



The STELLA control system

T. Granzer*, M. Weber and K. G. Strassmeier

Leibniz Institut für Astrophysik Potsdam, An der Sternwarte 16, D-14482 Potsdam, Germany

Abstract. The STELLA autonomous observatory comprises two different telescopes with different instrumentation, explicitly an imaging-photometer and a high-resolution spectrograph. This different instrumentation also demanded for different control software, which led to the decision to design a generic solution to robotic observing strategies, which not only can handle the existing hardware, but could also incorporate new hardware or observing strategies at relative ease. This article is about our solution.

Keywords : instrumentation: photometers – instrumentation: spectrographs

1. The STELLA autonomous observatory

STELLA (short for STELLar Activity) is a fully autonomous observatory located at Izaña observatory in Tenerife, Spain. Izaña observatory is located at longitude $16^{\circ}30'35''\text{W}$ and at latitude $28^{\circ}18'00''\text{N}$ at an altitude of 2390m above sea level. STELLA consists of two independent, 1.2m telescopes, each of them serving a single instrument to avoid the necessity of instrument change. Both telescopes have been manufactured by Halfmann Telescope in Augsburg, Germany, and are modern az-alt telescopes with a rather high slewing speed of $10^{\circ}/\text{sec}$. The observatory housing was finished in spring 2002, well before the telescopes had been delivered. It basically consists of two roll-off roof-halves, which are driven by two crane motors each. The early completion of the building left ample time for testing the reliability of building operation whose main purpose is a reliable protection of the instruments during bad weather periods. The meteorological observing conditions are measured by two separate weather stations. So far, the building automation has never failed to protect the instruments during harsh weather phases. All of the algorithms developed in the beginning are still in place unchanged, the only addition to the meteorological system has been the addition of a dust meter to allow the detection of low-altitude Sahara dust, a rather common phenomenon on the Canaries known as Calima.

*email: tgranzer@aip.de

Stella-I, which is an f/8 Cassegrain system with two available Nasmyth ports was delivered to Izaña in November 2004. This telescope is optimized for wide field-imaging with an field-of-view of 24 arc minutes, but was originally feeding the STELLA Echelle Spectrograph (SES), which achieved first light on this telescope in September 2005. An adapter was constructed for the second Nasmyth port (the one without the derotator), which contained the fiber input to the spectrograph, a focus pyramid and a gray beam-splitter to divert 4% of the light to a small, uncooled KAF-0402 detector which was used for acquisition and guiding. This system is described in more details in Strassmeier et al. (2004).

In late December 2005, the second telescope, STELLA-II has been delivered. STELLA-II is a very simple optical system with an spherical mirror at f/8.4. It has a single focal station at F1, where light is picked up with a fiber, which, in turn, is connected to the SES.

The formal inauguration of the observatory took place in May 2006, and the SES science demonstration program was started soon thereafter. The second instrument, a wide-field imaging photometer termed WiFSIP (Wide Field Stella Imaging Photometer) was completed not before April 2009. It underwent heavy testing on an 80-cm telescope clone of STELLA-I, called RoboTel located at the AIP campus in Babelsberg. WiFSIP features in brief:

- 4kx4k ITL chip with $15\mu\text{m}$ pixel and support four-amplifiers readout
- The useable field-of-view is 22 arc-minutes squared.
- Johnson UBV(RI)_C, Sloan u'g'r'i'z' filters, and a full Strömrgren set uvby including H β wide and narrow.
- H α photometry with a wide (FWHM=18nm) and a narrow filter (FWHM=3nm).

It was finally delivered to Tenerife in April 2010, where it was mounted on STELLA-I. At the same time, the spectrograph input fiber was moved over to STELLA-II achieving 'second first light' in May 2010. Since then, both telescopes are operating, but further updates are already planned. The spectrograph will get a new, more efficient camera in late September 2011, and the CCD detector in WiFSIP will be replaced due to performance problems with the current chip.

1.1 Current status of the imaging telescope

Due to severe space limitations, STELLA-I could not be equipped with an off-axis guiding system, which was originally anticipated to fit to the side of the main WiFSIP camera. Instead, a piggy-back mounted auxiliary telescope is used for acquisition and guiding. This auxiliary telescope is a 15cm refracting telescope equipped with a KAF-3200 detector. Differential bending between main and auxiliary telescopes amounts to 136 arc seconds in elevation between horizontal and vertical pointing. A

