The New ***orb2orb*** Program

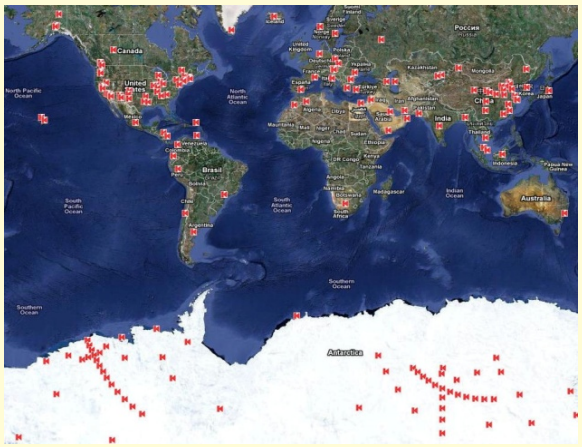
Kent Lindquist
Boulder Real Time Technologies

May 2017

Introduction - KMI

Kinemetrics, Inc.

- Founded in 1969
- OYO Corp owned in 1991
- ISO9001 since 1999
- $35M FY2012 revenue (mostly international)



HQ's in Pasadena CA with Sales and Project offices in Switzerland & Abu Dhabi

Introduction – KMI Team

KINEMETRICS

Designs and manufactures sensors and digitizers – Provides complete systems design, installation and operations

QUANTERRA®

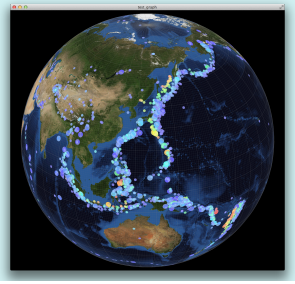
Designs High-End Digitizers

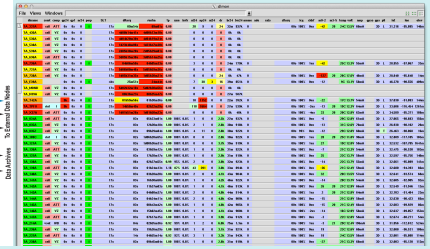
metrozet

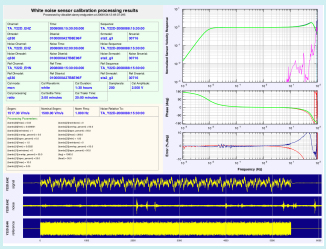
Designs High-End Sensors

BRTT

Antelope Software







Kinemetrics / BRTT
Comprehensive Hardware, Software, and Services

Kinemetrics Systems Solutions

- Turnkey complete systems including enterprise-class computing centers and full communications

Kinemetrics Hardware Manufacturer

- World class Kinemetrics and Quanterra dataloggers
- World class Kinemetrics, Metrozet and Streckeisen sensors

BRTT Software Developer

- World class acquisition software for all Kinemetrics hardware products
- Proven track record for large networks with difficult remote deployments (USArray)
- World class, com
- hensive automated and interactive seismic processing software
- Data neutral architecture for support of non-seismic environmental monitoring networks
- Extraordinary Command & Control capabilities with SOH displaying

Kinemetrics Services

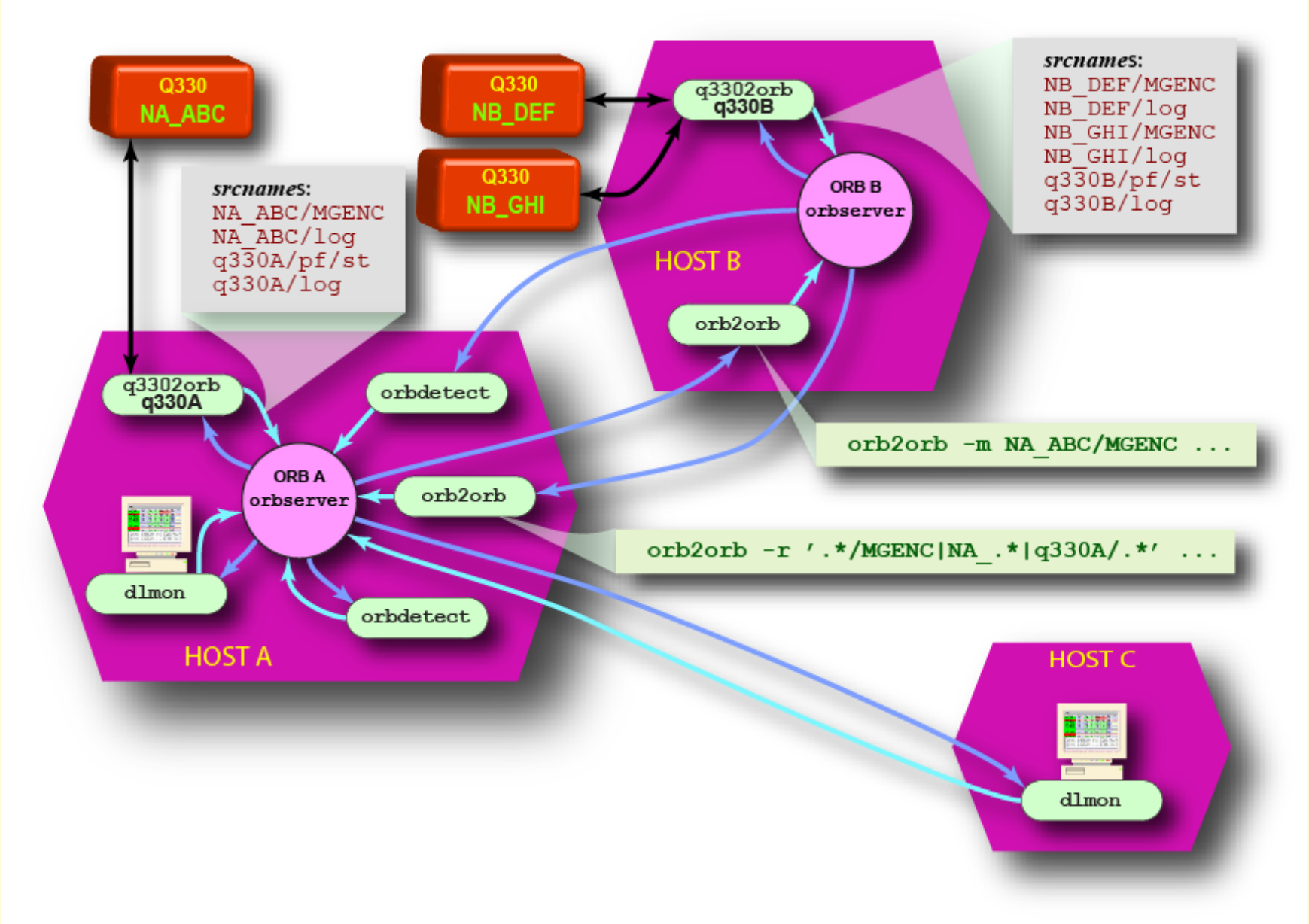
- Complete systems procurement, installation and training including all aspects of both hardware and software
- Network operations

Outline:

- Original design of *orb2orb*
- Current usage
- Design goals
- Current status
- Detailed architecture explanation
- Command line
- Parameter-file structure
- Switching advice
- Future developments

