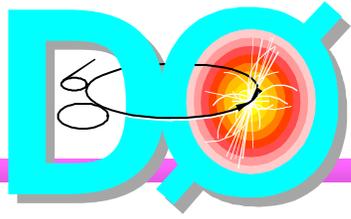


# EPICS Channel Access

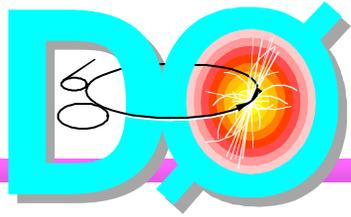
Geoff Savage



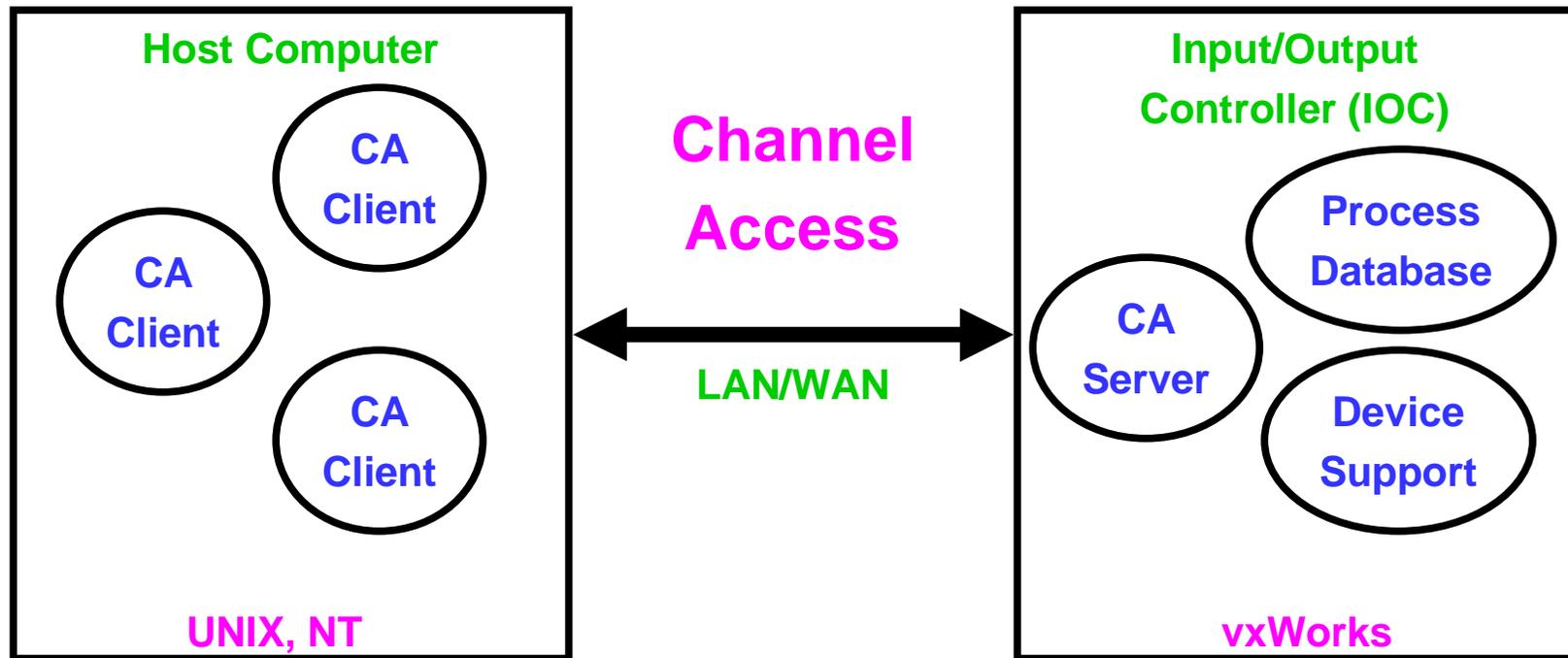
# Outline

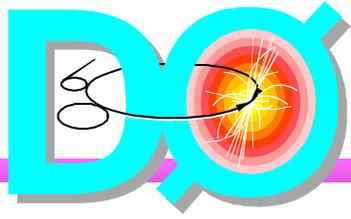
---

- **EPICS channel access (CA)**
  - ◆ **Communication**
  - ◆ **Libraries**
  - ◆ **Services**
  - ◆ **Calls**
  - ◆ **EPICS events**
  - ◆ **Data type conversion**



# EPICS Communication





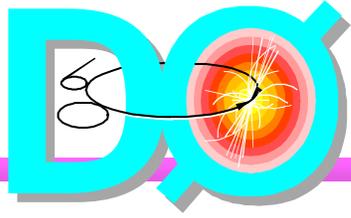
# CA Libraries

- **Client**

- ◆ Connections to channel access servers
- ◆ Read and write across connection
- ◆ IOC must be on the network

