OLOG AND CONTROL SYSTEM STUDIO: A RICH LOGGING ENVIRONMENT

K. Shroff#, L. Dalesio, A. Arklic, NSLSII, Upton, NY, USA
E. Berryman, FRIB, East Lansing, MI, USA

Abstract

Leveraging the features provided by Olog [1] [2] and Control System Studio [3], we have developed a logging environment which allows for the creation of rich log entries. These entries in addition to text and snapshots images store context which can comprise of information either from the control system (process variables) or other services (directory, ticketing, archiver). The client tools using this context provide the user the ability to launch various applications with their state initialized to match those while the entry was created.

INTRODUCTION

The goal is to provide an environment to facilitate in the creation of log entries along with information which could be used to 1. Better organize the log entries and provide interesting means to query and retrieve the log entries 2. Provide the information required to integrate with various applications and services.

An Olog log entry consists of a create time, an owner, text and attachments, additional log entries also consists of at least one logbook and one or more tags and/or properties. Logbooks and tags provide a mechanism to organize log entries into groups or hierarchies while properties can be used to attach sets of key value data which can be used both for organizational purposes and also the information required for integration with controls and experimental applications.

ARCHITECTURE

The Figure 1. Represents the general architecture of the Olog logging environment which consists of the following major components, the Olog webservice, client libraries in java and python, various client applications including CS-Studio, logbook webclient and various scripts and utilities. The service and the client libraries provide a uniform interface which simplifies and decouples the architecture. The uniform interfaces enable 1. the separation of concerns, thus allowing for the creation of simple and performant clients and service 2. the independent, easy evolution of each of the pieces.

Olog service

The Olog service is a REST style web service [4] [5], which provides the functionality to create, update and query for log entries. The service is stateless i.e no session data is kept for any client, this helps in creating redundant and load balancing systems. In keeping with the RESTful nature, the service exploits existing well-defined technologies to define its interface, the client use HTTP request to identify the resource and define the operation. The data (groups of log entries) in encoded using XML or JSON, which have good support in all current programming and scripting languages. This approach of using standard technologies minimizes the implementation effort on both the application and the service side.

The service allows for the creation of a log entry containing text and attachments, additionally users can use tags, logbooks and properties to include additional information useful to organize the log entries or provide integrations with other systems and services. Olog can be queried for log entries based to creation time, owner, logbooks, tags, properties and the text description. The support for pagination ensures that large requests can be handled in reasonably performant manner.

Client Libraries

While the RESTful Olog web service provides a simple well defined service interface, there also exists client libraries in java and python which provide language specific implementation. These libraries handle the tasks associated with the XML and JSON encoding, managing the HTTP request calls, etc..thus simplifying the client application. The client libraries can also include utility methods that provide the functionality that is common for multiple applications.

Client Applications and Script

The use of Uniform Interfaces and stateless communication has resulted in environment consisting of various client applications which are specialized for different use cases.

Logbook webclient a simple webclient built using bootstrap, jquery and html5; serves as a portable tool to quickly search for a group of entries and also to create and update simple log entries consisting only of text and attachments.

CS-Studio a rich client which in addition to providing means to search and create log entries also provides integration with the applications in CS-Studio. The Olog integration in CS-Studio allows applications to define the data that should be included in a log entry associated with that application, thus when users attempt to create a log entry they are provide with an automatically initialized rich logentry pre populated with properties, tags, text and attachments to capture the context of the CS-Studio applications. The log viewer application in CS-Studio can then consume these rich entries and seamlessly integrate

