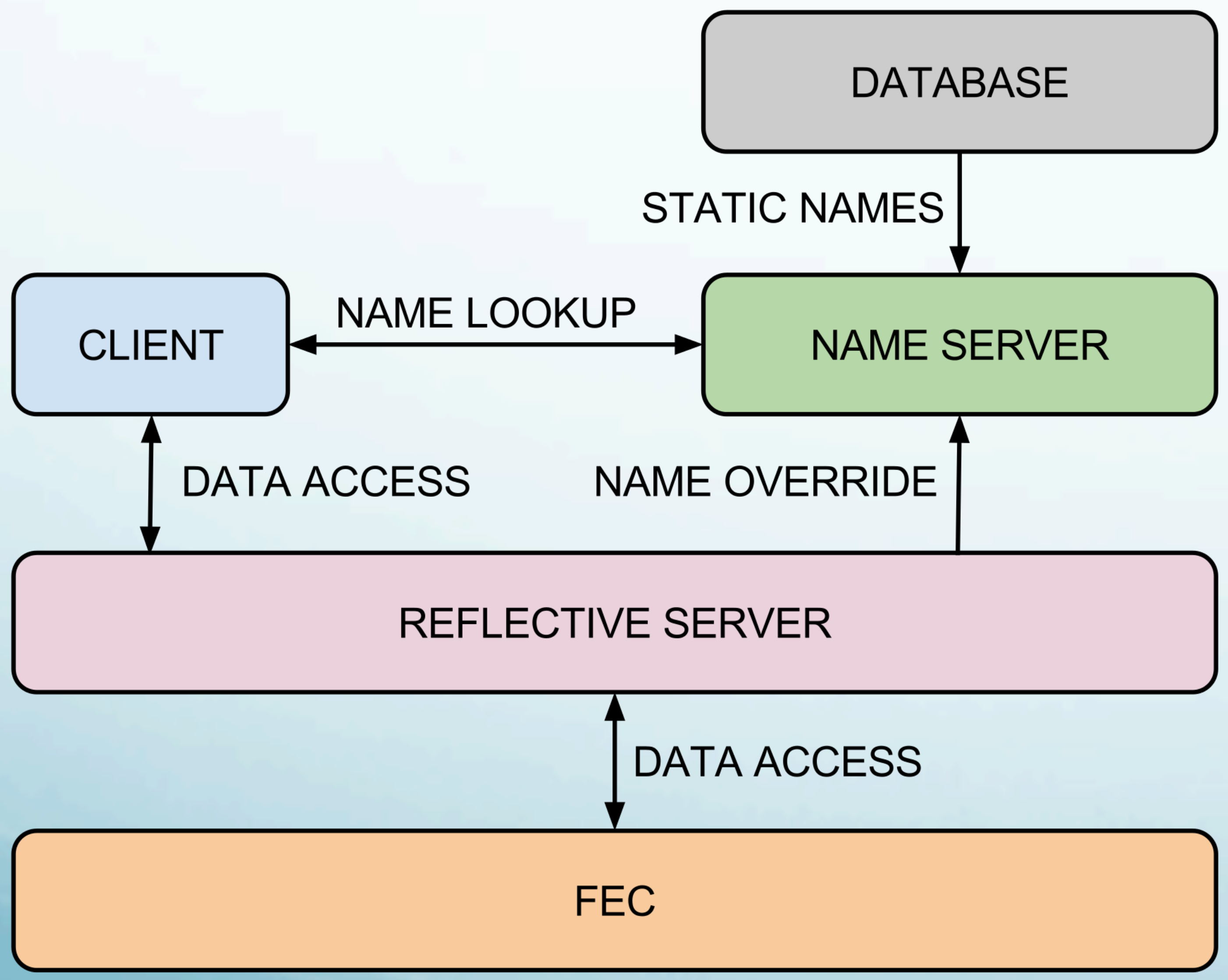
APPLICATIONS OF TRANSPARENT PROXY SERVERS IN CONTROL SYSTEMS*

B. Frak, T. D'Ottavio, M. Harvey, J. Jamilkowski, J. Morris, S. Nemesure, BNL, Upton, U.S.A.

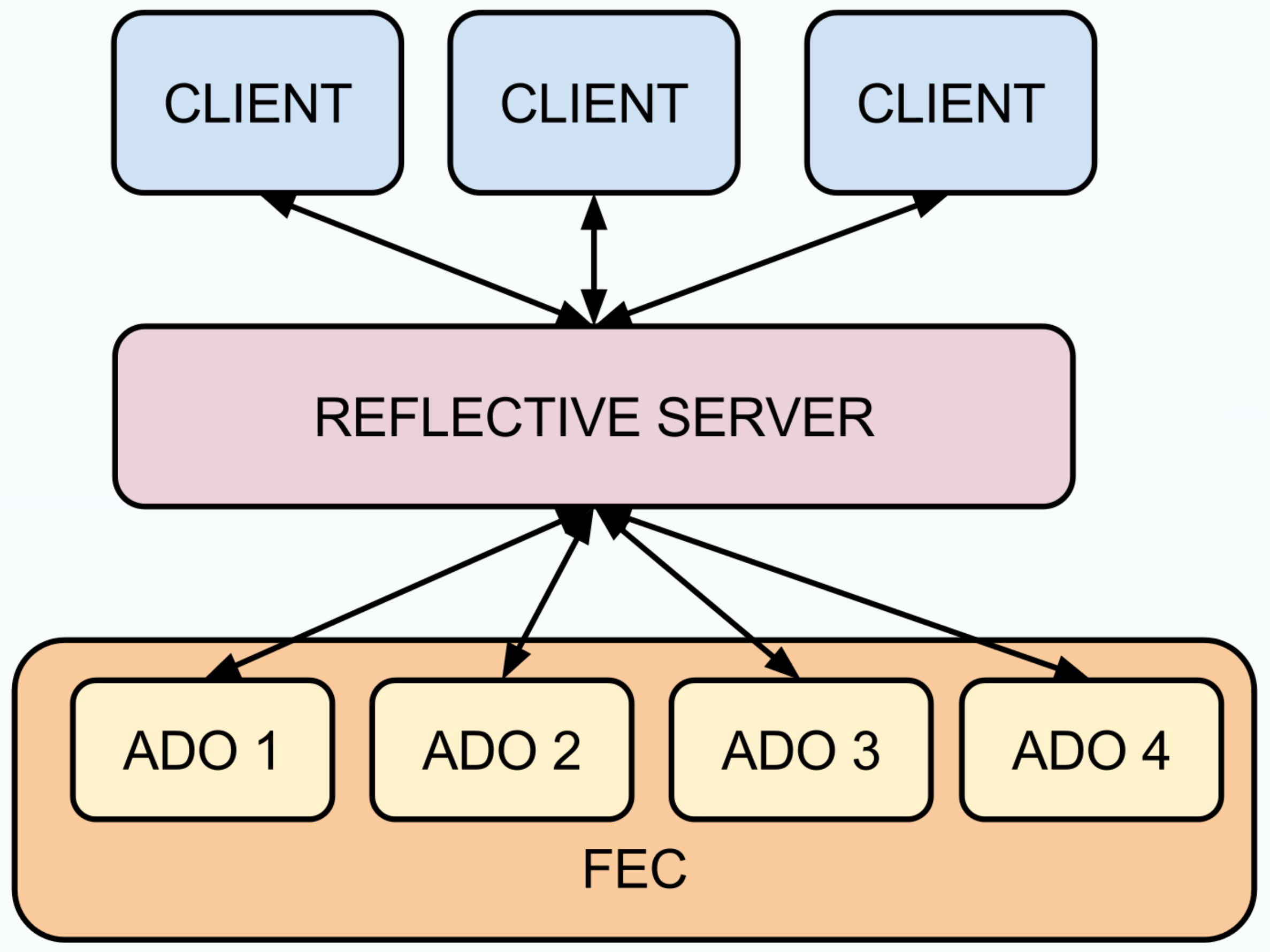
Transparency

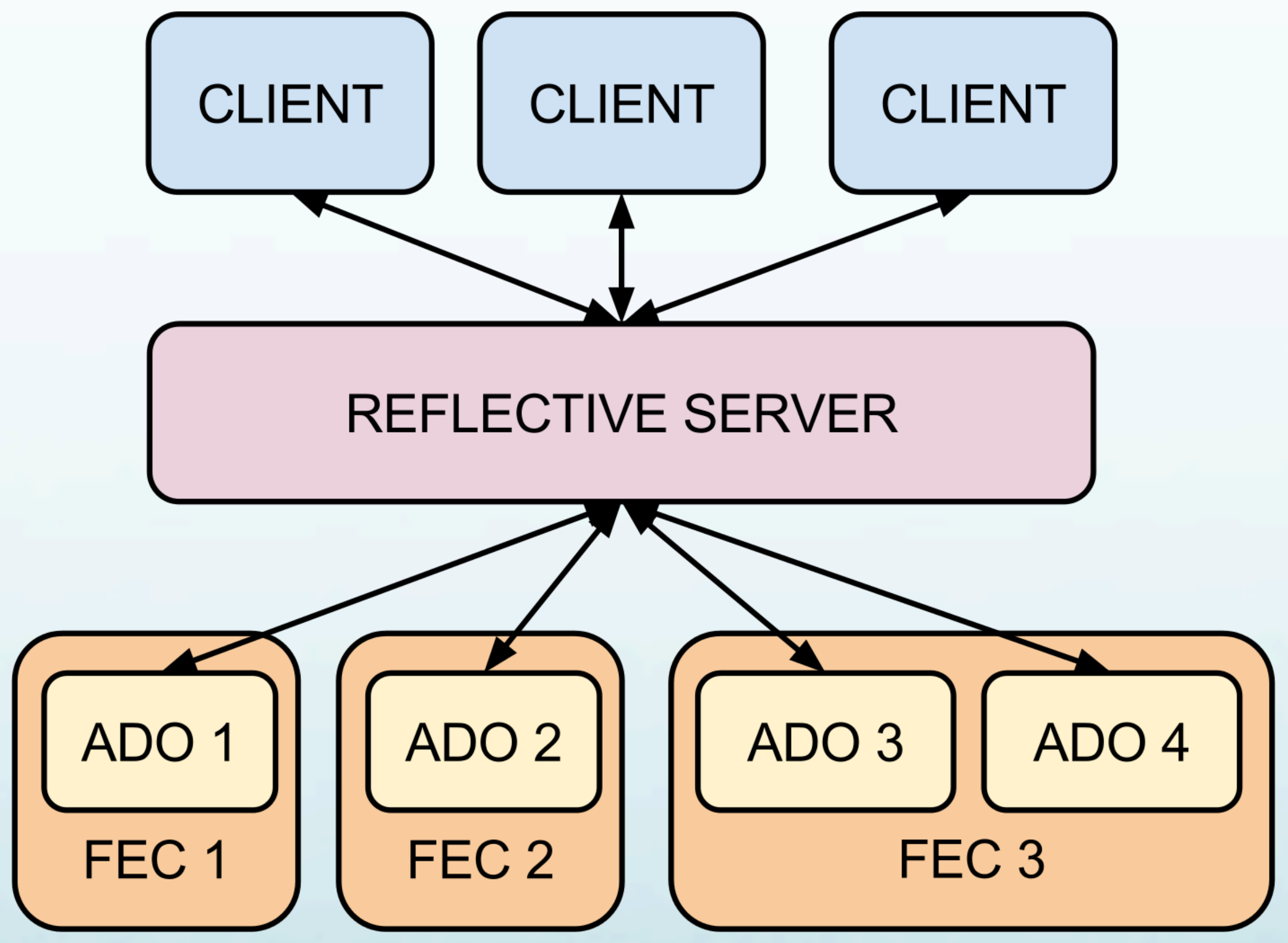
Transparency is achieved by modifying the name server records on as needed basis. The Reflective Server when bootstrapping device objects modifies their entries in the master Controls Name Server to reflect their new "location". This record includes a host name as well as RPC program and version numbers required for all client server communication. These records remain unchanged for the duration of proxy server lifetime and each Reflective Server instance has a responsibility to restore original entries upon shutdown. Clients always obtain the