AstroGrid-D

Deliverable

First Integration of Robotic Telescopes[1]

Deliverable	5.3
Authors	Working Group Resource Management for Grid Jobs
Editors	Frank Breitling, Thomas Granzer, Harry Enke
Date	August 31th, 2007
Document Version	1.0
Current Version	1.0
Previous Versions	0.9.5

A: Status of this Document

Deliverable 3 of working group 5.

B: Reference to project plan

This deliverable refers to the task TA V-3 *First Integration of Robotic Telescopes*, dealing with a generalized description of a robotic telescope as a GRID resource and with a general description of an astronomic observation that allows it to be deployed as a standard GRID resource.

C: Abstract

The grid integration of robotic telescopes as developed by AstroGrid-D is discussed. The architecture is based on the Globus Toolkit, the Remote Telescope Markup Language (RTML) and the central information service Stellaris. The usage of RTML as well as the submission and cancellation of observation requests is explained and exemplified for an observation with the robotic telescope STELLA-I. Moreover common software components are presented and their installations for the integration of new robotic telescopes is discussed.

[1]This work is part of the AstroGrid-D project and D-Grid. The project is funded by the German Federal Ministry of Education and Resarch (BMBF).

D: Changes History

Version	Date	Name	Brief summary
1.0.0	2007-8-31	Frank Breitling	final changes
0.9.5	2007-8-31	Harry Enke	corrections
0.9.1	2007-8-08	Frank Breitling	corrections by S. White, removing notification
0.9.0	2007-8-06	Frank Breitling	comments by H. Enke adding submission using WS GRAM, more details on introduction, templates, monitoring, notification, uid.
0.8.1	2007-8-01	Thomas Granzer	minor corrections
0.8.0	2007-8-01	Frank Breitling	major update to the submission mechanism using gsissh
0.5.0	2007-7-20	Frank Breitling	draft completion
0.1.0	2007-7-17	Frank Breitling	draft creation

E:

Contents

1 Introduction

The grid integration of robotic telescopes in AstroGrid-D is based on two technologies: the grid middleware of the Globus Tooklit [5] and the *Remote Telescope Markup Language* RTML [6] for the exchange of observation requests.

Grid technology on the one hand provides an ideal framework for a robotic telescope network. For example, through the concept of Virtual Organizations it provides solutions for the management of user access to grid resources, data and metadata. Hence, it allows the immediate access to computational and storage resources for data analysis. RTML on the other hand provides a protocol for the description of generalized observation requests and is a standard recommended by the Heterogeneous Telescope Network [6].

The architecture for the grid integration is illustrated in Fig. 1. It is built around the central information service, Stellaris, and the observer. Stellaris stores all metadata related to telescopes and observations providing immediate access to information relevant for scheduling new observations. Also the observer has a central role in being able to control all telescopes. Eventually the observer will be replaced first by a resource broker and later a metascheduler. A resource broker is a central service which accepts observation request, selects appropriate telescopes and delegates the observation requests accordingly. A metascheduler has additional functionality to resource broking. It can create and optimize individual schedules for simultaneous or sequential observations with multiple telescopes.

In this document the submission of RTML observation requests to a robotic telescope using the Globus Toolkit is discussed. As an example, the scheduling of an observation of the nearby stellar binary Gl 586A with the robotic telescope STELLA-I [9] is illustrated. In addition the software components and their installation is documented.