#shroffk@bnl.gov

ISBN 978-3-95450-139-7

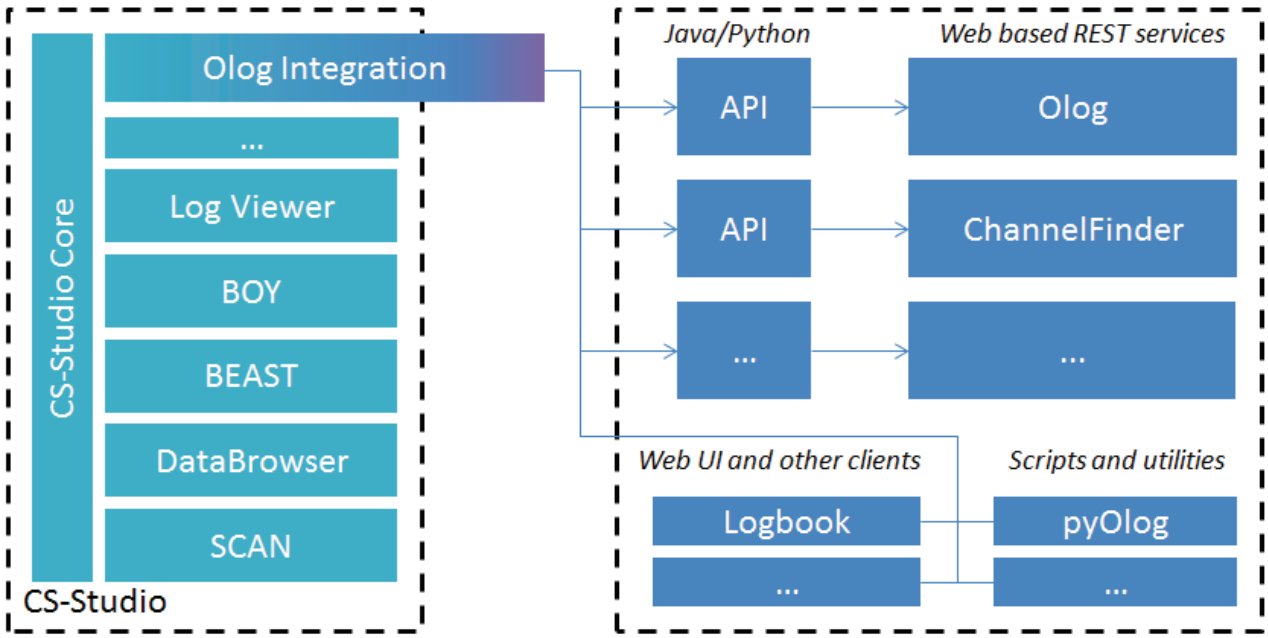

Figure 1: Architecture of the Olog framework showing the various pieces along with their dependency hierarchy.

with the applications and services available using the defined context. For example, while creating a log entry from the BEAST alarm client the log entry can be initialized with a tag associated with the system of the alarming process variable (pv) or process variables, a property which includes the information about the process variables like name, status and severity and an additional property consisting of a configuration file associated with other CS-Studio applications like databrowser. Olog can now be queried for log entries associated with alarms on one or more process variables, using the context information in the rich log entry other CS-Studio applications like the diagnostic probe can be launched for the associated pvs or the databrowser can be launched to display historic archived values for the pvs at the time of the entry creation.

pyOlog scripts the python client library is used for experimental logging. While the raw experimental data is saved in dedicated high performances stores, Olog entries are created, either automatically or manually, along the various steps of an experiment (data acquisition and data processing). For example entries are created at the start of a scan which includes information like sample type/orientation/scan type and the scripts used to run a particular scan, or at the end of a data processing step containing the reduced data set/data plot along with references to the raw data file [6]. These queryable log entries hold references and link providing users an easy way to diagnose a particular scan, review the results of a data processing step and locate the raw data.

CONCLUSION

The Olog architecture provides a simple, scalable, rich logging framework for controls operation and experiment logging purposes. The use of uniform interfaces, stateless session has enabled the decoupling of the various modules and allows for easy independent development of each component. The use of tags, logbooks and properties enables the creation of better organized rich log entries which providing information to better integrate with other controls and data processing applications.

REFERENCES

[1] Olog; http://olog.github.com/
[2] Eric Berryman, Olog, EPICS spring meeting (2013)
[3] Control System Studio; http://controlsystemstudio.github.com
[4] Fielding, Roy Thomas (2000), Architectural Styles and the Design of Network-based Software Architectures, Doctoral dissertation, University of California, Irvine
[5] Pautasso, Cesare; Zimmermann, Olaf; Leymann, Frank (2008-04), "RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision", 17th International World Wide Web Conference (WWW2008) (Beijing, China)
[6] NSLS-II High Level Application Infrastructure and Client API Design, G. Shen, L. Yang, K. Shroff Presented at the 2011 Particle Accelerator Conference (PAC'11)

ISBN 978-3-95450-139-7