Inner Workings: Pushes, pulls & state info

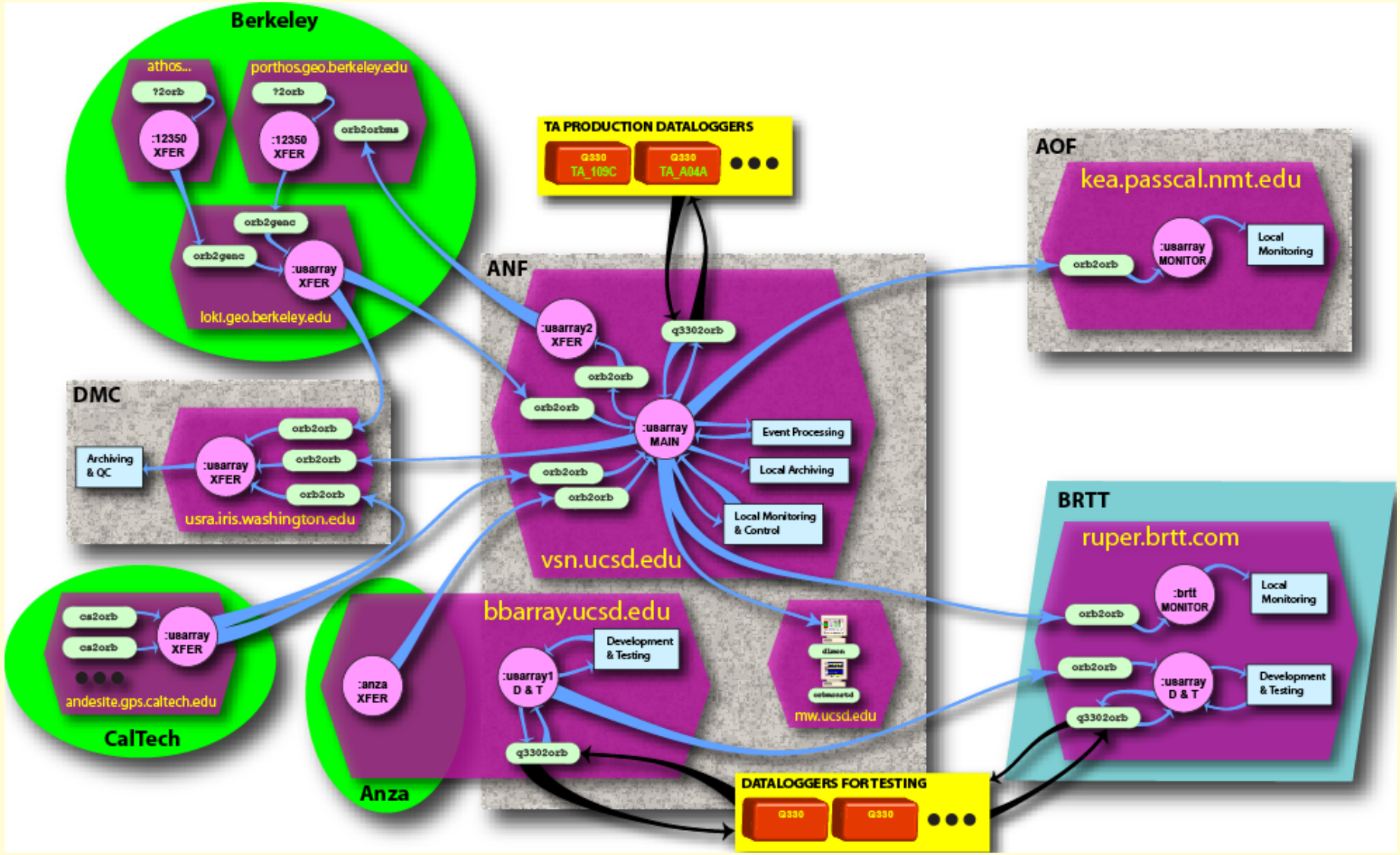
- ARTS has been designed to facilitate automatic transfers of real-time continuous data from one ORB to another: **`orb2orb`**
- Where to run ORB packet transfer clients, like **`orb2orb`**? At one ORB, at the other ORB, anywhere else with an IP connection
- Answer - usually, on the same host as the output ORB so that the pull is going across the long-haul link


Q330
NA_ABC
Q330
NB_DEF
Q330
NB_GHI
srcnames:
NA_ABC/MGENC
NA_ABC/log
q330A/pf/st
q330A/log
q3302orb
q330B
ORB B
orbserver
HOST B
srcnames:
NB_DEF/MGENC
NB_DEF/log
NB_GHI/MGENC
NB_GHI/log
q330B/pf/st
q330B/log
orb2orb
orb2orb -m NA_ABC/MGENC ...
q3302orb
q330A
orbdetect
ORB A
orbserver
orb2orb
orb2orb -r '.*/MGENC|NA_.*|q330A/.*' ...
dlmon
orbdetect
HOST A
HOST C
dlmon

Inner Workings: Pushes, pulls & state info

- Note that most of the inter-host data transfers are done with ORB client pulls
- Note the simplex ORB links
- Independency of ORB-client links; use of threading
- Note the potential feedback data loop between **`orb2orb`** instances on hosts A and B
- Client state processing with Antelope state files


DATALOGGER Q4120
DATALOGGER Q330
DATALOGGER Q330
DATALOGGER K2
qt2orb
q3302orb
k22orb
Datalogger Import Clients
From External Data Nodes
orb2orb
cs2orb
ew2orb
Data Node Import Clients
Object Ring Buffer (ORB)
orbserver
orb2orb
orb2cd1s
orb2ew
Data Node Export Clients
To External Data Nodes
orbdetect
orbassoc
orbmag
Data Processing Clients
orb2db
orb2dbt
Data Archiving Clients
Data Archives
ORB Client Software Modules
ARTS ORB - Client TCP/IP communication
Other communication


Berkeley
athos...
porthos.geo.berkeley.edu
:12350 XFER
orb2orbms
orb2genc
:usarray XFER
loki.geo.berkeley.edu
TA PRODUCTION DATALOGGERS
Q330 TA_109C
Q330 TA_A04A
AOF
kea.passcal.nmt.edu
orb2orb
:usarray MONITOR
Local Monitoring
ANF
:usarray2 XFER
q3302orb
:usarray MAIN
Event Processing
Local Archiving
Local Monitoring & Control
vsn.ucsd.edu
DMC
Archiving & QC
usra.iris.washington.edu
CalTech
ca2orb
andesite.gps.caltech.edu
Anza
:anza XFER
bbarray.ucsd.edu
:usarray1 D & T
Development & Testing
mw.ucsd.edu
BRTT
ruper.brtt.com
:brtt MONITOR
:usarray D & T
DATALOGGERS FOR TESTING
Q330

orb2orb: Current usage

- Many network-to-network dataflow links
  - up to tens of connections to neighboring networks
- Installations with many orb-protocol connections to smart dataloggers
  - up to 100's of individual *orb2orb* connections
- Integral part of network data-acquisition

- These multiple *orb2orb* connections become challenging to configure, maintain, and monitor