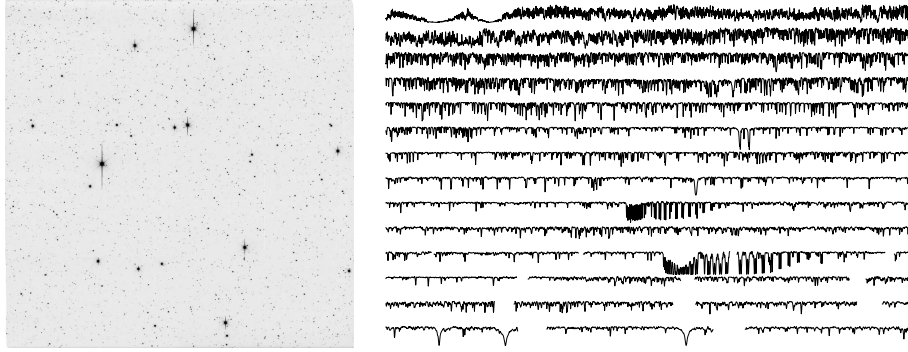


Figure 1. To the left, a single image of NGC 7092, exposure time 40sec in Sloan r'. Image taken on 16th June, 2010. To the right, a stacked spectrum of HD 1.

differential pointing model to compensate for this differential bending was derived with an original RMS of 2 arc seconds. Disappointingly it degraded after half a year to currently roughly 45 arc seconds RMS. This is acceptable for short exposure times up to 25 minutes, depending on pointing direction, but rules out very long exposures.

The current calibration schema relies on sky-flats only, but an installment of an additional dome flat is planned. In each twilight, three filters can be measured at ten exposures each. With a total of 18 filters, this allows for repeated observations of flat fields with an average cycle period of three days. See 3.1.2 for more details on the flatfielding process. Prior to the twilight flat, more precisely in the thousand seconds before the sun sets or rises, bias frames, totaling roughly 30 per twilight, are obtained. Though currently not used in the reduction process, again 1000 seconds before the start of the bias exposures, dark exposures are obtained. Regular science frames start at an sun height of -14° , but non-standard observations (time critical or targets of opportunity) may commence at any time the sun is at least 6° below the horizon. From first light until the time of this writing (August 2011) a total of 78335 science frames has been obtained.

The first science program, a monitoring program on open stellar clusters is not affected by guiding errors due to the rather short exposure times. Fig. 1, left-hand side shows a full-frame exposure of one of the target clusters, NGC 7092. The image was taken in Sloan r' at an exposure time of 40sec. More than 7000 sources have been detected in this frame, at an average FWHM of 1.3 arc seconds.

1.2 Current status of the spectrographic telescope

SES is a bench-mounted, white-pupil spectrograph with a fixed wavelength range of roughly 380-860nm (Weber et al. 2008). Fig. 1, right hand side, shows a combined spectrum of HD 1, an object observed during the commissioning phase (Strassmeier et al. 2010). SES is fed with a $50\mu\text{m}$ fiber, allowing for a resolution of $R=\Delta\lambda/\lambda \approx$

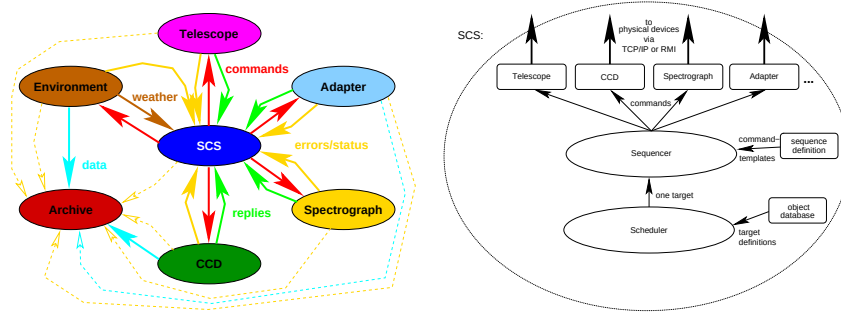


Figure 2. In the left panel, a sample generic schema in an automated telescope is shown. SCS denotes the Site Control System, i.e. the central node, which is further split up in the right panel. Other parts of the system, like Telescope refer to peripheral devices, which may or may not be present in a concrete system. The right panel shows generic action of the SCS. From an object database, a scheduler picks the current target and passes it over to the sequencer, which, in turn, fills a provided observing sequence template with missing definitions from the target. This yields an actual flow of commands, which are handled by small communication helper entities, visible in the top row of boxes. Note that a generic implementation makes no assumption on the number of physical devices, but decides on run-time how many communication helper objects are required (Figure from Granzer et al. (2001)). For details, please refer to the text.

55000. For stability, it is located in a temperature controlled environment in a separate room below the telescope bay. Heat foils in the room and heating elements on the spectrograph cover allow us to keep the temperature on the optical table at an average of 19.6°C, with an RMS of 0.68°C, which is a little bit higher than the anticipated 0.5°C, but still allows radial-velocity determination with a long-term error down to $\approx 50\text{m/s}$ (see Weber & Strassmeier (2011)).

The calibration sequence on the spectroscopic instrument starts with a single Thorium-Argon spectrum, followed by a block of 20 bias exposures. Then, another Thorium-Argon spectrum follows. Flat-field is then calibrated by obtaining 60 single exposures. Finally, just before the roof opens, another Thorium-Argon spectrum is obtained. Calibrations are done with dedicated halogen or Thorium-Argon lamps. In total four lamps are present to allow on-the-flight replacement if one burns out. The proper calibration lamp is selected with a linear stage in a light-tight calibration box residing in the main electronics room of the observatory. Its light is fed with a $200\mu\text{m}$ calibration fiber up to the science fiber entrance.