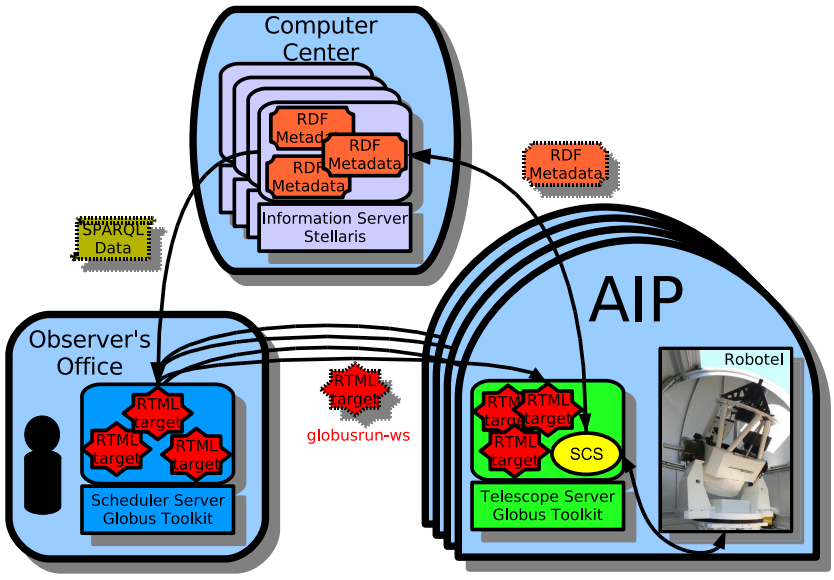

Figure 1: Architecture of the robotic telescope network of AstroGrid-D. Telescopes, observer and the information service "Stellaris" are the main components. Observation requests are submitted in RTML using the Globus Toolkit. Metadata is stored in RDF by Stellaris and retrieved through the SPARQL query language. The observer will eventually be replaced by a metascheduler.

2 RTML Descriptions of Telescopes and Observations

To integrate a telescope in the network, it is first registered with the information service to make its properties and status public to the network. Also each observation is registered with the information service providing an overview of the network activity. In compliance with the adopted standard all information regarding telescope resources and observation request is exchanged via RTML documents. Therefore, also a telescope's capabilities is represented in an RTML document.

To store the RTML metadata in Stellaris a transformation from RTML to RDF/XML [7] is necessary, which is the data format of Stellaris. Details about the description of telescopes in RTML, transforming RTML into RDF/XML using XSLT [3], uploading of this metadata to Stellaris and the retrieval of using the SPARQL[8] query language for RDF has been documented in [2]. The same procedure is applied here.

2.1 Template for Observations

Listing 6 shows the RTML document describing an observation of the nearby stellar binary Gl 586A with the robotic telescope STELLA-I. This document can also serve as a template for the description of new observations requests. For example, if a different target is to be observed, the <Target> tag has to be adapted. If a different instrument is to be used, the appropriate instrument has to be added to the <Schedule> tag. XML editors provide efficient user interfaces for a save editing of RTML templates based on schemas. A good example is the XML editor included in Eclipse [4]. In addition a comprehensive template should be provided for each telescope, which shows all available instruments and their correct configuration. It is planned to make the templates available through Stellaris.

2.2 Unique Identifier of Observations

As every RTML document, each observation request has a unique identifier. This unique identifier corresponds to a unique observation ID and is contained in the `uid` attribute of the RTML document. According to the RTML schema, this `uid` is composed by a prefix (`rtml://`), a telescope identifier encased in a reverse network notation (e.g. `de.aip.STELLA-I/`) and a document name conforming to the XML *NMTOKEN* definition, i.e. a set of characters without spaces or commas, but using slashes (e.g. `binaries/Gl586A`) is allowed. It is the observer's responsibility to assign the `uid`. If a `uid` was already assigned previously, the observation request will not be accepted and returned with an error. This increases the observers awareness of already scheduled or completed observations and of available data before a new observation request for a previously observed target is submitted. The information about the available data is accessible though Stellaris. However, if the same target is reobserved this should be reflected by the `uid`. Its structure can easily provide unique and meaningful identifiers. Ambiguities can always be avoided by appending e.g. the date or the name of the observer. The usage of automatically generated but meaningless numbers as `uids` is discouraged in favor of clear and structured observation schedules and histories.

3 Submitting Observation Requests

If an RTML observation request has been prepared, it is submitted to the telescope server through the grid middleware. The telescope server is responsible for controlling the telescope, scheduling of observation requests and the access to the data from the observation. For scheduling the telescope server has to be able to receive observation requests and to translate them into commands the telescope controller understands.

The first task is accomplished via the Globus Toolkit. It provides access to resources through Virtual Organization management and a suitable security layer. An installation of the Globus Toolkit at the telescope server is required. The submission of RTML observations is accomplished through GSI-OpenSSH command `gsiscp`.