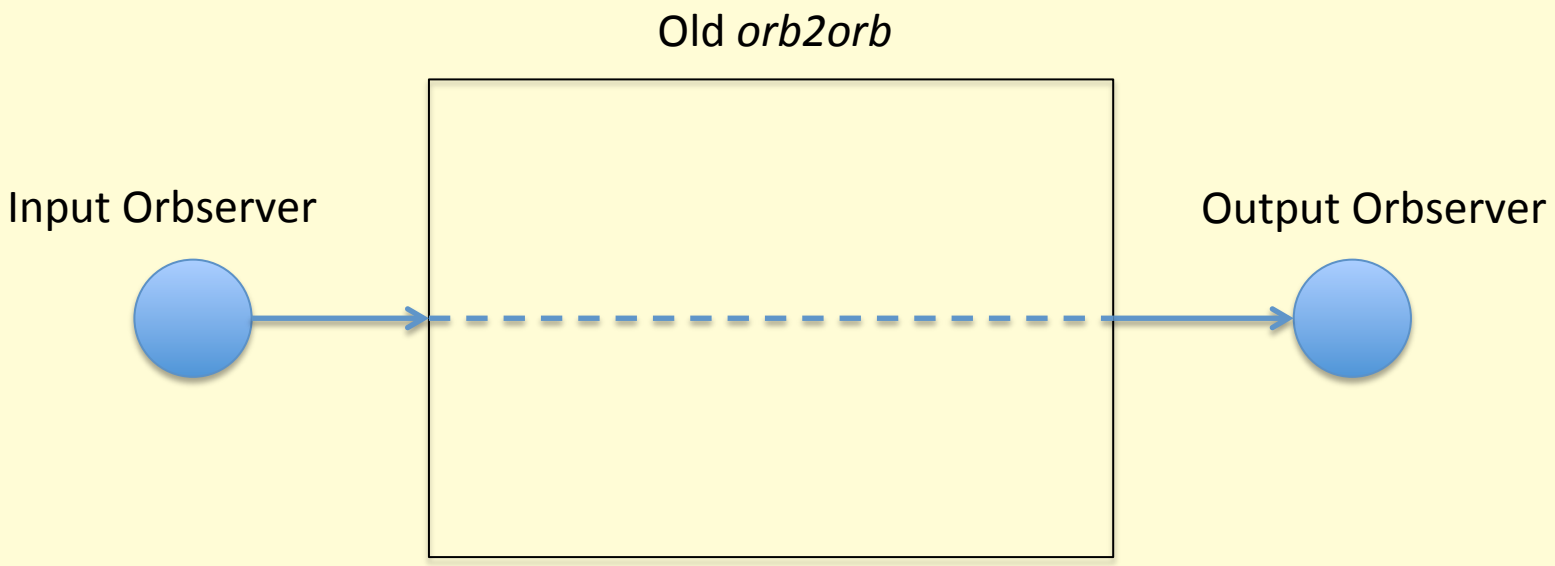
orb2orb: new version

- Design goals
  - Provide datalogger acquisition functionality like *q3302orb* and *altus2orb*
    1. Data ingestion and delivery
       - including repackaging / renaming
       - Point-Of-Contact (POC) call-in capability for dataloggers on dynamic IPs
       - Ultimately: failover support
    2. State-of-Health (SOH) monitoring
       - *dlmon* capabilities
    3. Command-and-control
       - *dlcmd* capabilities
  - Multithreading:
    - multiple *orb2orb* connections with one instance
    - connectivity from *M* source orbs to *N* destination orbs
  - Consolidate slew of related programs (*orb2orb*, *orbxchange*, *orbxthreads*, *orbclone*, etc.)
  - Preserve backwards-compatibility with old *orb2orb*

orb2orb: new version status

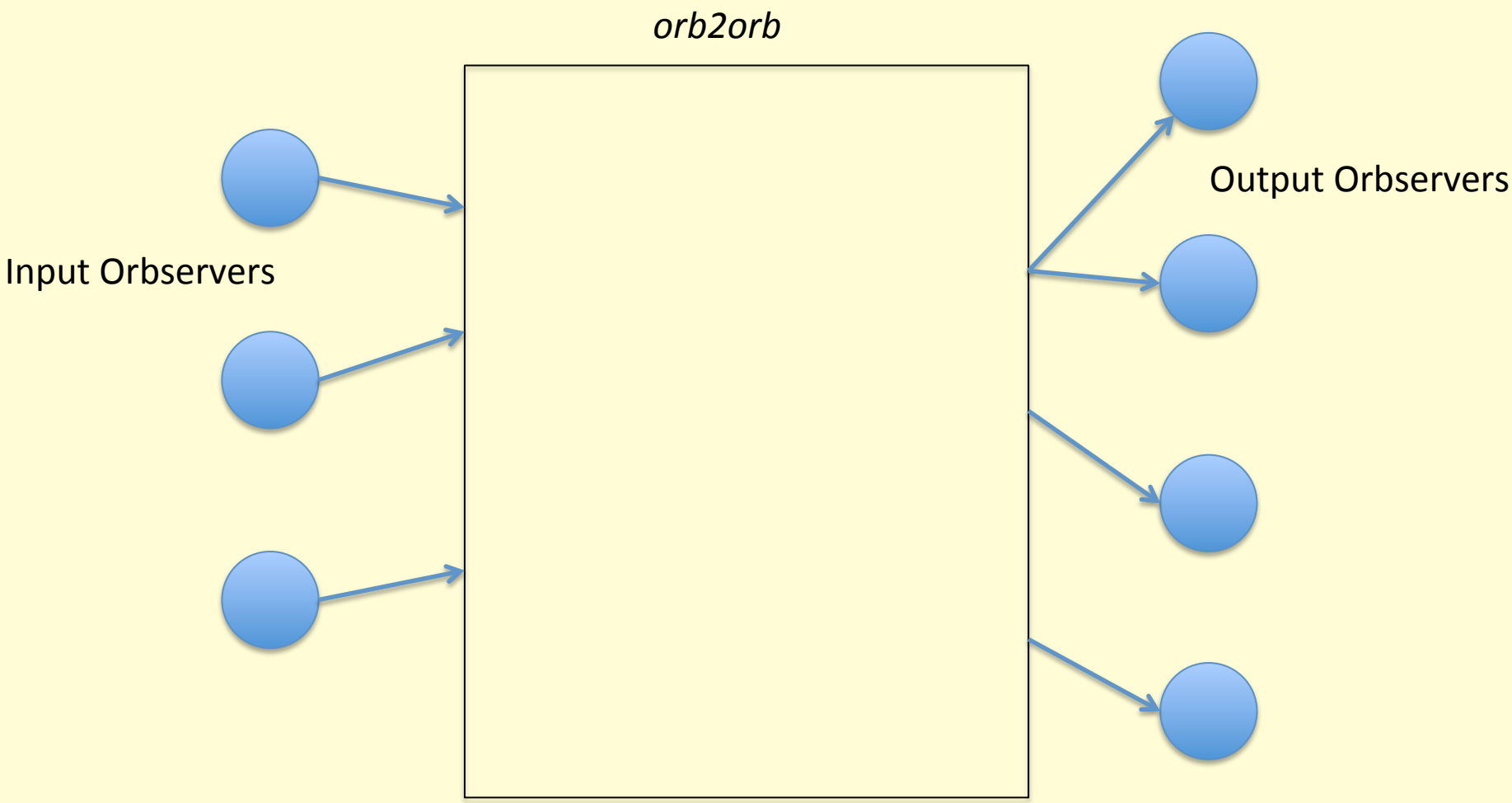
- Data acquisition capabilities
- many-to-many connections in one instance
  - Fully Multithreaded
- *dlmon*-compatible SOH output
- Backwards compatibility with
  - Legacy command-line format
  - Legacy parameter-file format
  - [N.B. Not all parameters/options supported]
- Embedded in GSN *rtdemo*(1)
- New *libooorb* (see ***ooorb***(3)) object-oriented orbserver interaction library (C++)

