror, it has moved into the given position for focus.
(12) Instrument Prepared: this event is triggered if the detector is idle, and all the parameters of the instrument are configured.
(13) Exposed: this event arises when the exposure of the detector has finished.
(14) Read: this event is triggered when the data from the detector have been read out.
(15) Stopped: this event is triggered if all the devices have been in the 'Idle' state.
(16) Error: this event arises when any of the states of the devices, especially the 'Error' state, can affect the observation.

If the weather is bad, such as being cloudy, rainy and so on, this event arises. The first three events are triggered by users, but the others are triggered by monitoring parameters related to the system.

The STD of scheduling is shown in Figure 6.

As the diagram shows, there is no way to leave the error state. The automatic or manual events that need to take place before leaving an 'Error' state can be added according to the experience of the observatory's engineers and updated later. By incorporating flexibility and inheritance into this framework, administrators can efficiently upgrade software in existing telescope facilities.

2.3 Data Storage

The Flexible Image Transport System (FITS), as the standard format used for astronomical imaging data, was originally developed in the late 1970s to enable the exchange of astronomical image data

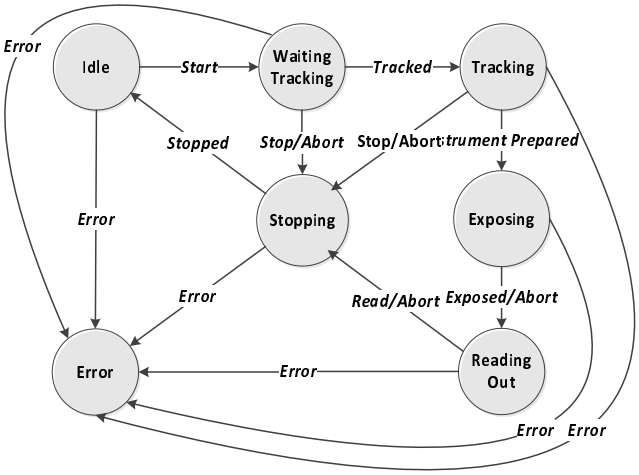

Fig. 6 The STD of scheduling.

between different types of computers, with different word lengths and different means of expressing numerical values (Hanisch et al. 2001).

For existing small and mid-sized telescopes, massive amounts of historical data have been accumulated and new data are continually accumulated. We choose the Xinglong Observatory Public FITS Header Standard (XPFHS) (Lin et al. 2013) to be the standard of data storage format in this general framework. In Lin's paper, a flexible system of data storage was designed in detail. Hence we implement it in our general framework, which follows the principle of inheritance and flexibility.

3 APPLICATIONS OF THE FRAMEWORK

3.1 Flexible Solutions for Observation

As mentioned above, all the processes involved with observation are controlled with the model that uses the FSM. Therefore, the OCS could respond flexibly and stably for any event under any state, such as changes in the observing schedule, variations in the environment or malfunctions in the equipment. The method for acquiring observations include:

(1) Instruments (Photometry/Spectroscopy): in this framework, the instrument is divided into two parts: the front-end system and the detector. For photometric instruments, the front-end system includes a filter wheel, flip mirror, etc., to change the band, optical path or other parameters. For a spectroscopic one, the front-end system includes a slit, flip mirror, moving units, etc., to change the central wavelength, resolution, dispersion or other parameters. No matter if the instrument is photometric or spectroscopic, the scheduling model of this framework can handle them in the same way. Their configuration is the only distinction between different instruments.
(2) Adjusting the Schedule (automatic/manual): To improve data quality, an observer might like to change configurations as external conditions change, such as an environmental change. During a given observation, observers may find some interesting targets from the acquired data. Then the original schedule needs to be changed rapidly. As mentioned above, the schedule could be adjusted conveniently at any time. Observers can change the schedule before one sequence, or change it after the current sequence to acquire more data, or abort at any time to start a new

sequence to catch a new target. This process is usually handled manually at first, but it can be automated with this flexible framework if the algorithms are reliable.

(3) Guider (automatic/manual): In this framework, the guider can be treated as an instrument controlled with the OCS, which works during the whole process of observation. The target object for the guider, on-axis or off-axis, is provided automatically by scheduling or manually by the operator. The offset position for the telescope is fedback to change the schedule. The framework provides an interface to change the schedule under any state, therefore, the guider, which is automatic or manual, can be conveniently added.