The telescope server receives the observation request through the script `tel-schedule` and delegates the request to the scheduling program of the telescope specified in the `uid` of the RTML. The following example shows the submission of the observation request `Gl586A.rtml` to the telescope server `photon.aip.de`:

```
cat Gl586A.rtml | gsissh photon.aip.de tel-schedule
```

tel-schedule (Listing 1) will complete the following tasks:

1. validate the observation request with the RMTL schema
2. check that the *uid* has not yet been assigned
3. delegate the observation request to the telescope specific scheduler

   and if successful,

4. transform the RTML into the RDF format of the information service
5. upload this information to the information service

Listing 1: `tel-schedule`, the shell script for receiving RTML observation requests at the telescope server, validating the format and delegating it to the specific telescope scheduler script (Listing 2). On successful scheduling, the observation request is also submitted to the information service Stellaris.

```
#!/bin/sh

[ "$1" ] && {
    echo "Usage: cat observation.rtml|gsissh photon.aip.de tel-submit"; exit; }

cd /usr/local                   #start from telescope software folder

exec <&0                        #connecting to STDIN

while [ 1 ] ; do                #split concatenated files from `cat *.rtml ...`
    read || break               #read one line from STDIN to $REPLY
    in="$in\n$REPLY"            #paste these lines together

    #set $observation to the uid of the RTML
```

```
    [ "$observation" ] || observation=`\
           echo $REPLY|grep 'uid="rtml://'|sed 's/.*rtml:\/\///; s/\"//'`

    #read until RTML is complete
    [ "`echo $REPLY|grep '</RTML>'`" ] || continue;

    [ -f "dis$observation.rtml" ] && {
        echo -e "\e[0;31mError:\e[0m $observation already submitted";
        continue; }

    mkdir -p "`dirname $observation`"
    echo -e "$in"|sed 1d >| $observation.rtml
    tel=`echo $observation|sed 's/\/.*//'`

    #verify observation request based on the RMTL schema
    xmllint --noout --schema globotel/RTML-nightly.xsd $observation.rtml || {
        echo -e "\e[0;31mError:\e[0m $observation does not validate";
        continue; }

    #delegate the observation request to the telescope specific scheduler
    $tel/controller/schedule $observation.rtml || continue

    xsltproc globotel/xml2rdf.xsl $observation.rtml >$observation.rdf
    curl --silent --show-error --upload-file $observation.rdf \
        http://mintaka.aip.de:24060/context/observations/$observation
    observation=''; in=''
done
```

The third step requires an individual solution for each robotic telescope's interface. It is contained in a program which has a link called `schedule` pointing to. Listing 2 shows the corresponding program for STELLA-I. STELLA-I is controlled via a special XML syntax. Therefore the `schedule` program first transforms the RTML observation requests using XSLT and the stylesheet of Listing 7 into an XML document which the STELLA-I controller understands. The XML file corresponding to Listing 6 is shown in Listing 8. The XML document is then submitted to the STELLA-I controller.

Step four follows the procedure described in Sec. 2. Further details and the required the XSLT stylesheet (`xml2rdf.xsl`) is found in [2].

Listing 2: `schedule`, the script for translating RTML observation requests into telescope specific commands and directing them to the telescope controller - here for the case of STELLA-I.

```
#!/bin/sh

[ "$1" ] || { echo "Usage: $0 observation.rtml"; exit; }

STELLA_controller="`dirname $0`"
xml="`echo $1|sed 's/\.rtml$//'`.xml"

#transform the RTML into an XML document STELLA-I understands
xsltproc $STELLA_controller/rtml2stellaI.xsl $1 > $xml

#set the path to the libraries of the java controller
SJARS=`ls $STELLA_controller/ext/*`;SJARS=`echo $SJARS |sed 's/.jar\ /.jar:/g'`
export CLASSPATH=$STELLA_controller/class/:$SJARS:.:/usr/java/jre1.6.0

#schedule the observation with the java controller
java stella.util.SchedulerAccess MasterMind -add $xml
```

3.1 Network Observations

After an observation has been scheduled, the telescope scheduler conducts observations based on priority levels specified in the observation request. So if there is a demand for an immediate network observation, e.g. triggerd by a transient event, it can be conducted by submitting an observation request of sufficiently high priority to the desired telescopes. However, the priorities levels and the expected reactions still have to be defined and depend on the capabilities of the instrument.