Acquiring on the target star is a two-staged process. A piggy-back telescope identical to the one on STELLA-I is used to raw-acquire the target star. Within the fiber unit at F1 of STELLA-II, a small Unibrain-520b video camera is used as a fiber-viewing camera. This camera features a 640×480 pixel progressive scan CCD at a pixel size of $7.4\mu\text{m}$. It has a tunable exposure time from $1\mu\text{sec.}$ up to 65sec. , which allows acquiring and guiding of stars from 0^{m} down to 14^{m} . After the coarse acquisi-

tion with the auxiliary telescope has completed, the fiber viewing camera takes over. Its field-of-view spans roughly 2×1.5 arc minutes, enough to accommodate differential bending effects between the main and the auxiliary telescope. Guiding is done on the light spilled over at the fiber entrance using a simple center-of-gravity method.

2. Principles of a generic software layout

STELLA as an autonomous observatory covers two different observing techniques, spectroscopy and imaging. Though both have some tasks in common, it was very clear from the beginning that a modular, generic schema would provide the best solution for the software architecture. Ideally the observing strategies would not be hard-coded into the system, but would be made available to the system in a sort of plug-and-play fashion. As we expect that observing strategies will grow more and more complex with the experience gained, we did not want to hinder ourselves with a rigid software, but designed a system that is able to grow with time. In the following subsections we will lay the foundation of a generic schema and will then proceed to the concrete implementation process.

2.1 A generic schema

Decomposing the complex system of a robotic telescope starts with a first step of simplification, a grouping of logical and/or physical subsystems and interlinking them. An approach that allows for a growing schema without even the need to restart the system, is a simple publisher-subscriber approach, where new software modules can enter a running system by simply subscribing to nodes that provide special services. The hierarchical order in a system is then maintained by the services provided and the order how the subsystems are interlinked. Consider Fig. 2, left hand side. Arrows mark here one publisher/subscriber pattern, the arrow originating from the provider and ending at the subscriber. It is immediately apparent that commands form such a publisher/subscriber circuit, defining in a sense that the main publisher of commands is the central node, the Site Control System (SCS). Whenever a command is issued, its subscriber should publish replies to these commands. These replies should at least carry information on the completion of the command previously issued, but it may be beneficial to split the level of replies further into acknowledge and done messages, which is actually done in the STELLA system. The protocol was constructed in a fashion similar to the message exchange system working on the Liverpool Telescope (Steele et al. 2000). Granzer et al. (2001) gives more details on the command exchange protocol.

These two principal message circuits should already suffice to build a working system – at least if no errors occur. Errors (and in fact also status messages which can be considered harmless errors) form a third circuit, here all components act as publishers. Errors are handled centrally at a single error subscriber within the SCS. Weather messages are depicted as a fourth circuit in Fig. 2, but can in principle be

regarded as special error messages as their occurrence and handling is in fact very similar to errors. Finally, the scientific data is depicted as the data circuit here. For the entire system to work, it is sufficient to introduce a rather unspecific data sink called archive here. Omitted in the figure are possible further command/reply cycles. For example a guider camera sending positional updates to the telescope or an imager requesting focus changes. In the picture this would simply add another publisher of commands, in the case of an auto-guider it resides inside the adapter, and the telescope additionally subscribing to this publisher. A remote or even manual observing mode is implemented at comparable ease by replacing the SCS by a command publisher that gets its input directly from a (remote) observer. If implemented carefully, a change of publishers can also occur without a restart of the entire system.

To mention it again, Fig. 2 only depicts a possible layout. Far simpler systems, consisting only of a telescope and a CCD are possible, but also much more complex schemas including special calibration devices or a staged acquiring system with more than a single adapter can be incorporated seamlessly in the above picture.

2.2 A generic description of the central node

A next step in a generic description would be the splitting of the SCS. This is outlined in the right hand panel of Fig. 2. It starts with a pool of possible targets, which may be present as simple files or more sophisticated objects in a database. From this pool of possible objects, an algorithm chooses one to become the target that is to be observed. Many different algorithms have been described for this scheduling problem, see e.g. Granzer (2004). STELLA uses dispatch scheduling like most robotic or autonomous telescopes (Fraser & Steele 2004).

After a single target has been selected, it is passed over to an entity that performs the actual observation. This is labeled sequencer in the figure, but might also be termed work-flow engine. Basically, the selected target decides which observing strategy should be executed for it. This strategy is available to the sequencer as a command-template list, missing values are now filled in from the target and from current observing conditions to yield an actual command sequence. At the moment we leave the command template unspecified, but in section 3.1 we introduce the STELLA solution. The sequencer is responsible for publishing this commands at proper timings to its internal command publishing entities, see next section. Again, the number of publishers may vary with application, but the principle logical chain may be retained.

