Astronomical Data Analysis Software and Systems IX
ASP Conference Series, Vol. 216, 2000
N. Manset, C. Veillet, and D. Crabtree, eds.

Using XML and Java for Astronomical Instrument Control

T. J. Ames

NASA/Goddard Space Flight Center, Code 588

L. C. Koons, K. B. Sall, C. E. Warsaw

AppNet, Inc.

Abstract. The Astronomical Instrument Markup Language is a domain-specific implementation of the more generalized Instrument Markup Language. NASA Goddard Space Flight Center and AppNet, Inc. are developing AIML to command and control astronomical instruments. Our software architecture combines the platform-independent processing capabilities of Java with the power of XML. A key aspect of the object-oriented architecture, implemented in Java, involves software that is driven by the AIML instrument description. Our techniques, initially targeted for an airborne observatory, will eventually enable astronomers around the world to easily access infrared instruments (e.g., telescopes, cameras, and spectrometers) located in various remote and inhospitable environments.

1. Problem Statement and Background

Traditionally, instrument command and control systems have been highly specialized, consisting mostly of custom code that is difficult to develop, maintain, and extend. Such solutions are initially very costly and are inflexible to subsequent engineering change requests. Instrument description is too tightly coupled with details of implementation.

The *Instrument Remote Control (IRC)* project addresses these problems. NASA Goddard Space Flight Center and AppNet, Inc. are developing a very general and highly extensible framework that applies to virtually any kind of instrument that can be controlled by a computer (e.g., telescopes, microscopes and printers).

Our initial prototyping effort was targeted for HAWC (High-resolution Airborne Wideband Camera), a first-light instrument of SOFIA (the Stratospheric Observatory for Infrared Astronomy). A production-level application of this technology is for one of the three candidate detectors of SPIRE (Spectral and Photometric Imaging REceiver), a focal plane instrument proposed for the European Space Agency's Far Infrared Space Telescope. The detectors are being developed by the Infrared Astrophysics Branch of NASA/GSFC.

2. IRC Project Goals

The Advanced Architectures and Automation Branch (Code 588)[1] of NASA / GSFC has defined the following goals for the Instrument Remote Control project:

- Reduction of life cycle costs of future ground, airborne, and space-based instrumentation projects;
- Provide an easy to use, intuitive GUI, which provides real-time control, either interactive or scripted, and assessment of the experiment;
- Design an extendable and flexible architecture which can support a wide variety of astronomical instruments;
- Platform independence, allowing the assigning of processes to different computers regardless of operating system;
- Makes it easy to develop, modify, and extend control software;
- Explicitly promoting reuse by design by utilizing emerging technologies that foster reuse;
- Greatly reducing implementation time for facility instruments.

3. Solution: Java and XML

A key aspect of the object-oriented architecture, implemented in Java, involves software that is driven by an instrument description. The *Astronomical Instrument Markup Language (AIML)* is a domain-specific implementation of the more generalized *Instrument Markup Language (IML).* The software architecture combines the platform-independent processing capabilities of Java with the vendor-independent data description syntax of Extensible Markup Language (XML), a human-readable and machine-understandable way to describe structured data. IML is used to describe command sets (including parameters, datatypes, and constraints) and their associated formats, telemetry, and communication mechanisms. The software uses this description to present graphical user interfaces to control and monitor the instrument.

Recent efforts have extended to command procedures (scripting) and representation of data pipeline inputs, outputs, and connections. Near future efforts are likely to include an XML description of data visualizations, as well as the potential use of XSL (Extensible Stylesheet Language) to permit astronomers to customize the user interface on several levels: per user, instrument, subsystem, or observatory-wide. We also plan to provide instrument developers with customized XML editors and other tools that facilitate easy instrument description and design iteration.

4. Code Example

The XML excerpt below defines a single ASCII "Move" (telescope) command, which consists of three arguments (RA, DEC, and Epoch). RA and DEC use the same IRC-defined datatype, but have a different range of values. Epoch is an unconstrained floating point value.