3.2 Monitoring

All status information, e.g. information about the beginning of a new observation, is provided to Stellaris by the telescope controller. This provision of dynamic metadata and its retrieval will be covered in deliverable 2.7.

3.3 Data Management

Data of observations will become available through the corresponding telescope server, if not requested otherwise. For example, the data of the observation of Gl 586A will be found at `/usr/local/de.aip.STELLA-I/Gl586A/`. The data can be retrieved through the Globus Toolkit using `gsiscp`. Access rights are automatically set to the grid user and can be changed via `gsissh`.

3.4 Submission using WS GRAM

The submission of RTML observation requests can also be accomplished through the Grid Resource and Allocation Manager of the Globus Toolkit (WS GRAM). The motivation for the usage of WS GRAM is the compatibility to software developed for WS GRAM such as, job schedulers, monitoring tools and certain user interfaces.

The XML job template and its invocation is show in Listing 3. It transfers all observation requests located in the `/tmp/rtml/` folder to the users directory at the telescope server. At the server the `tel-submit-GRAM` program calls the `tel-schedule` program for each observation request as discussed above. However, the practical usage is complicated by the fact that a job status should not be "done" after the observation request was scheduled, but only after the observation has been completed. Since this can take many nights a persistant solution outlasting system reboots has to be worked out. However, all developments independent of WS GRAM can always be applied.

Listing 3: XML job template for the submission of RTML observation request using WS GRAM.

```
<!-- globusrun-ws -submit -S -F photon.aip.de -f tel-schdule-GRAM.xml -->

<job>
  <executable>/usr/local/globus/gtk/bin/tel-schedule-GRAM</executable>
  <fileStageIn>
    <transfer>
      <sourceUrl>gsiftp://photon.aip.de/tmp/rtml/</sourceUrl>
      <destinationUrl>file:///${GLOBUS_USER_HOME}/rtml/</destinationUrl>
    </transfer>
  </fileStageIn>
</job>
```

4 Canceling Observation Requests

The cancellation of observation requests only requires the unique observation ID and no transmission of any additional data. Canceling is also accomplished through GSI-OpenSSH command `gsissh` and a program `tel-cancel` at the telescope server. The syntax to cancel an observation with the `uid` `de.aip.STELLA-I/G1586A` is

```
gsissh photon.aip.de tel-cancel de.aip.STELLA-I/G1586A
```

tel-cancel (Listing 4) completes the following tasks:

1. remove the observation from the telescope schedule
2. if successful, update the information service

Listing 4: `tel-cancel`, the script for canceling observations based on the observation `uid`. The script passes the observation `uid` to the telescope specific canceling program (Listing 5).

```
#!/bin/sh

cd /usr/local

observation=$1; [ "$observation" ] || { \
    echo "Usage: gsissh photon.aip.de $0 'de.aip.STELLA-I/G1586A.rtml ...'";
    exit;
}

for observation in $@; {
    tel=`echo $observation|sed 's/rtml:\/\///; s/\/.*//'`

    #cancel the observation request with the telescope specific cancel program
    $tel/controller/cancel $observation
}
```

Listing 5: `cancel` is the program for canceling an observation request of a particular telescope based on the observation `uid`. Here the example of STELLA-I is shown.

```
#!/bin/sh

[ "$1" ] || { echo "Usage: $0 observation.rtml"; exit; }

observation=`echo $1|sed 's/rtml:\/\///; s/de.aip.STELLA-I\///'`

#set the java paths for the java controller
STELLA_controller="`dirname $0`"
SJARS=`ls $STELLA_controller/ext/*`;SJARS=`echo $SJARS |sed 's/.jar\ /.jar:/g'`
export CLASSPATH=$STELLA_controller/class/:$SJARS:.:/usr/java/jre1.6.0

#canel the observation
java stella.util.SchedulerAccess MasterMind -del $observation
```

5 Software Components and Installation

The grid telescope software has been developed as a thin layer on top of the Globus Toolkit with minimal requirements for installation and usage. Therefore the installation process is very simple. Only server components have to be installed at the telescope server. Additional client software beyond a Linux environment and the Globus Toolkit is not required. The software components fall into two categories. Standard components, common to all telescope servers, and components for each specific telescope server and telescope.