3. Defining a single observation

We will now try to decompose the task of making a single observation into smaller logical tasks, basically describing the work of the sequencer. Consider, for instance,

you want to image a certain region in the sky. You are safe assuming that the scheduler already has been taking care of visibility issues, so that you can immediately start with moving the telescope to the specified location. *At the same time* you can make the appropriate color filter available to the system, in many cases this will mean that you rotate the filter wheel. This is also a good time to check the status of any calibration lamps. Are they turned off? *After* the telescope has reached the target position, you might want to fine acquire your target, and/or check the focus. If acquisition is independent from the focus, like in our case with an auxiliary telescope, you can already start a focusing sequence *in parallel* to the acquisition, or, if a good focus is required for the acquisition phase, you may only start the latter *after* your focus routine has finished. After the acquisition has been successful to the degree requested, the auto guider can start. Now you are ready to take your image, but first make sure that the filter wheel has achieved its desired position (if you need the filter in place for focus determination you of course have to ensure that already before focusing the instrument) and the read-out of the last image has been finished.

3.1 Decomposing into parallel and sequential tasks

From this general example, one can immediately draw some conclusions:

- Any work-flow engine must have the possibility to execute tasks in parallel as well as pure sequential order.
- Conditional sentences (if-then) are required to keep task execution aligned.
- A task should define starting and completion conditions.
- The additional possibility to execute in loops, bound to some Boolean trigger condition, might help to keep the work-flow readable.

One can now introduce a (meta) programming language that enforces these rules. In STELLA we decided to base this language on XML-markup, which gives you all the variety of available XML-editors and syntax checkers. The principle distinction between parallel and pure sequential tasks is introduced with the Task and Step elements: Tasks may run in parallel, but steps are enforced to be executed one after the other. The order in which tasks are executed is only governed by their appropriate Condition clauses, as soon as it is fulfilled, the task commences. It is considered complete, when its Complete clause gets true. Each task can be composed of different Boolean operations, denoted in True and False elements, but also, more importantly, of the commands that should be passed to appropriate subscribers. This is achieved via one or more Command elements, their names are directly linked to Java class names and are thus resolvable at run-time. Replies to these commands are identified with prefixes DONE or ACK (followed by the running number of the acknowledge message). If a sequence consists of equal commands, they are discriminated further by their ref attribute, which has to be attached to complementary DONE or ACK

```

1 <!DOCTYPE Sequence SYSTEM "/z/stella/stella/xml/target.dtd">
2 <Sequence type="single">
3   <Step>
4     <Task id="move">
5       <Condition>
6         Roofopen
7       </Condition>
8       <Command id="main">
9         MOVE_TELESCOPE
10      </Command>
11      <Command>
12        FLAT_FIELD_OFF_CALIBRATION
13      </Command>
14      <Command>
15        TH_AR_OFF_CALIBRATION
16      </Command>
17      <Command>
18        MOVE_HOME_CALIBRATION
19      </Command>
20      <Command id="sets" ref="suresky">
21        SELECT_FILTER_CALIBRATION
22      </Command>
23      <True>
24        OnGroup
25      </True>
26      <False>
27        GuidingTelescope
28      </False>
29      <False>
30        GuidingAdapter
31      </False>
32      <Complete>
33        DONE_MOVE_TELESCOPE:main &AND; DONE_FLAT_FIELD_OFF_CALIBRATION &AND; DONE_TH_AR_OFF_CALIBRATION &AND; DONE_MOVE_HOME_CALIBRATION &AND; DONE_SELECT_FILTER_CALIBRATION
34      </Complete>
35      <Task id="find">
36        <Condition>
37          Task move
38        </Condition>
39        <Command id="main">
40          FIND_AUXILIARY
41        </Command>
42        <True>
43          OnTel
44        </True>
45        <Complete>
46          DONE_FIND_AUXILIARY:main
47        </Complete>
48      </Task>
49      <Task id="get1">
50        <Condition>
51          Task find &AND; DONE_MOVE_TELESCOPE:main:1000
52        </Condition>
53        <Command>
54          START_GUIDE_TELESCOPE
55        </Command>
56        <True>
57          GuidingTelescope
58        </True>
59        <Complete>
60          DONE_START_GUIDE_TELESCOPE
61        </Complete>
62      </Task>
63      <Task id="acquire">
64        <Condition>
65          Task get1
66        </Condition>
67        <Command id="main">
68          ACQUIRE_AUXILIARY
69        </Command>
70        <Complete>
71          DONE_ACQUIRE_AUXILIARY:main
72        </Complete>
73      </Task>
74    </Task>
75  </Step>
76  <Task id="autofocus">
77    <Condition>
78      Task_acqu2nd
79    </Condition>
80    <Command>
81      AUTO_FOCUS_ADAPTER
82    </Command>
83    <Complete>
84      DONE_AUTO_FOCUS_ADAPTER
85    </Complete>
86  </Task>
87  [...]
88 </Step>
89 <Step>
90   <Task id="guide">
91     <Condition>
92       Task_recent
93     </Condition>
94     <Command id="guider">
95       START_GUIDING_ADAPTER
96     </Command>
97     <Command>
98       STATUS_TELESCOPE
99     </Command>
100    <True>
101      GuidingAdapter
102    </True>
103    <Complete>
104      DONE_START_GUIDING_ADAPTER:guider+100000 &AND; DONE_STATUS_TELESCOPE
105    </Complete>
106  </Task>
107  <Task id="expose">
108    <Condition>
109      Task_guide &AND; &NOT; CCDBusy
110    </Condition>
111    <Command id="exposure">
112      SCIENTIFIC_CCD
113    </Command>
114    <True>
115      CCDBusy
116    </True>
117    <Complete>
118      ACK1_SCIENTIFIC_CCD:expose
119    </Complete>
120  </Task>
121  <Task id="normal">
122    <Condition>
123      ACK1_SCIENTIFIC_CCD:expose
124    </Condition>
125    <False>
126      CCDBusy
127    </False>
128    <Complete>
129      &NOT; CCDBusy
130    </Complete>
131  </Task>
132 </Step>
133 <Step>
134   <Task id="cleanup">
135     <Condition>
136       Task_expose
137     </Condition>
138     <Command ref="cleanadapter">
139       STOP_GUIDING_ADAPTER
140     </Command>
141     <False>
142       GuidingAdapter
143     </False>
144     <Complete>
145       DONE_STOP_GUIDING_ADAPTER:cleanadapter
146     </Complete>
147  </Task>
148   <Task id="cleantel">
149     [...]
150  </Task>
151 </Step>
152 </Sequence>

