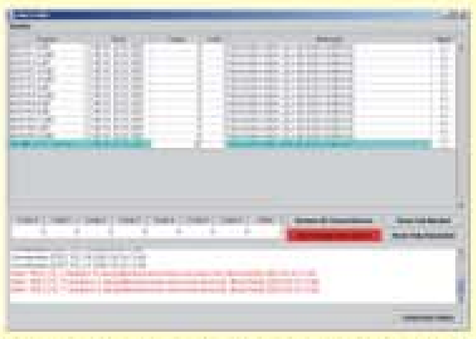
cosylab
CONTROL SYSTEM LABORATORY

Jožef Stefan Institute

Advanced Control System (ACS) Overview and Technical Features

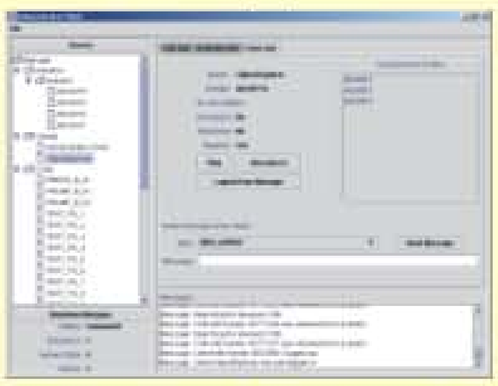
K. Zagar, M. Plesko, M. Sekoranja, I. Verstovsek, D. Vitas (JSI and Cosylab),
G. Chiozzi, B. Jeram, H.Sommer, R. Georgieva (ESO),
D. Fugate (NRAO), R. Cirami (AOT)

Generic Tools



Alarm Table shows alarms that are present in the control system.

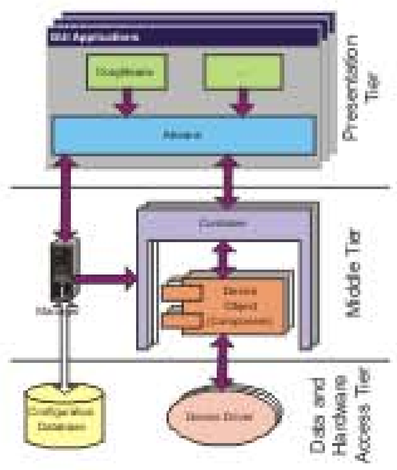
Object Explorer: a generic application that allows access to components and their properties.



Administrator Client shows all containers and components within them.

Log Viewer subscribes with the Notification Service to receive and display logging events.

Management Services



Manager: knows about all components.

Container: provides environment for components (services, lifecycle, ...)

Object Model

Component:
Represents a device (e.g., a power supply).
May contain other components (e.g., water cooling).
Contains properties and operations (device actions).

Property: a value within the device (e.g., the power supply's current).

Characteristic: an attribute of a component or property (description, graphing range, alarm thresholds, ...)

Base classes handle the boring stuff (monitors, asynchronous communication, ...).

Overview

The *Advanced Control System* (ACS) offers an **infrastructure** for development of control systems. It addresses common issues found in control systems, such as:

- Conveying data from input channels to operators (synchronous, monitoring, ...)
- Sending of commands from operators to output channels.
- Dispatching and handling of alarms.

The ACS was designed for use in large experimental physics facilities: particle accelerators, telescopes and the like. Thus, it's **strengths** are:

- Adherence to **standards** and **proven solutions** (CORBA middleware, ...).
- **Easy integration** with process orchestration system (observation scheduling).
- Easy integration of other control systems (e.g., EPICS, TINE, ...).
- **Scalability**: ability to handle large number of control points.
- **Maintainability**: lasting understanding of how the system was build.
- Ease of writing applications (Java, Abeans, Python, ...).

Middleware

Common Object Request Broker Architecture (**CORBA**): Hides the details of network communication.

Naming Service
resolves component names to their references.

Telecom Log Service
used by containers to submit log entries

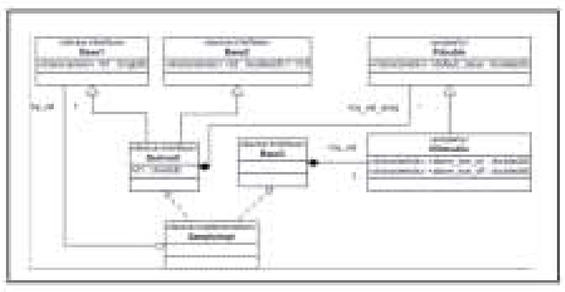
Notification Service
decoupled publisher/subscriber implementation for logs, archive entries and alarms

Configuration Database

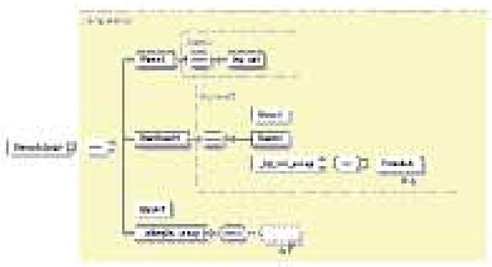
Records are stored in **XML**
·Easily extensible
·Capable to persist complex objects
·Allows validation using XML schema

Data Access Object (DAO): hides the complexities of XML to the components that use configuration data.

Data Access Layer (DAL): allows access to the database via CORBA. Given the name of a record, serves the corresponding DAO object.



An example object model.



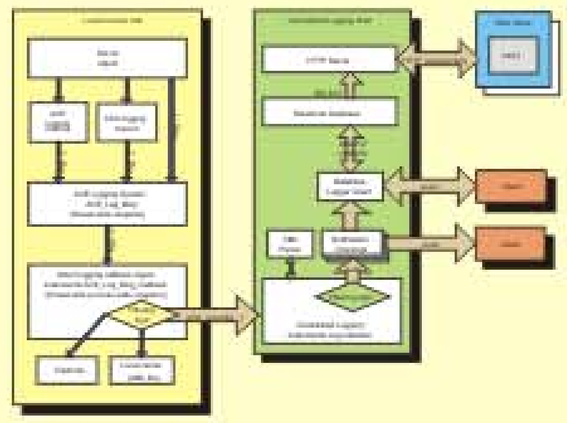
XML schema required for persisting the example object model.

Logging and Archiving

Every container provides logging services to its components.

Centralized Logger
collects log entries submitted by various parts of the system.

Archiving
special log entries, conveying property values.



For information on the past, present and future of the ACS, as well as where it is currently used and how, please see the poster by G. Chiozzi et al, "The ALMA Common Software (ACS): Status and Developments" (MP703).