The standard components located in `/usr/local/globotel/` are:

1. `tel-schedule` (Listing 1)
2. `tel-cancel` (Listing 4)
3. `xml2rdf.xsl`
4. RMLT schema files

In addition the `tel-schedule` and `tel-cancel` programs are linked into the `GLOBUS_PATH` such that the programs are executable for and in the `PATH` of every grid user, i.e. observer.

The telescope specific components located in `/usr/local/de.aip.STELLA-I/` for STELLA-I are:

1. `schedule` (Listing 2)
2. `cancel` (Listing 5)
3. telescope controller and related components (e.g. `rtml2stellaI.xsd`)

The telescope specific components are expected in a directory corresponding to the telescope identifier located in `/usr/local/`. Also the RTML, RDF and XML files as well as the data files from the observations reside in this directory. The actual control over the individual telescopes is granted through the `schedule` and `cancel` programs. They have to be provided by the telescope administrators. All software presented here is available through the AstroGrid-D web page for robotic telescopes [1].

6 Conclusion

The grid integration of robotic telescopes developed in AstroGrid-D is based on the Globus Toolkit, the Remote Telescope Markup Language and the information Service Stellaris. The Globus Toolkit provides access to telescopes and data from observations through its Virtual Organization management. Description of telescopes and generalized observations is exchanged in RTML and through the central information service Stellaris. Observations requests are created through RTML templates and submitted to available telescopes. The information about submitted observations and the status of telescopes is stored in Stellaris. Software components for the access of integrated telescopes as well as the submission and cancellation of observation request have been presented. The software constitutes a small layer on top of the Globus Toolkit. It is easy to install and simple

to use. No additional software to the Globus Toolkit is required on the client side. The submission and cancellation of an observation has been illustrated for the robotic telescope STELLA-I. The development of improved interfaces and a metascheduler are the next steps which will reveal the full advantages of network observations.

A Listings

Listing 6: RTML observation request (Gl586A.rtml) with the observation `uid` designating this observation to telescope STELLA-I. This example also servers as a template for observations requests.

```
<?xml version="1.0" encoding="UTF-8"?>
<RTML version="3.1a" mode="request" uid="rtml://de.aip.STELLA-I/Gl586A"
   xmlns="http://www.rtml.org/v3.1a" xmlns:xsi="http://www.w3.org/2001/
       XMLSchema-instance"
   xsi:schemaLocation="http://www.rtml.org/v3.1a http://monet.uni-sw.gwdg.de/
       XMLSchema/RTML/schemas/RTML-nightly.xsd">
   <History>
      <!-- CREATION OF DOCUMENT -->
      <Entry timeStamp="2007-02-07T15:30:00">
         <Agent name="T. Granzer"/>
         <Description>RTML Example: Simple Example</Description>
      </Entry>
   </History>
   <Project ProjectID="gl586perihel">
      <Title>Gl 586A perihel</Title>
      <Abstract>This target is active 3 days before and after perihel</Abstract
          >
      <Contact>
         <Name>Strassmeier</Name>
         <Username>AIP</Username>
         <Communication>
            <Email>kstrassmeier@aip.de</Email>
         </Communication>
      </Contact>
   </Project>
   <Schedule status="active">
      <Spectrograph>
         <SpectralRegion>optical</SpectralRegion>
      </Spectrograph>
      <AirmassConstraint maximum="2"/>
      <DateTimeConstraint>
         <DateTimeStart value="2007-04-01T00:00:00"/>
         <DateTimeEnd value="2007-04-07T00:00:00"/>
      </DateTimeConstraint>
      <Priority>3</Priority>
      <Target name="Gl 586a">
         <Coordinates>
            <RightAscension>
               <Value units="hours">15.46933652</Value>
            </RightAscension>
            <Declination>
               <Value units="degrees">-9.3480694</Value>
            </Declination>
         </Coordinates>
         <TargetBrightness>
```

```
            <Magnitude>6.86</Magnitude>
            <Type>V</Type>
         </TargetBrightness>
      </Target>
      <Catalogue>
         <Target>
            <Coordinates>
               <RightAscension>
                  <Value units="hours">15.46934383</Value>
               </RightAscension>
               <Declination>
                  <Value units="degrees">-9.348610555555556</Value>
               </Declination>
            </Coordinates>
            <TargetBrightness>
               <Magnitude>6.95</Magnitude>
               <Type>V</Type>
            </TargetBrightness>
         </Target>
         <Target>
            <Coordinates>
               <RightAscension>
                  <Value units="hours">15.470066629</Value>
               </RightAscension>
               <Declination>
                  <Value units="degrees">-9.35839</Value>
               </Declination>
            </Coordinates>
            <TargetBrightness>
               <Magnitude>7.74</Magnitude>
               <Type>V</Type>
            </TargetBrightness>
         </Target>
      </Catalogue>
      <Exposure>
         <Value units="seconds">600.0</Value>
      </Exposure>
      <Observation id="single"/>
   </Schedule>
   <Telescope name="STELLA-I"/>
</RTML>
```