(4) Focus (automatic/manual): The process of focusing involves a series of special exposures, the data of which do not need to be saved. The framework in this paper provides a selection for an instrument to save data or not. In addition, the framework provides an interface to configure the instrument, such as focal length. As Figure 7 shows, the focal length can be changed at any time before the state of 'Exposing.' Obviously, this process can be automatic or manual in this framework.

(5) Error diagnosis and device protection: We introduce the state 'Error' in scheduling and device control. When the event of 'Error' is triggered, the sequence of observations will stop and the system will change to the state of 'Error.' If the event of 'Error' is enough of an emergency, such as poor environmental conditions, the enclosure of the telescope would be closed for protection. The method of leaving the state of 'Error' is defined differently for different telescopes, which reflects the inheritance and flexibility of this framework.

(6) Other special requirements: There are many other special requirements for different telescopes. The framework in this paper does not define all conditions, but provides a flexible interface. The solutions are based on the inherited resources at first, and can be flexibly upgraded in the future.

3.2 Practical Applications

The general framework in this paper has been partly implemented for the 1.26 m optical/infrared telescope at Xinglong Observatory of National Astronomical Observatories, and the 1 m wide-field telescope at Nanshan station of Xinjiang Astronomical Observatory.

The 1.26 m optical/infrared telescope was built in 1985, and the optics and control systems were upgraded during 2011–2012. After upgrading, the telescope could obtain four images simultaneously, in the r, i (optical) and J/H (infrared) bands. The OCS for the telescope is designed based on our framework, including scheduling and device control (mount/dome/focuser). The communication is based on the TCP/IP protocol. Control for the four detectors is designed by the manufacturer. The special aspect of this system is that we need to monitor the state of all the detectors in scheduling, which could be solved simply by redefining the state of detectors. Meanwhile, we need to support chopping and nodding for infrared observations. Due to the flexibility and stability of FSM, the chopping and nodding just amounts to a small change in target position during observing.

The 1 m wide-field telescope, built in 2012, is a prime-focus telescope with a field of $1^\circ \times 1^\circ$. The telescope supports ASCOM. The detector is controlled by software under Linux. The dome is controlled by a Programmable Logic Controller. The detector and the dome provide communication with OCS by user defined network protocol. The OCS we designed is divided into three parts, including the telescope agent, CCD agent and scheduling. The telescope agent can connect with the telescope control system (including mount, filter wheel and focuser) by ASCOM protocol, and dome control system by UDP protocol. To apply our framework, we need to redefine the state and event of the mount using a definition from ASCOM. The CCD agent links with the CCD control system under Linux by the TCP/IP protocol. Scheduling, designed under our framework, is responsible for the whole procedure of observations; it is linked with two agents by the UDP protocol. A particularity of this telescope is that the telescope needs to wait for reading-out to decrease interference on circuitry in the detector. The OCS is based on the original software for inheritance.

4 CONCLUSIONS

Besides presenting a complete and uniform framework, in this paper we focus more on its flexibility and inheritance. The former can provide possibilities for the framework to be widely used in various telescopes. The latter can offer the best opportunities for using existing resources. Therefore, the general framework presented in this paper is more suitable for existing small and mid-sized telescopes. This framework had been partly applied in different existing telescopes in China. In practical application, we did not change any hardware in the original system to suit our framework. Rather, we just added some software interfaces that can work via a network. However, the interface needs the support from the manufacturer of devices or engineers based at the observatory. Therefore, the design of general interfaces for hardware needs to be applied in our general framework.

References

Bailyn, C., Clemens, C., Johnson, J., et al. 2007, Report of the Committee for Renewing Small Telescopes for Astronomical Research

Castro-Tirado, A. J. 2011, Acta Polytechnica, 51, 16

Denny, R. B. 2004, Society for Astronomical Sciences Annual Symposium, 23, 35

Hanisch, R. J., Farris, A., Greisen, E. W., et al. 2001, A&A, 376, 359

Kubánek, P. 2010, Advances in Astronomy, 2010, 902484

Lin, Q., Lu, X., & Jiang, X. 2013, New Astron., 21, 33