```

Figure 3. Excerpt of the sequence template that does a single spectroscopic observation. Omitted are tasks that do a fine alignment on the fiber-viewing camera.

expressions. Commands that need further information either from the target that is to be observed, or from the environmental condition, may specify appropriate `id` or `parameter` attributes. An `init` attribute is a possibility to prescribe fixed parameters to a command. Loops are introduced at the step level: An optional `while` phrase enforces repetitive execution of this step as long as the while condition is fulfilled.

We now want to provide two concrete examples of observing rules coded in XML grammar to make the concept more tangible and to provide a guideline for the interested user to code his/her own strategy in a similar way. The entire software package (100% Java) can be obtained on request from TG.

3.1.1 Example: A Spectroscopic Observation

In Fig. 3 you see a comprehensive excerpt of our definition of a single spectroscopic observation. It is composed of three main steps, which might be identified as preparation, exposure, and cleanup. The first step consists of eight tasks, five of them are shown. On the first invocation, only the task `move` will have a condition that evaluates to true, all other tasks depend on completion of some other tasks. Within the `move` task, the telescope is moved (command with `id` `main`, line 8), all calibration lamps are turned off, the calibration lamp exchange unit is retraced and the calibration selection

wheel is checked for science throughput. Global triggers like the guiding state of the telescope and the adapter are reset. `OnGroup` is a trigger that counts the time spent on science group observation. It is (re)started here (line 24). After moving the telescope, we upload the expected field-of-view to the auxiliary telescope (line 41), and put the telescope in the guiding-enabled mode (line 55). Because it turned off operations until after the movement of the telescope has been completed, the telescope might need some extra time to accept further commands, a delay of 1000ms is added to the commencing condition of this task (line 52). After the telescope is in guiding mode, we acquire using the auxiliary telescope in line 69. After a successful acquisition, we repeat this process with the fiber-entrance viewing camera, these tasks are omitted here. Since we need the target star to be visible in the fiber-viewing camera for proper focusing of the telescope, we have to wait for the completion of this second acquire (line 78). As the focusing unit consist of a small linear stage right at the fiber entrance, the focusing command is sent to the adapter (line 81). This completes the preparation step, and the exposure step follows with a start of the guiding loop, again targeted at the fiber-viewing camera and thus sent to the adapter. Its condition in line 92 refers to one of the omitted tasks in the previous step, but again it is the only one that is true in the first evocation. After the guiding has started, we wait another 100 seconds to even-out the transient response of the guider loop. Tests have shown that the transient response of the guider falls short most of the times after approximately 20 seconds, but as exposure times in the spectrograph are normally in the order of an hour, this enlarged waiting period seems well justified. It helps ensuring that the fiber is evenly illuminated. The actual exposure is started with the command in line 112. Note that it not only requires the guiding task to be completed, but also that the spectrograph CCD is not currently busy with read-out of the old exposure. As soon as the CCD acknowledges that the shutter has been closed (line 123), the exposure step is finished and in the final step some clean-up procedures are executed to leave the system in a state that *any* sort of new observation can be started.