orb2orb: old architecture

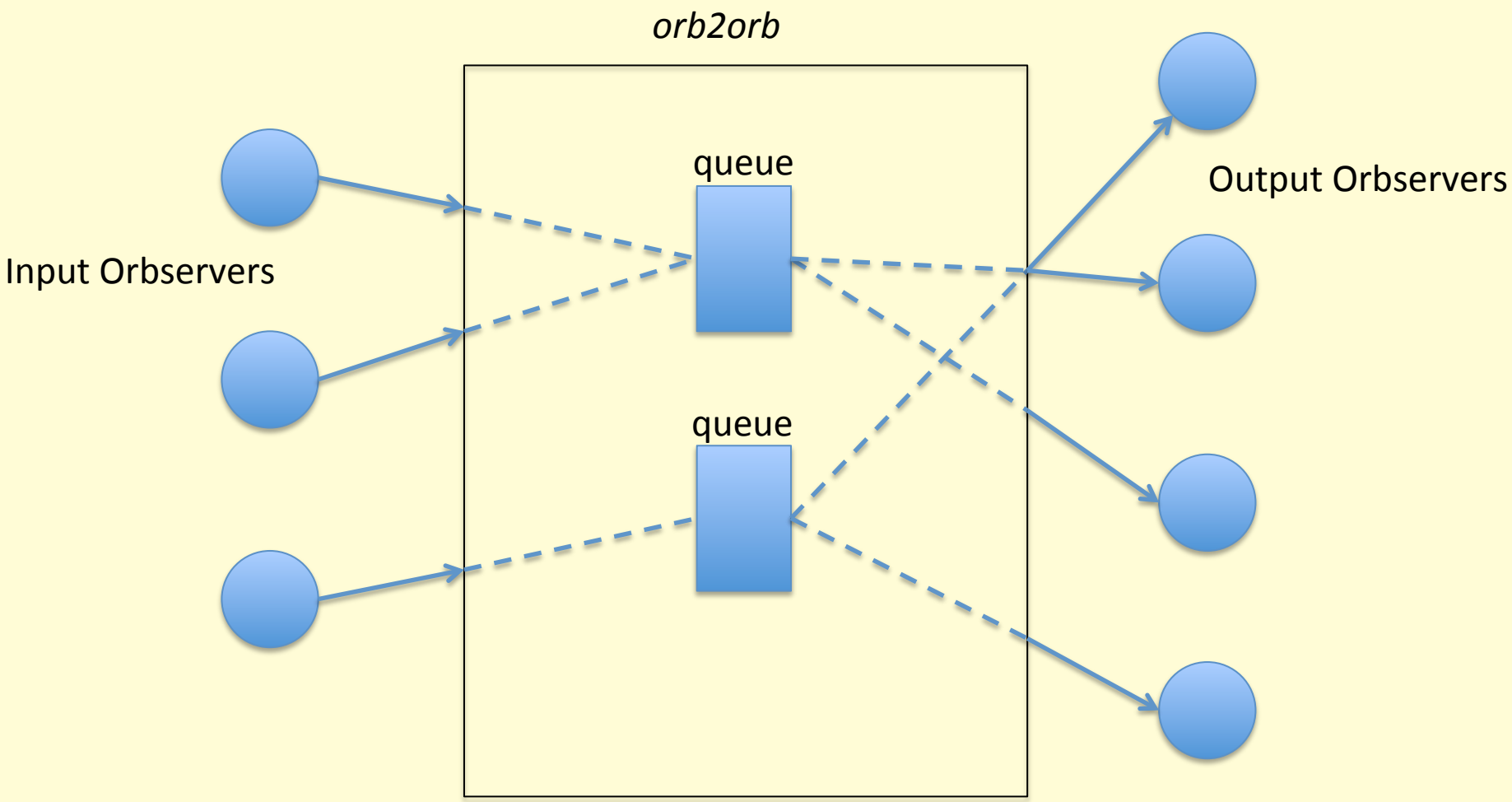


- Served well for many years
- Large networks might have hundreds of individual instances
- Manual configuration becomes burdensome
- Insufficiently supportive of direct data-acquisition role from dataloggers