Listing 7: XSLT stylesheet (rtml2stellaI.xsl) for the transformation of RTML documents into the STELLA-I specific XML format.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- rtml2stellaI.xsl      is an XSLT stylesheet for converting RTML
                           observation requests into STELLA-I directives.

     Example Usage         xsltproc rtml2stellaI.xsl file.rtml

     Author                Frank Breitling (fbreitling at aip.de), Mar 26, 2007

     Copyright 2007        AstroGrid-D (http://www.gac-grid.org)

     Licensed under the Apache License, Version 2.0 (the "License");
     you may not use this file except in compliance with the License.
     You may obtain a copy of the License at
```

```
         http://www.apache.org/licenses/LICENSE-2.0

     Unless required by applicable law or agreed to in writing, software
     distributed under the License is distributed on an "AS IS" BASIS,
     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
     See the License for the specific language governing permissions and
     limitations under the License.
 -->

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:rtml="http://www.rtml.org/v3.1a" exclude-result-prefixes="rtml">
    <xsl:strip-space elements="*"/>
    <xsl:output method="xml" indent="yes" doctype-system="/z/stella/stella/xml/
        target.dtd"/>

    <xsl:template match="/rtml:RTML">
        <Target access="enabled" type="single">
            <TargetName>
                <xsl:value-of select="substring-after(substring-after(@uid,'
                    rtml://'),'/')"/>
            </TargetName>
            <xsl:element name="Abstract">
                <xsl:attribute name="proposal">
                    <xsl:value-of select="rtml:Project/@ProjectID"/>
                </xsl:attribute>
                <xsl:value-of select="rtml:Project/rtml:Abstract"/>
            </xsl:element>
            <File>
                <xsl:value-of select="substring-after(substring-after(@uid,'
                    rtml://'),'/')"/>.xml</File>
            <User>
                <xsl:value-of select="rtml:Project/rtml:Contact/rtml:Name"/>
            </User>
            <Email>
                <Address>
                    <xsl:value-of select="rtml:Project/rtml:Contact/
                        rtml:Communication/rtml:Email"/>
                </Address>
                <Notify>
                    <Onblock/>
                    <Onfirstpick/>
                    <Oncomplete/>
                </Notify>
            </Email>
            <Institution>
                <xsl:value-of select="rtml:Project/rtml:Contact/rtml:Username"
                    />
            </Institution>
            <History complete="false" max="500"> </History>
            <Exception for="this">
                <Delaymax>30</Delaymax>
                <Delaytime>600000</Delaytime>
                <Delay>NO_STAR_ON_ACQUIRE</Delay>
            </Exception>
            <Select>
                <Requires>Roofopen &amp;amp;amp; InitTelescope</Requires>
                <Constraint relax="1.0" overdue="0">
                    <Variable>Date</Variable>
```