3.1.2 *Example: Sky-flat observations*

A more complex example of an observing work-flow is shown in Fig. 4, which is the sequence used for obtaining sky-flats during dusk twilight. Here we start with two tasks in parallel, `prep`, which selects the color filter and also restricts the read-out window on the science CCD to a small area around the center, and either task `point` or task `repoint`, depending whether this is the first flat field in this evening ('point') or whether this is the second or third flat sequence ('repoint'). In the first case, the telescope is moved to a zenith distance of 25° in the anti-solar azimuth, tracking enabled. This is close to the point of the twilight sky with vanishing illumination gradient (Chromey & Hasselbacher 1996). Experience showed that pointing the telescope to this sky position takes up to two minutes, even if the telescope is already on the spot. Thus, on recurrent flat observations we do not re-position the telescope but only erase the accumulated offsets during the previous flat-field sequence (task

```

1 <!DOCTYPE Sequence SYSTEM '../stella/stella/xml/sequence.dtd'>
2 <Sequence type="duskskyflat">
3   <Step>
4     <Task id="prep">
5       <Condition>
6         Roofopen &AND; InitTelescope &AND; Wfsip_UP
7       </Condition>
8       <Command id="filter">
9         SELECT_FILTER_CALIBRATION
10      </Command>
11      <Command parameter="FlatFrame" ref="miniwindow">
12        GEOMETRY_CCD
13      </Command>
14      <Broadcast>
15        stella.error.AccumulationStop
16      </Broadcast>
17      <Complete>
18        DONE_SELECT_FILTER_CALIBRATION:filter &AND; DONE_GEOMETRY_CCD:miniwindow
19      </Complete>
20    </Task>
21    <Task id="point">
22      <Condition>
23        AND: Offlat
24      </Condition>
25      <Command parameter="FlatPointing" ref="postel">
26        MOVE_TELESCOPE
27      </Command>
28      <Complete>
29        DONE_MOVE_TELESCOPE:postel
30      </Complete>
31    </Task>
32    <Task id="repoint">
33      <Condition>
34        Offlat
35      </Condition>
36      <Command init="75" ref="rot000">
37        DEROTATOR_TELESCOPE
38      </Command>
39      <Command init="0.0 0.0" ref="repostel">
40        TOTAL_OFFSET_TELESCOPE
41      </Command>
42      <Complete>
43        DONE_TOTAL_OFFSET_TELESCOPE:repostel &AND; DONE_DEROTATOR_TELESCOPE:rot000
44      </Complete>
45    </Task>
46    <Task id="offzenith">
47      <Condition>
48        Task_point || Task_repoint
49      </Condition>
50      <True>
51        Offlat
52      </True>
53      <True>
54        Task_point
55      </True>
56      <True>
57        Task_repoint
58      </True>
59      <Complete>
60        Task_point &AND; Task_repoint
61      </Complete>
62    </Task>
63    <Step>
64      <Step while="(FlatExposureTime <= MinimalExposure || SequenceLoop < 2) &AND; Solve
65        ight <= -12.0">
66        <Task id="exposure_normal">
67          <Condition>
68            AND: COSUSY &AND; FlatExposureTime <= MaximalExposure &AND; FlatExposureTime
69            <= MinimalExposure
70          </Condition>
71          <Command id="nstore" parameter="FlatExposureTime" ref="nsearch">
72            SCIENTIFIC_CCD
73          </Command>
74          <Code>
75            CODE_EXPOSE
76          </Code>
77          <True>
78            Task_abort0
79          </True>
80          <True>
81            Task_exposure_bright
82          </True>
83        </Task>
84      </Step>
85      <Task id="nsearch">
86        <Condition>
87          FlatExposureTime <= MaximalExposure
88        </Condition>
89        <Broadcast>
90          stella.error.SkyFlatAbort
91        </Broadcast>
92        <False>
93          Offlat
94        </False>
95        <Complete>
96          AND: Offlat
97        </Complete>
98      </Task>
99    </Step>
100  </Step>
101  <Step>
102    <Task id="fullframe">
103      <Condition>
104        Task_prepare
105      </Condition>
106      <Command parameter="FlatFrame" ref="fullwindow">
107        GEOMETRY_CCD
108      </Command>
109      <Complete>
110        DONE_GEOMETRY_CCD:fullwindow
111      </Complete>
112    </Task>
113    <Step while="(SequenceLoop < 5 &AND; SolHeight <= -6.0">
114      <Task id="flatexp000">
115        <Condition>
116          AND: COSUSY &AND; FlatExposureTime <= MaximalExposure
117        </Condition>
118        <Command id="store" parameter="FlatExposureTime" ref="found000">
119          SCIENTIFIC_CCD
120        </Command>
121        <True>
122          CODE_EXPOSE
123        </True>
124        <True>
125          Task_abort1
126        </True>
127        <Complete>
128          DONE_SCIENTIFIC_CCD:found000
129        </Complete>
130      </Task>
131      <Task id="dithoff000">
132        <Condition>
133          Task_flatexp000
134        </Condition>
135        <Command init="0.0 0.0035" ref="prep000">
136          INITIAL_OFFSET_TELESCOPE
137        </Command>
138        <Complete>
139          DONE_INITIAL_OFFSET_TELESCOPE:prep000
140        </Complete>
141      </Task>
142      <Task id="abort1">
143        <Condition>
144          FlatExposureTime <= MaximalExposure
145        </Condition>
146        <Broadcast>
147          stella.error.SkyFlatAbort
148        </Broadcast>
149        <False>
150          Offlat
151        </False>
152        <Complete>
153          AND: Offlat
154        </Complete>
155      </Task>
156    </Step>
157  </Step>
158 </Sequence>

```

Figure 4. Excerpt of the sequence template that does a single sky-flat observation during dusk twilight. We omitted tasks defining the same sequence with opposite derotator pointing.

repoint). As the completion of either of this two tasks suffices, we have a dummy `offzenith` task, line 46, that artificially sets the not-commenced task true.