location of device objects from the name server, which means modifying this central repository is the only way to transparently inject proxy instances to the live system.



Runtime Configuration

- One Reflective Server per FEC (the most common scenario)
- One Reflective Server for multiple scattered ADO instances (used to target individual, high volume applications)





Asynchronous Access

This is the key area where Reflective Server framework proves to be the most valuable. By positioning itself in front of backend infrastructure, it essentially removes all subscribe-publish related scaling issues. This mechanism relies on proxy instances becoming exclusive clients to FEC server instances, and thus taking the burden of handling all, client issued, asynchronous requests onto itself.

Extension Points

Extension is an advice, which cuts across all sets and gets (input and outputs) for all Reflective Server contained device objects. This advice is supplied to the RS runtime as a class file, which gets weaved with the existing set of advices already attached to the ADO set and get methods. The most basic extension point overrides two methods from the base aspect – one for input and one for output modification. The former has full control over the data sent to the slave ADO, while the latter controls the data shipped back to the clients. This tight pairing can be utilized by device object developers in a variety of ways during all stages of a development cycle as well as in deployed systems.

CONTROLS Collider - Accelerator Department

BROOKHAVEN NATIONAL LABORATORY

*Work supported by Brookhaven Science Associates, LLC under contract no. DE-AC02-98CH10886 with the U.S.