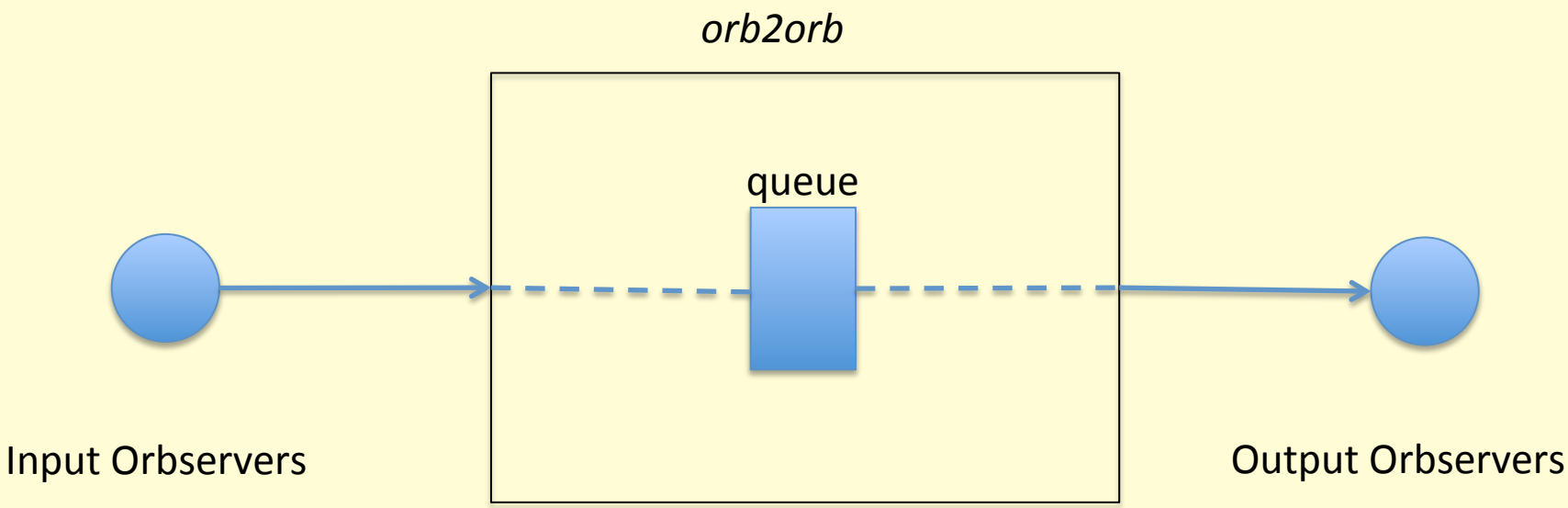
orb2orb: new architecture



orb2orb: new architecture

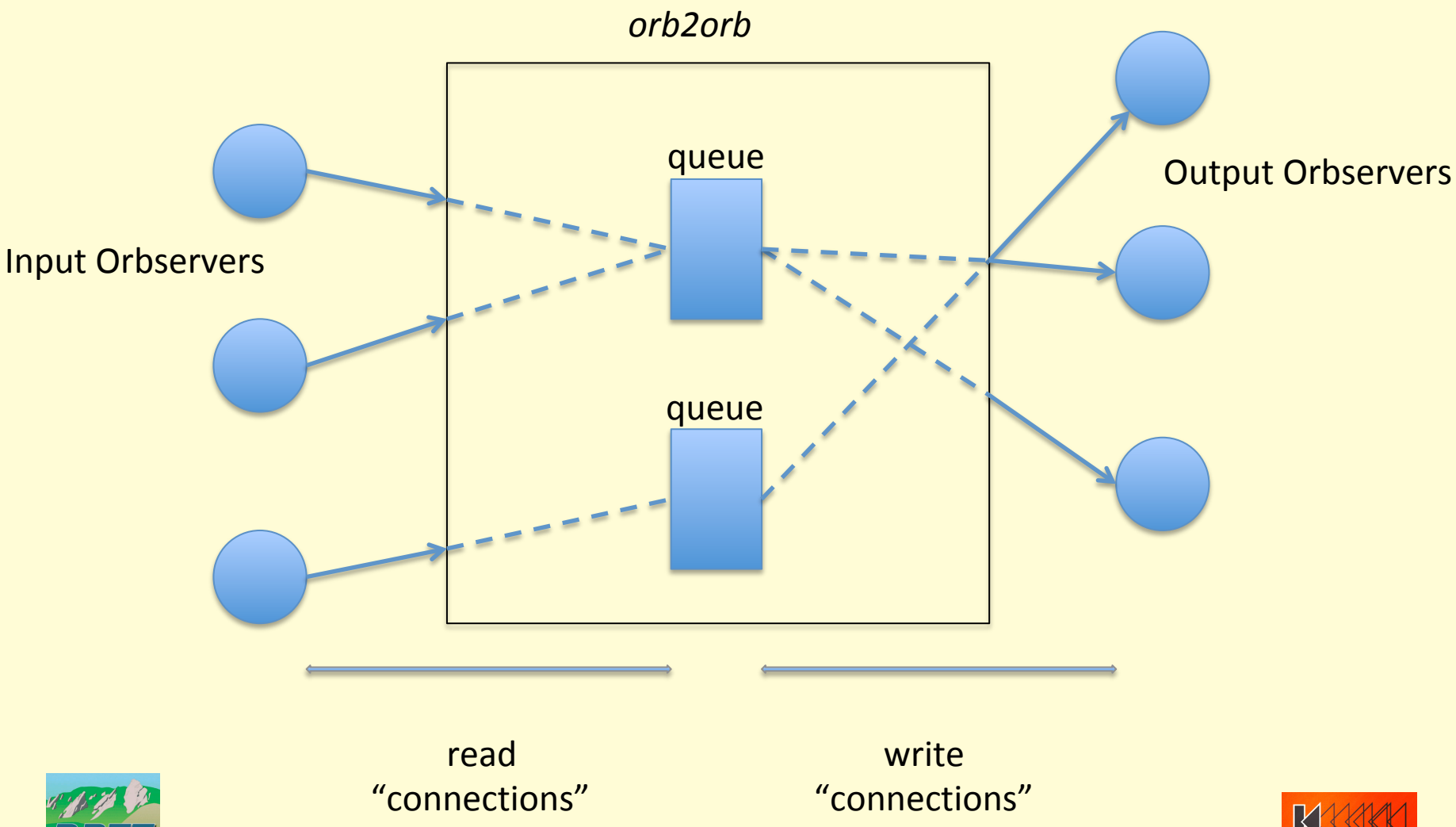


orb2orb: new architecture

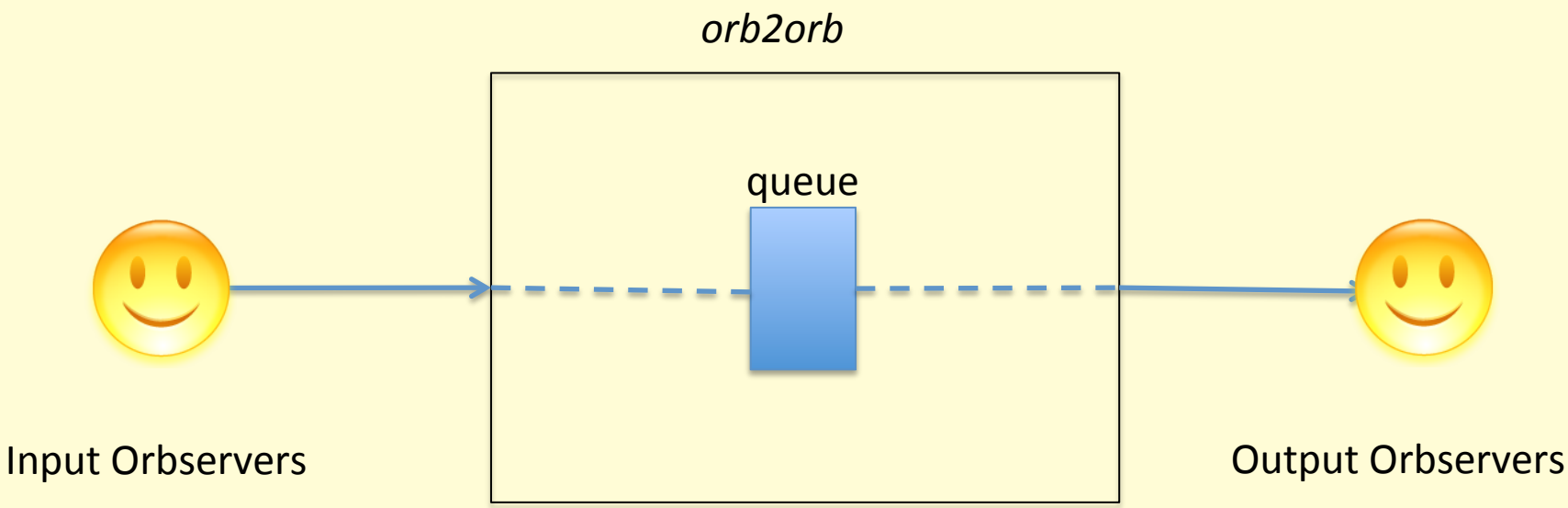


- Separate the connection into two parts:
  - The “read” half
  - The “write” half
- Configure each connection independently
- Add an internal *queue* to buffer data
- Allows you to acquire once, distribute to many destinations
- Allows you to fine-tune outputs
  - different match expressions to different outputs