Once the optimal sky-position is reached, we have to check if the current sky-level brightness is within the desired range. This is done with the step started in line 64, which is executed as long as the derived optimal flat exposure time is less than the minimal exposure time allowed. How is the optimal flat exposure time derived? Starting from a value depending on solar height and filter used, we take a short test exposure (remember, the read-out frame was set to a small window at the start of the sequence to minimize read-out time). The average light level of the exposure is computed and compared to the desired ADU level to gain an optimal exposure. If that exposure time is too short (i.e. the sky is still too bright), we simply wait and repeat the test measurement with the shortest allowed exposure time. If this exposure time is higher than the maximum exposure time allowed (again, a per-filter time), we conclude that the sky is already too dark to obtain a good flat in the current filter. We generate an error (line 91), which the global error handler interprets as a target abort, i.e. the flat sequence is discarded and a new target is loaded. Only if the calculated exposure time is within the bounds of the minimum and maximum time allowed, we leave this loop and enter the next step, line 102, which sets the full frame in the science CCD again. The omitted tasks (line 101) deal with circumstances where clouds passing in front of the sky reverse the expected overall dimming of the sky back-

ground, i.e. they register cases where the sky suddenly becomes significantly brighter during dusk. The loop started in line 115 finally does the five flat field observations in the current filter. If the optimal exposure time surpasses the maximum time, we again abort the execution (line 149), otherwise we do an exposure (line 121), followed by a small (additional) offset of the telescope in altitudinal direction (line 138). After these five images have been completed, we use the derotator to turn the science camera upside-down and repeat the exposures. These steps are omitted in the figure.

4. Describing an astronomical target

Again, we want to analyze the ingredients necessary for a single observation to be successfully executed, but this time from the target side, i.e. which information a target must supply in general terms. This can be addressed by answering the questions, *when* a target should be observed, *how* it should be observed, and *where* the target is located on the sky plus its brightness and its surrounding. For the last part of the question, the target must provide its right ascension and declination in a defined epoch, plus, if available and necessary, proper motions to migrate from the stated epoch to the apparent epoch of the observations. To properly identify the source, not only its brightness, but also the brightness of surrounding sources (plus their exact position) must be provided, ideally in the filter the acquisition system is using. The *how* section must address the exposure time (or the desired S/N) along with other changeable system setups, like color filters, read-out frames or even guiding strategies (open or closed loop) and acquisition minimum accuracies. As mentioned earlier, the answer to the first question, *when* a target should be observed is the answer to the scheduling problem, which we won't cover in detail here, but just add a few remarks on the experience we gained with our system.

4.1 Principles of target selection

The dispatching concept of evaluating a globally or even target-based merit function and to concurrently pick up the target with the apparent highest merit is conceptually easy, but translating an astronomer's scheduling request into abstract figures of merit proved difficult, if not even impossible. We found it much easier to base target selection on a three-staged approach, with first filtering out obviously non-desirable targets and do only the final selection with a dispatcher. First, consider what basic requirements a target may have or may not have. While one target may require the observatory dome to be open, others, like some calibration targets may be valid targets even if the dome is closed. Most targets may need a functional guiding system, while some might not even need a movable telescope. These very basic considerations constitute our first layer. To be more specific, each target supplies a Boolean expression that defines its very basic requirements. After this basic filter, the target supplies additional constraints by stating certain parameter ranges it allows for external parameters, like the minimum height of the target, or the maximum acceptable seeing.

```

1 <!DOCTYPE Target SYSTEM "/z/stella/stella/xml/target.dtd">
2 <Target access="enabled" submitted="2010-03-30T09:24:07 UTC" type="single">
3   <TargetName>
4     35 Cnc preD1
5   </TargetName>
6   <Select>
7     [...]
8   <Requires>
9     Roofopen &AND; InitTelescope &AND; AuxAvailable &AND; GuiderAvailable
10  </Requires>
11  <Constraint overdue="500000" relax="1.0">
12    <Variable>
13      SolHeight
14    </Variable>
15    <Max>
16      -12.0
17    </Max>
18  </Constraint>
19  <Constraint overdue="0" relax="1.0">
20    <Variable>
21      AltTarget
22    </Variable>
23    <Min>
24      10.0
25    </Min>
26  </Constraint>
27  <Merit>
28    <Timeslot class="stella.xml.PerNightMerit">
29      [...]
30    </Timeslot>
31    <Gain class="stella.xml.AirmassMerit">
32      [...]
33    </Gain>
34  </Merit>
35 </Select>
36 <Setup id="expose">
37   <Instrument>
38     SES
39   </Instrument>
40   <Constant class="java.lang.Double">
41     ExposureTime
42   </ConstantName>
43
44   <Constantvalue>
45     3600.0
46   </Constantvalue>
47   [...]
48 </Setup>
49 <Object id="main">
50   <ObjectName>
51     35 Cnc
52   </ObjectName>
53   <Position>
54     <RA>
55       128.83102292
56     </RA>
57     <Dec>
58       19.59006278
59     </Dec>
60   </Position>
61   [...]
62   <PbRA>
63     35.66
64   </PbRA>
65   <PbDec>
66     -12.82
67   </PbDec>
68   <V>
69     6.55
70   </V>
71   <B-V>
72     9.681
73   </B-V>
74   <Position>
75     <FieldOfView>
76       <RA>
77         128.830915
78       </RA>
79       <Dec>
80         19.590026388888887
81       </Dec>
82     </FieldOfView>
83   </Object>
84   [...]
85 </Object>
86 </Target>