---

[1] `http://aaaprod.gsfc.nasa.gov/website/`

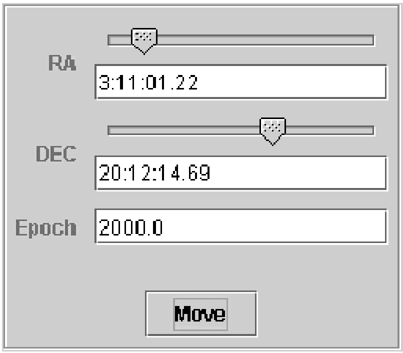

Figure 1. Move Command Interface.

```
<Command name="Move" >
  <Argument name="RA"
            type="gov.nasa.gsfc.irc.datatypes.Sexagesimal" >
    <ValidRange low="00:00:00.0" high="23:59:59.99" />
  </Argument>
  <Argument name="DEC"
            type="gov.nasa.gsfc.irc.datatypes.Sexagesimal" >
    <ValidRange low="-89:59:59.99" high="89:59:59.99" />
  </Argument>
  <Argument name="Epoch" type="Float" />
</Command>
```

In general, each Command has an associated RecordFormat which describes how the user's input will be formatted for transmission to the instrument.

```
<RecordFormat name="Move"  size="-1" ordered="true"
  terminator="&#10;" attributeSeparator=" ">
  <Format name="RA"  format="%s" header="RA=" />
  <Format name="DEC"  format="%s" header="DEC=" />
  <Format name="Epoch" format="%.1f" header="EPOCH=" />
</RecordFormat>
```

The above command and its associated format result in the following command string being sent to the actual instrument.

```
  Move RA="21:05:54.98" DEC="-12:50:07.98" EPOCH=1950.0<CR>
```

The GUI screen that is created by the Java framework from this XML description appears in Figure 1.

Note the sliders for the two constrained parameters, RA and DEC, and the lack of a slider for the unconstrained value, Epoch.

Binary commands, convenience commands, command procedures (using Python and JPython), complex telemetry, DMA connections to an instrument, extremely fast data rates (up to 15 MB/sec), and support for a Programmable

Signal Generator (using an XML version of SCPI standards) have also been implemented, as described on IRC Web site.[2]

5. Benefits

To summarize, the IRC approach to instrument description and control offers a number of benefits over more traditional development paradigms.

- *Extensible Framework.* Instrument software is easier to develop, modify, and extend than traditional custom solutions. The IRC framework is also flexible enough to accommodate a wide variety of instruments. Significant code reuse is anticipated, with savings of up to 70% to 90% over traditional development paradigms. Architecture supports intelligent autonomy technology.
- *Clearly Defined Interface.* Implementation is separate from description since the interface between hardware and software engineers is clearly defined. AIML instrument description supplied by hardware engineers describes what software developers need to know about the instrument.
- *Supports Iterative Development.* Multiple iterations of the instrument description are expected during design and implementation. Changes in the instrument description are automatically reflected in the runtime system.
- *Cross Platform, General Solution.* Software runs on multiple platforms in a distributed environment. Multi-mission reuse is possible by changing XML configuration files.
- *Simplified Commanding and Data Acquisition.* Customizable, intuitive graphical user interface with integrated support for command procedures (scripting). IRC gives immediate access to telemetry by a subscription mechanism that hides data communication details.

References

NASA/Goddard IRC website (prototype, papers, presentations, DTD):
`http://pioneer.gsfc.nasa.gov/public/irc`

NASA/Goddard XML for Astronomy website:
`http://pioneer.gsfc.nasa.gov/public/xml`

Stratospheric Observatory for Infrared Astronomy (SOFIA):
`http://sofia.arc.nasa.gov/`

Sall, K. Ames, T. & Warsaw, C. 1998, in ASP Conf. Ser., Vol. 172, Astronomical Data Analysis Software and Systems VIII, ed. David M. Mehringer, Raymond L. Plante, & Douglas A. Roberts (San Francisco: ASP), 103, NASA's Instrument Control Markup Language

St. Laurent, S. 1999, Inside XML DTDs: Scientific and Technical, 300-309

St. Laurent, S. 1999, XML: A Primer, Second Edition, 294-297

---

[2] `http://pioneer.gsfc.nasa.gov/public/irc`