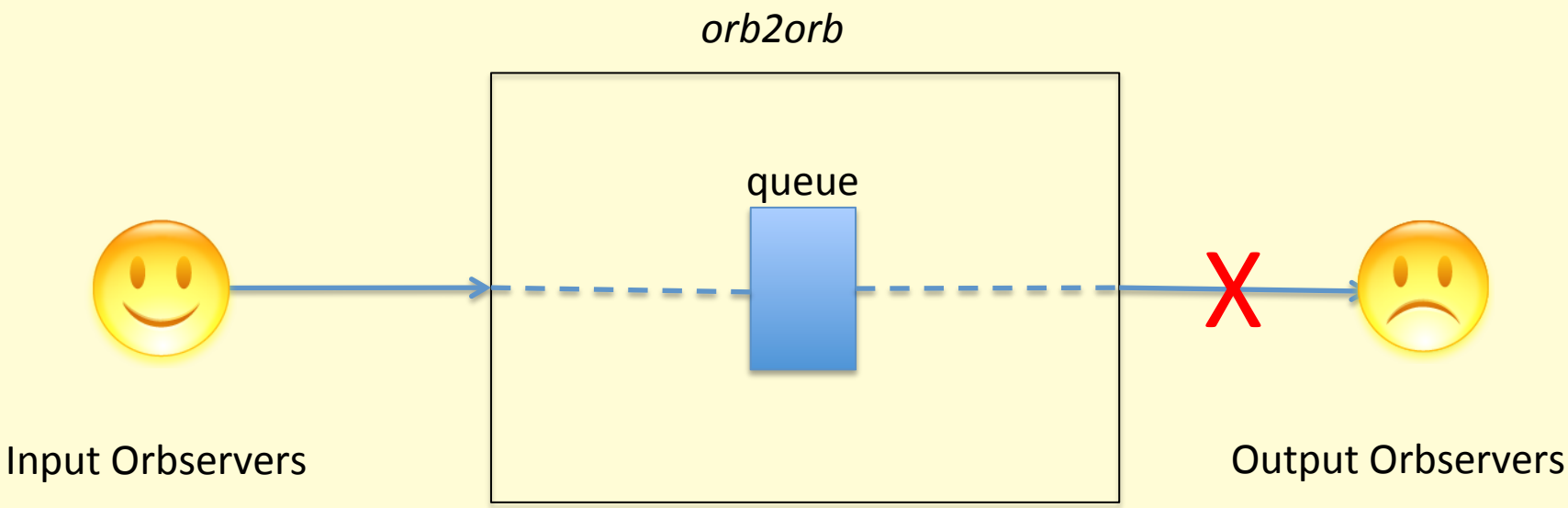
orb2orb: new architecture



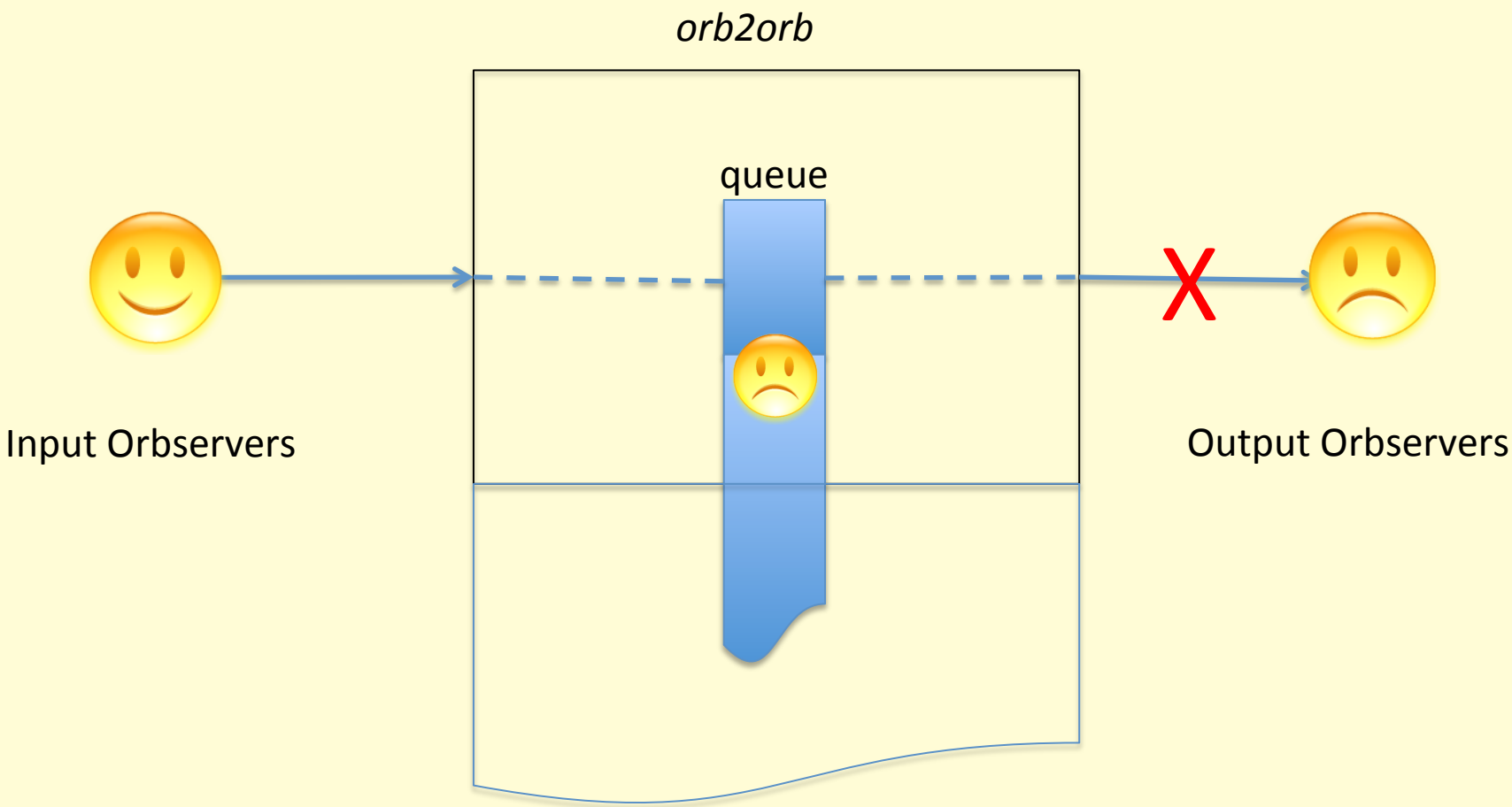
orb2orb: new architecture



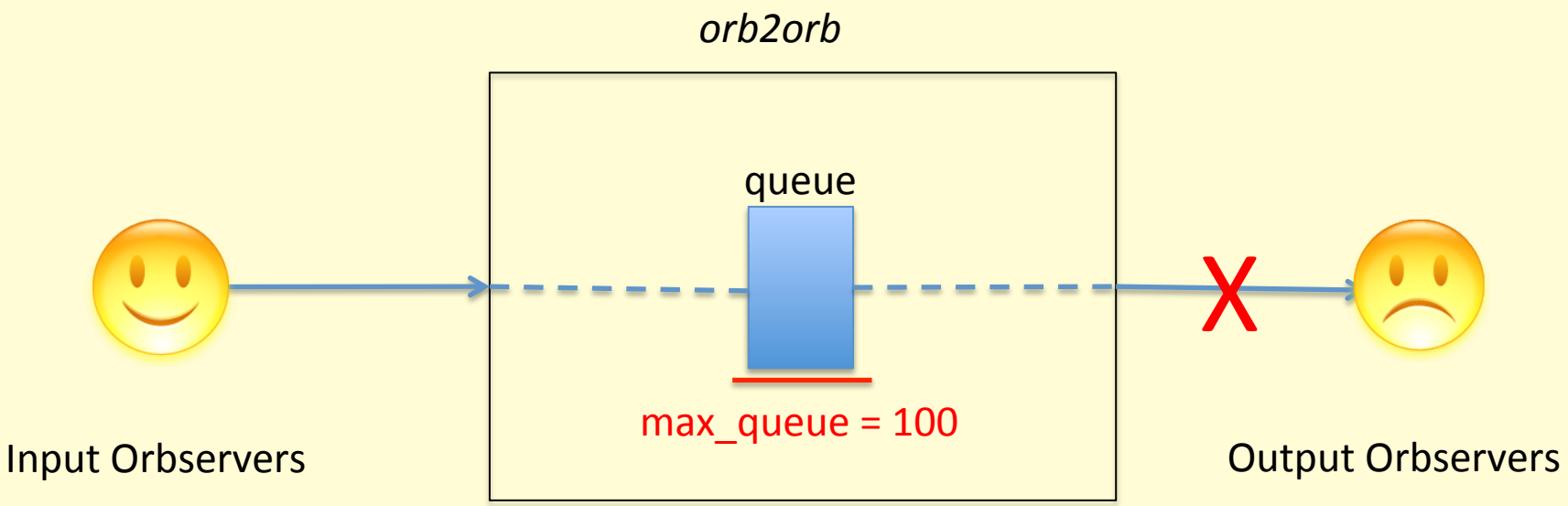
orb2orb: new architecture



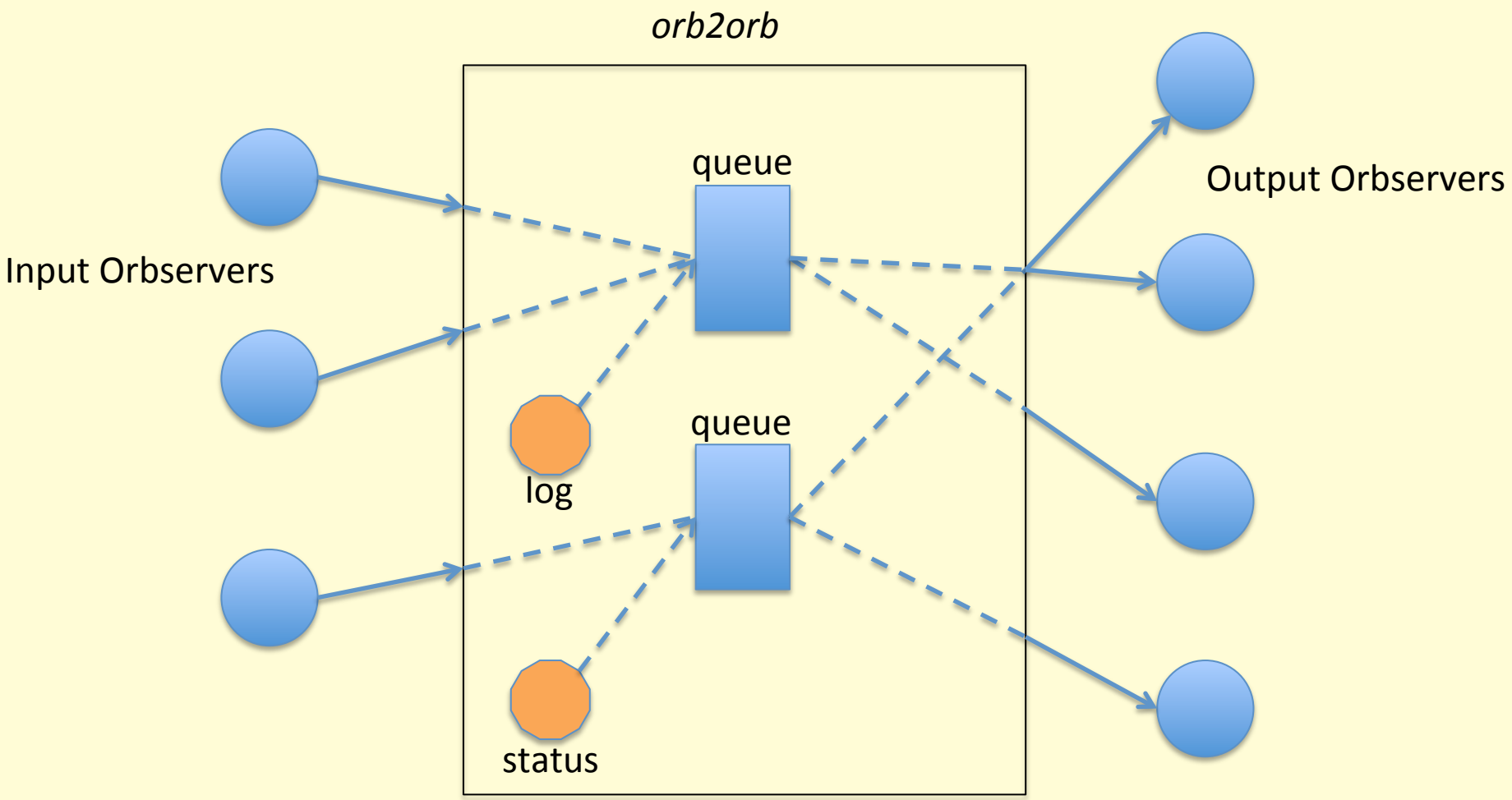
orb2orb: new architecture



orb2orb: new architecture



orb2orb: new architecture



orb2orb: dlmon output

dlmon

File Views Windows

dlname	comt	orbname	dir	queue	nq	nsn	nsnw	npk24	nrjo24	nrjn24	nrju24	aps	pr	dr	tp	runtm	SLT	dltncy	rss
1/orb2orb	rint	bbarray.ucsd.edu:gsn@	->read	mainq		181	181	1,433	0	0	84	532	11.50	49k	1.28	01m57s	00s	38s	4MB
2/orb2orb	rint	:gsn@	<-write	mainq	0	181	180	1,352	0	0	0	533	10.80	46k	1.29	01m57s	00s	38s	4MB
3/orb2orb	rint	:gsn@	<-write	statusq	0	1	0	57	0	0	0	759	0.47	2.9k	0.00	01m57s	00s	02s	4MB