```
            <From formatclass="java.text.SimpleDateFormat"
                formatpattern="yyyy-MM-dd'T'HH:mm:ss zzz">
                <xsl:value-of
                    select="rtml:Schedule/rtml:DateTimeConstraint/
                        rtml:DateTimeStart/@value"/>
                <xsl:text> UTC</xsl:text>
            </From>
            <To formatclass="java.text.SimpleDateFormat"
                formatpattern="yyyy-MM-dd'T'HH:mm:ss zzz">
                <xsl:value-of
                    select="rtml:Schedule/rtml:DateTimeConstraint/
                        rtml:DateTimeEnd/@value"/>
                <xsl:text> UTC</xsl:text>
            </To>
        </Constraint>
        <Constraint relax="1.0" overdue="500000">
            <Variable>SolHeight</Variable>
            <Max>-12.0</Max>
        </Constraint>
        <Constraint relax="1.0" overdue="0">
            <Variable>AltTarget</Variable>
            <Min>30.0</Min>
        </Constraint>
        <Merit>
            <Timeslot class="stella.xml.ConstantMerit">
                <Constant class="java.lang.Double">
                    <Constantname>impact</Constantname>
                    <Constantvalue>1.0</Constantvalue>
                </Constant>
            </Timeslot>
            <Gain class="stella.xml.ConstantMerit">
                <Constant class="java.lang.Double">
                    <Constantname>impact</Constantname>
                    <Constantvalue>30.0</Constantvalue>
                </Constant>
            </Gain>
        </Merit>
    </Select>
    <Setup id="set0">
        <Instrument>FilterWheel</Instrument>
        <Constant class="java.lang.String">
            <Constantname>Filter</Constantname>
            <Constantvalue>clear</Constantvalue>
        </Constant>
    </Setup>
    <Setup id="set1">
        <Instrument>SES</Instrument>
        <Constant class="java.lang.Double">
            <Constantname>ExposureTime</Constantname>
            <Constantvalue>
                <xsl:value-of select="rtml:Schedule/rtml:Exposure/
                    rtml:Value"/>
            </Constantvalue>
        </Constant>
        <Constant class="java.lang.Double">
            <Constantname>ExposureIncrease</Constantname>
            <Constantvalue>1.0</Constantvalue>
        </Constant>
    </Setup>
```

```
            <Object id="main">
                <ObjectName rec="0">Gl 586A</ObjectName>
                <Position>
                    <xsl:apply-templates select="rtml:Schedule/rtml:Target"/>
                </Position>
                <xsl:apply-templates select="rtml:Schedule/rtml:Catalogue/
                    rtml:Target"/>
            </Object>
        </Target>
    </xsl:template>

    <xsl:template name="Target" match="rtml:Target">
        <RA>
            <xsl:value-of select="15 * rtml:Coordinates/rtml:RightAscension/
                rtml:Value"/>
        </RA>
        <Dec>
            <xsl:value-of select="rtml:Coordinates/rtml:Declination/rtml:Value"
                />
        </Dec>
        <Epoch>2000.0</Epoch>
        <Equinox>2000.0</Equinox>
        <V>
            <xsl:value-of select="rtml:TargetBrightness/rtml:Magnitude"/>
        </V>
    </xsl:template>

    <xsl:template match="rtml:Schedule/rtml:Catalogue/rtml:Target">
        <FieldOfView>
            <xsl:call-template name="Target"/>
        </FieldOfView>
    </xsl:template>

</xsl:stylesheet>
```

Listing 8: XML document (Gl586A.xml) with directives for the STELLA-I telescope. It corresponding to the RTML observation request of Listing 6.

```
<?xml version="1.0"?>
<!DOCTYPE Target SYSTEM "/z/stella/stella/xml/target.dtd">
<Target access="enabled" type="single">
  <TargetName>Gl586A</TargetName>
  <Abstract proposal="gl586perihel">This target is active 3 days before and
      after perihel</Abstract>
  <File>Gl586A.xml</File>
  <User>Strassmeier</User>
  <Email>
    <Address>kstrassmeier@aip.de</Address>
    <Notify>
      <Onblock/>
      <Onfirstpick/>
      <Oncomplete/>
    </Notify>
  </Email>
  <Institution>AIP</Institution>
  <History complete="false" max="500"/>
  <Exception for="this">
    <Delaymax>30</Delaymax>
```