- **Server**

- ◆ Connections to information in IOC
- ◆ Read and write access
- ◆ Channel access client calls
- ◆ Client must be on the network



# CA Structure

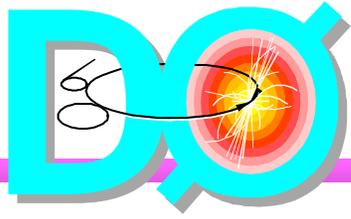
---

**CA Client Library**

**CA Server Library**

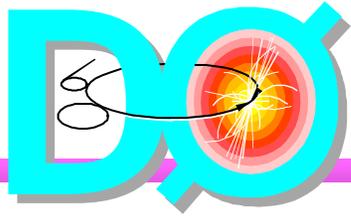
**EPICS Process Database**

**Device Support**

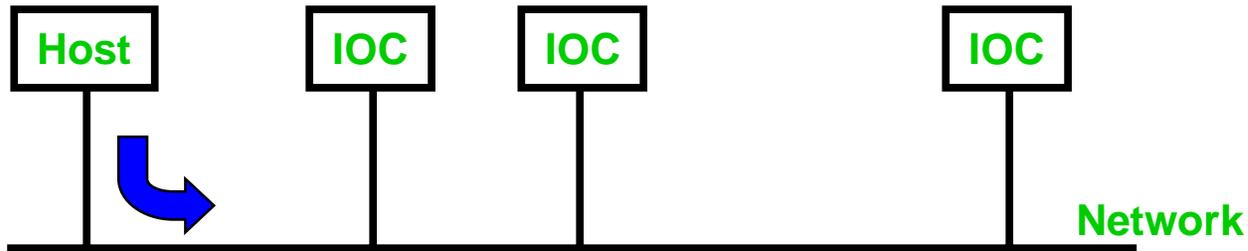


# CA Services

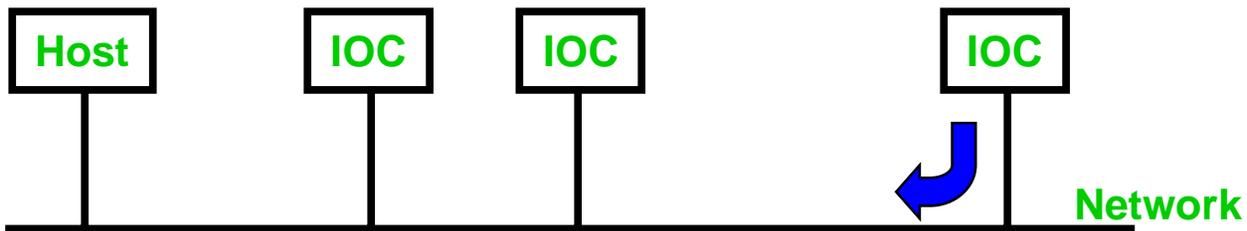
- **Dynamic channel location**
- **Get (read)**
- **Put (write)**
- **Connection monitoring**
- **Automatic reconnect**
- **Conversion to client types**



# Dynamic Connection

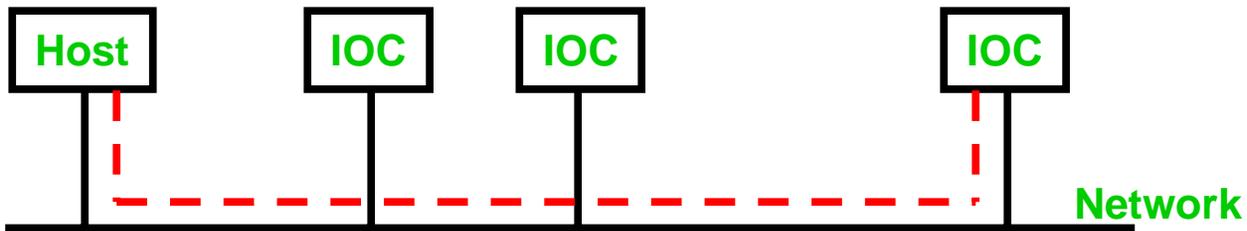


Who has channel 'RM1/AD00'?

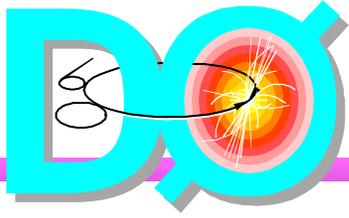


- ◆ Each name unique
- ◆ UDP broadcasts
- ◆ TCP connections

I have channel 'RM1/AD00'.



Create connection.