```

Figure 5. An example of a spectroscopic target XML file. Only the relevant parts are shown.

Such constraints easily filter out half of the targets, leaving the actual dispatching schema easier to control. How this is all put together is described in the next section.

5. Putting it all together

5.1 From observing templates to concrete observing sequences

For observational work-flows similar to the examples presented in the last section, some target-specific information has to be available to yield proper sequences. Again, if using XML-markup, one can benefit from existing helper applications. Consequentially, different groups came up with similar solutions, the most elaborated one may be the RTML markup language (Pennypacker et al. 2002; Hessman 2006). Though RTML may be the best choice for a general description of observations for a variety of different telescopes, like it was the proposed standard for the Heterogeneous Telescope Network (HTN, Granzer (2006)), we considered it not specific enough to serve our purpose of a target description for a *single* system.

5.1.1 Example: A single target

In Fig. 5, we show an excerpt of a spectroscopic target definition. The `Select` element comprises all three layers of target selection. Within the `Requires` element, we find a Boolean expression stating that this target requires an open dome (line 9, `Roofopen`), an initialized and thus working telescope, a working auxiliary camera, and a working guiding camera, which is the fiber-viewing camera. Only if these requirements are met, we additionally request, via two `Constraint` elements that the sun should be at least 12° below the horizon and that the target should be at least at

an elevation of 10° . Both constraints have to be fulfilled during the entire anticipated execution time, with the exception that at the end of the observation, a 500 sec. margin for the solar height is allowed (`overdue` attribute, line 11). If these two layers are surpassed, the target may enter dispatch selection, with its two merit functions, a slow-varying `Timeslot`, and a fast-varying `Gain`, both indicated by their Java class name. The time slot evaluates to zero if the specified number of observations in this night has been reached, otherwise it evaluates to one, while the gain is a function of air mass of the target which peaks at one as the target passes the meridian. How the target should be observed is stated in the `Setup` sections. For clarity, only a single setup that defines the exposure time is shown. Note that the `id` attribute on this setup equals the reference in the target sequence, compare to Fig. 3, line 111. The same analogy is found in the `Object` section of the target. Its reference `main` is matched by the sequence commands in lines 8, 40, and 69. If the sequencer (at run-time) encounters a command template that requires target input from a target setup section or an object section, the current target is searched for the requested `id` ('`exposure`' or '`main`' in this case). Depending on the command being rendered, either setup or object sections are searched. If the appropriate element could be found, the command-depending rendering process collects the information required for the specific command and completes it. For the `move` command in line 8 or the `acquire` command in line 69, only the main position of the target is queried, while for the `find` command in line 40, the entire field-of-view is used. Setup section like the `expose` setup are further scanned for the known constant names, in this case `ExposureTime` or `SignalToNoise`. If any of these are found, the command can be validly rendered and the sequence may continue.

References

- Chromey F. R., Hasselbacher D. A., 1996, *PASP*, 108, 944
Fraser S., Steele I. A., 2004, *SPIE*, 5493, 331
Granzer T., 2004, *AN*, 325, 513
Granzer T., 2006, *AN*, 327, 792
Granzer T., Weber M., Strassmeier K. G., 2001, *AN*, 322, 295
Hessman F. V., 2006, *AN*, 327, 751
Pennypacker C., Boer M., Denny R., et al., 2002, *A&A*, 395, 727
Steele I. A., Newsam A. M., Mottram C. J., et al., 2000, *SPIE*, 4011, 133
Strassmeier K. G., Granzer T., Weber M., et al., 2004, *AN*, 325, 527
Strassmeier K. G., Weber M., Granzer T., et al., 2010, *AN*, 331, 368
Weber M., Granzer T., Strassmeier K. G., et al., 2008, *SPIE*, 7019,
Weber M., Strassmeier K. G., 2011, *A&A*, 531, A89