```
    <Delaytime>600000</Delaytime>
    <Delay>NO_STAR_ON_ACQUIRE</Delay>
  </Exception>
  <Select>
    <Requires>Roofopen &amp;amp;amp; InitTelescope</Requires>
    <Constraint relax="1.0" overdue="0">
      <Variable>Date</Variable>
      <From formatclass="java.text.SimpleDateFormat" formatpattern="yyyy-MM-dd'
          T'HH:mm:ss zzz">2007-04-01T00:00:00 UTC</From>
      <To formatclass="java.text.SimpleDateFormat" formatpattern="yyyy-MM-dd'T'
          HH:mm:ss zzz">2007-04-07T00:00:00 UTC</To>
    </Constraint>
    <Constraint relax="1.0" overdue="500000">
      <Variable>SolHeight</Variable>
      <Max>-12.0</Max>
    </Constraint>
    <Constraint relax="1.0" overdue="0">
      <Variable>AltTarget</Variable>
      <Min>30.0</Min>
    </Constraint>
    <Merit>
      <Timeslot class="stella.xml.ConstantMerit">
        <Constant class="java.lang.Double">
          <Constantname>impact</Constantname>
          <Constantvalue>1.0</Constantvalue>
        </Constant>
      </Timeslot>
      <Gain class="stella.xml.ConstantMerit">
        <Constant class="java.lang.Double">
          <Constantname>impact</Constantname>
          <Constantvalue>30.0</Constantvalue>
        </Constant>
      </Gain>
    </Merit>
  </Select>
  <Setup id="set0">
    <Instrument>FilterWheel</Instrument>
    <Constant class="java.lang.String">
      <Constantname>Filter</Constantname>
      <Constantvalue>clear</Constantvalue>
    </Constant>
  </Setup>
  <Setup id="set1">
    <Instrument>SES</Instrument>
    <Constant class="java.lang.Double">
      <Constantname>ExposureTime</Constantname>
      <Constantvalue>600.0</Constantvalue>
    </Constant>
    <Constant class="java.lang.Double">
      <Constantname>ExposureIncrease</Constantname>
      <Constantvalue>1.0</Constantvalue>
    </Constant>
  </Setup>
  <Object id="main">
    <ObjectName rec="0">Gl 586A</ObjectName>
    <Position>
      <RA>232.0400478</RA>
      <Dec>-9.3480694</Dec>
      <Epoch>2000.0</Epoch>
```

```
      <Equinox>2000.0</Equinox>
      <V>6.86</V>
    </Position>
    <FieldOfView>
      <RA>232.04015745</RA>
      <Dec>-9.348610555555556</Dec>
      <Epoch>2000.0</Epoch>
      <Equinox>2000.0</Equinox>
      <V>6.95</V>
    </FieldOfView>
    <FieldOfView>
      <RA>232.050999435</RA>
      <Dec>-9.35839</Dec>
      <Epoch>2000.0</Epoch>
      <Equinox>2000.0</Equinox>
      <V>7.74</V>
    </FieldOfView>
  </Object>
</Target>
```

F: References / Bibliography

References

[1] Breitling, F. Grid Integration of Robotic Telescopes. `http://www.gac-grid.org/project-products/RoboticTelescopes.html`, 2007.

[2] Breitling, F. and Granzer, T. Providing Static Metadata of Robotic Telescopes to Stellaris. Technical Report D2.4, AstroGrid-D project, May 2007.

[3] J. Clark. XSL Transformations (XSLT) Version 1.0. `http://www.w3.org/TR/xslt`, November 1999.

[4] Eclipse - an open development platform. `http://www.eclipse.org/`, 2007.

[5] Ian T. Foster. Globus toolkit version 4: Software for service-oriented systems. In *NPC*, pages 2–13, 2005.

[6] F. V. Hessman. Remote Telescope Markup Language (RTML). *Astronomische Nachrichten*, 327:751–+, September 2006.

[7] F. Manola and E. Miller. RDF primer. `http://www.w3.org/TR/rdf-primer/`, February 2004.

[8] SPARQL Query Language for RDF, 2006.

[9] K. G. Strassmeier, T. Granzer, M. Weber, M. Woche, M. I. Andersen, J. Bartus, S.-M. Bauer, F. Dionies, E. Popow, T. Fechner, G. Hildebrandt, A. Washuettl, A. Ritter, A. Schwope, A. Staude, J. Paschke, P. A. Stolz, M. Serre-Ricart, T. de la Rosa, and R. Arnay. The STELLA robotic observatory. *Astronomische Nachrichten*, 325:527–+, October 2004.