# Simple CA Client

```
#include <cadef.h>
main( int argc, char **argv) {
    dbr_double_t data;
    chid mychid;

    ca_task_initialize();

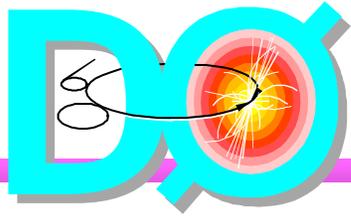
    ca_search(argv[1], &mychid);
    ca_pend_io(5.0);

    ca_get(DBR_DOUBLE, mychid (void *)data);
    ca_pend_io(5.0);
}
```

CA initialization

Synchronous find and  
connect to the channel.

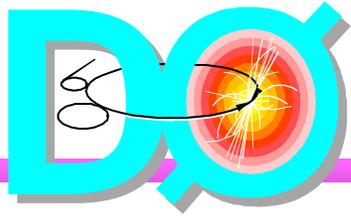
Synchronous get  
of the value.



# Executing CA Actions

---

- Requests are buffered in host
- Buffer sent when full or when user requests
- `ca_pend_io(timeout)`
  - ◆ flush the send buffer then wait for outstanding queries to complete or the specified time expires



# CA Calls

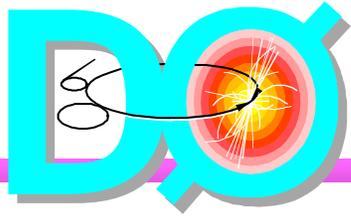
---

- **Connection**

- ◆ `ca_search()` - connect
- ◆ `ca_clear_channel()` - disconnect
- ◆ `ca_change_connection_event()` - sets connect/disconnect callback

- **Synchronous**

- ◆ `ca_get()` - read
- ◆ `ca_put()` - write

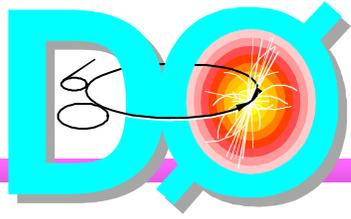


## CA Calls ...

---

- **Asynchronous**

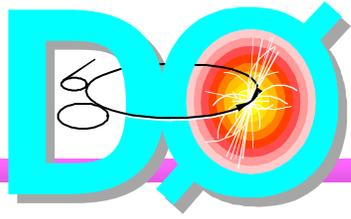
- ◆ **ca\_get\_callback()** - run callback when value is returned
- ◆ **ca\_put\_callback()** - run callback when value is sent
- ◆ **ca\_add\_event()** - run callback when event occurs
- ◆ **ca\_clear\_event()** - remove event callback



## CA Calls ...

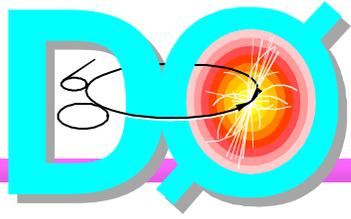
---

- **Request processing**
  - ◆ **ca\_pend\_io()** - flush send buffer and wait for requests to complete
  - ◆ **ca\_pend\_event()** - flush send buffer and wait for asynchronous events
  - ◆ **ca\_flush\_io()** - flush send buffer
  - ◆ **ca\_poll()** - perform outstanding CA background activity



## CA Calls ...

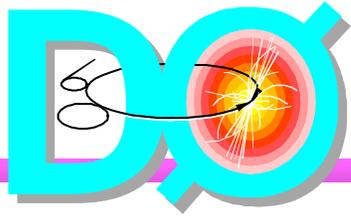
- **Synchronous group**
  - ◆ **ca\_sg\_create()** - create group. Guarantees that a set of CA request have completed.
  - ◆ **ca\_sg\_delete()**
  - ◆ **ca\_sg\_block()** - flush send buffer and wait for group completion
  - ◆ **ca\_sg\_get()**
  - ◆ **ca\_sg\_put()**



# CA Call Categories

---

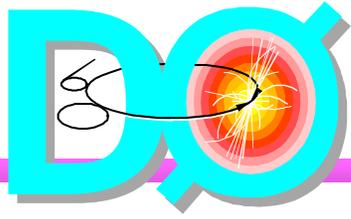
- Connection
- Synchronous
- Asynchronous
  - ◆ speed
- Request processing
- Synchronous group
  - ◆ group completion



# Analog I/O Events

---

- **Event occurs for an error condition**
  - ◆ **SCAN** - problem scanning record
  - ◆ **READ** - problem reading value
  - ◆ **Limit alarms** - new value exceeds alarm limits
- **Trigger alarm monitor**
- **Alarm monitor processes event**



# Data Type Conversion

---

- **EPICS database request types**
  - ◆ **DBR\_STRING, \_DOUBLE, \_FLOAT, \_LONG, \_CHAR, \_ENUM**
- **EPICS database native types**
- **Server performs data type conversions**
- **Client performs endian conversions**
- **Polite clients request data in the native type and converts in client**