Reflective Servers: Seamless Offloading of Resource Intensive Data Delivery

S. Clark, T. D'Ottavio, M. Harvey, J.P. Jamilkowski, J. Morris, S. Nemesure
Collider-Accelerator Department, Brookhaven National Laboratory, Upton, New York, U.S.A.

Background

Over 77,000 Accelerator Device Objects (ADOs) provide the software interface to settings and measurements for accelerator equipment [1]. A majority of the ADOs are hosted on Front End Computers (FECs) with limited memory and CPU resources.

ADOs have four primary data operations:

- Synchronous gets - blocking while retrieving data
- Synchronous sets - updating a set point
- Metadata fetches - retrieving static properties about a readback or set point
- Asynchronous gets - receiving live streams of data updates

Asynchronous gets require maintenance of information which can consume substantial memory when numerous clients are connected. Many applications establish these by default, and resource overconsumption causes FECs to crash, interrupting users and operations.

A prior Reflective ADO project, developed ~10 years ago, faced reliability and maintainability issues inherent to the system and fell out of use [2].

Requirements

Develop a system which removes asynchronous data delivery load from ADOs while also…

- Handling connection interruptions gracefully
- Providing easy configuration and deployment
- Fitting seamlessly into the Controls ecosystem

Architecture

Reflective Server (RS): This is responsible for interfacing directly with client applications and providing facilities to communicate with ADOs efficiently.

- Logically separate frontend (application) & backend (device) communication interfaces
- Binding layer for connection management, request translation, reference counting, data caching

Asynchronous data updates are multiplexed through 1 connection to an FEC.
Data updates received by Reflective Server are fanned-out to many client applications through the frontend interface. Thread concurrency prevents delays if unresponsive or failed clients exist.

Synchronous get requests may optionally retrieve data through the binding layer cache. If data is available, an unnecessary call to the device is prevented.

All other requests are proxied directly through to the device.

Reflective-Aware Central Name Service (rCNS): Proxy to the Central Name Service, allows applications to lookup RPC program information for ADOs using common identifiers.

- Tracks each RS instance and ADOs it reflects.
- Intercepts requests for reflected ADOs, providing information pointing clients to the responsible RS.
- Clients able to bypass reflection by simply CNS instead of rCNS. This is necessary for for real-time processes requiring low-latency.

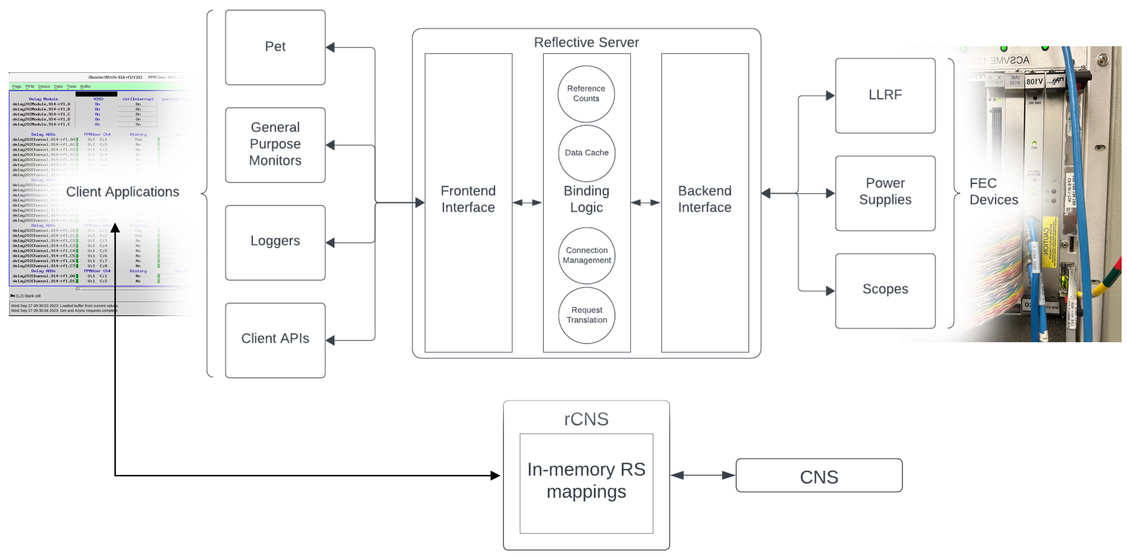

Figure 1: Reflective Architecture Design

Future Work

- Broader rollout of across the complex to test the system's scalability
- Robust, centralized configuration system to simplify RS management
- Implement EPICS frontend & backend interfaces for interoperability with EIC components
- Identify ADOs using “getters” incompatible with the RS get cache and developing a seamless solution.

Performance

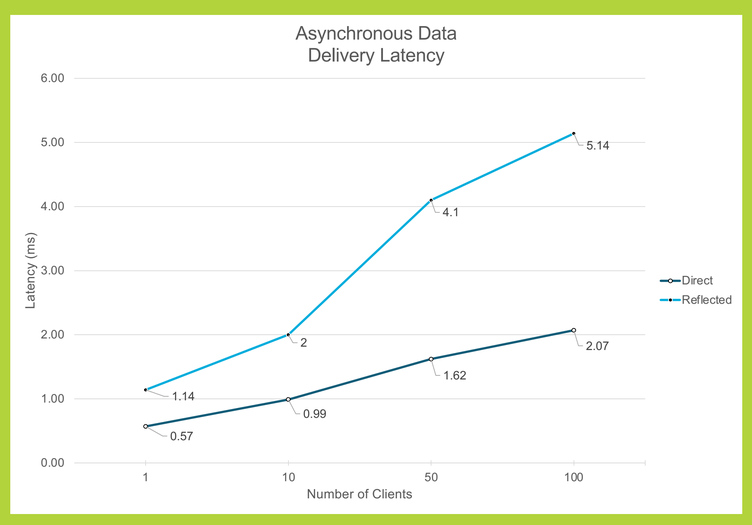

Figure 2: Latency measurements of Asynchronous Data Deliveries vs. Number of Clients connected

Ability to handle much higher client load than FECs directly - 100’s of clients concurrently

Analysis shows only 2x latency increase using Reflection versus communicating directly with FECS

Latency measures include…

- Network round-trip time
- Data marshalling in Reflective Server
- Request/Response processing
- Data fan-out in binding logic

Type	# of Calls	Time / call (μsec)
Direct	1,000	334
Reflected	1,000	317

Figure 3, above: Round-trip time for Synchronous Get Calls, utilizing the binding-layer cache

Figure 4, below: Round-trip time for Synchronous Get Calls, directly proxying to FEC via Reflective Server

Type	# of Calls	Time / call (μsec)
Direct	1,000	334
Reflected	500	594

References

1. D. Barton et al., “Rhic control system,” Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, vol. 499, no. 2, pp. 356–371, 2003, The Relativistic Heavy Ion Collider Project: RHIC and its Detectors.
2. B. Frak, “Applications of transparent proxy servers in control systems,” in ICALEPCS, 2013.

U.S. DEPARTMENT OF ENERGY
www.bnl.gov
Brookhaven National Laboratory