"connection"

orb name

queue name

direction

number of packets in queue

run time

Status Latency

Data Latency

Resident Set Size (memory)

orb2orb: dlmon output

dlmon

File Views Windows

dlname	comt	orbname	dir	queue	nq	nsn	nsnw	npk24	nrjo24	nrjn24	nrju24	aps	pr	dr	tp	runtm	SLT	dltncy	rss
1/orb2orb	rint	bbarray.ucsd.edu:gsn@	->read	mainq		181	181	1,433	0	0	84	532	11.50	49k	1.28	01m57s	00s	38s	4MB
2/orb2orb	rint	:gsn@	<-write	mainq	0	181	180	1,352	0	0	0	533	10.80	46k	1.29	01m57s	00s	38s	4MB
3/orb2orb	rint	:gsn@	<-write	statusq	0	1	0	57	0	0	0	759	0.47	2.9k	0.00	01m57s	00s	02s	4MB

Added:

- *Number of sourcenames*
- *Number of waveform sourcenames*
- *Number of packets, last 24 hours*
- *Number of packets rejected, last 24 hours, too old*
- *Number of packets rejected, last 24 hours, too new*
- *Number of packets rejected, last 24 hours, won't unstuff*
- *Average packet size (bytes)*
- *Packet rate (packets per second)*
- *Data rate (bits per second)*
- *Throughput (ratio of seconds acquired to real-time elapsed)*

orb2orb: dlmon output

dlmon

File Views Windows

dlname	comt	orbname	dir	queue	nq	nsn	nsnw	npk24	nrjo24	nrjn24	nrju24	aps	pr	dr	tp	runtm	SLT	dltncy	rss
1/orb2orb	rint	bbarray.ucsd.edu:gsn@	->read	mainq		181	181	1,433	0	0	84	532	11.50	49k	1.28	01m57s	00s	38s	4MB
2/orb2orb	rint	:gsn@	<-write	mainq	0	181	180	1,352	0	0	0	533	10.80	46k	1.29	01m57s	00s	38s	4MB
3/orb2orb	rint	:gsn@	<-write	statusq	0	1	0	57	0	0	0	759	0.47	2.9k	0.00	01m57s	00s	02s	4MB

dlmon: All logs

Freeze Unfreeze Done

```
Thread 'connection_1/orb2orb_pre': Connected to orb <inputorb ='bbarray.ucsd.edu:gsn@'> for reading
Thread 'connection_1/orb2orb_pre': Matched 253 sources after selecting on expression '@pf/gsnlist'
Thread 'connection_1/orb2orb_pre': Repositioned orb <inputorb ='bbarray.ucsd.edu:gsn@'> from statefile 'state/GSNimport
```

orb2orb: command line

```
orb2orb   [-v]                          [CURRENT SYNTAX]
          [-m match]
          [-p pf]
          [-r reject]
          [-S statefile]
          [-t targetname]
                    [[orbtag orbname] ...] [start-time [period|end-time]]

orb2orb   [-v]                          [LEGACY SYNTAX]
          [-m match]
          [-p pf]
          [-r reject]
          [-S statefile]
          [-t targetname]
                    orbin orbout [start-time [period|end-time]]
```

orb2orb: command line

- Example from *rtdemo* GSN:

```
orb2orb -v -S state/GSNimport  inputorb  bbarray.ucsd.edu:gsn  outputorb  :gsn
```

- "orbtag" parameters label each actual orbname
  - just as in *q3302orb*, *altus2orb*

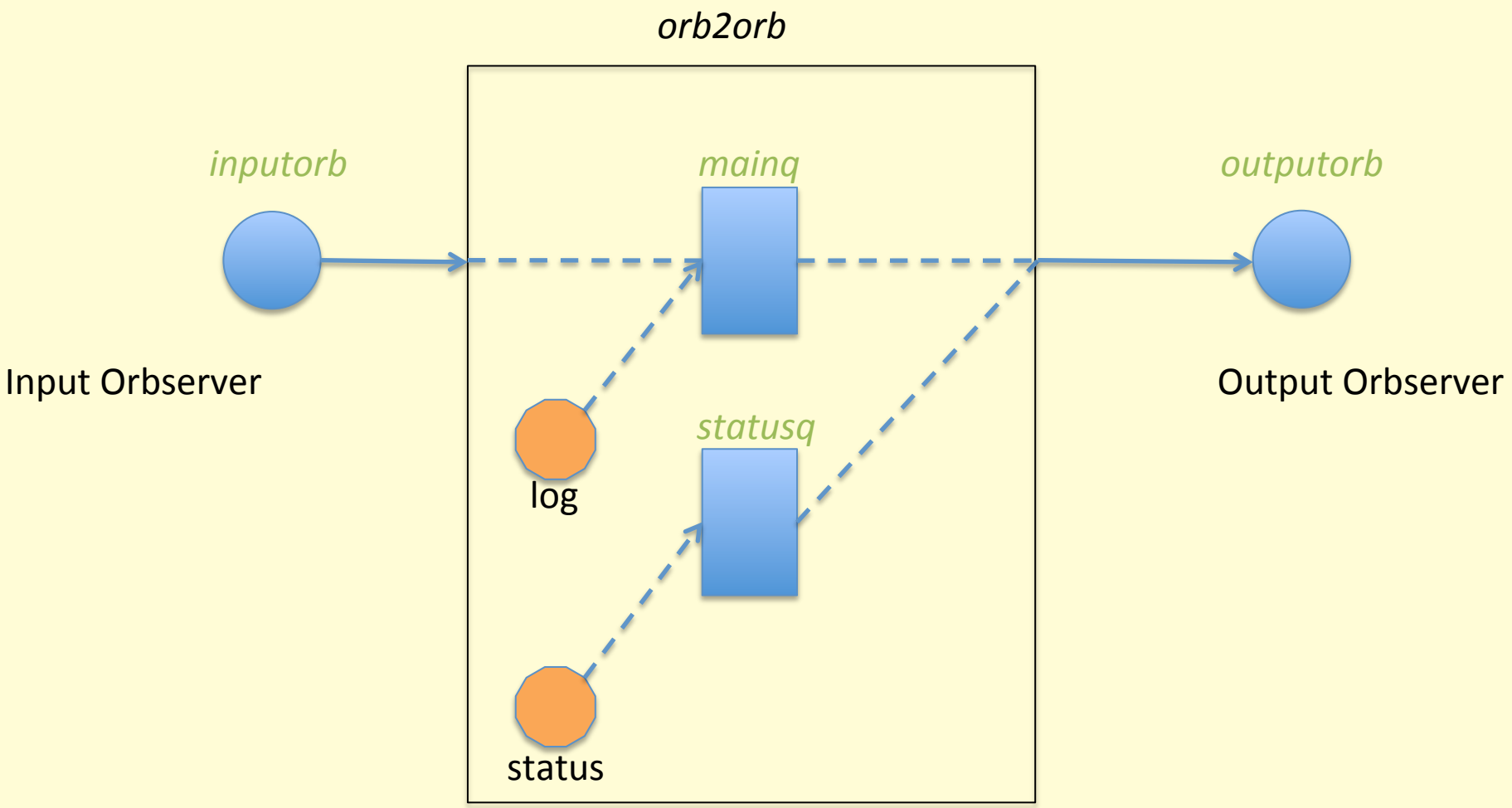
orb2orb: parameter file

```
connections &Tbl{
        &Arr{
                read_from_orbtag        inputorb
        }
        &Arr{
                write_to_orbtag         outputorb
        }
}
```

orb2orb: parameter file

```
connections &Tbl{
        &Arr{
                read_from_orbtag        inputorb
        }
        &Arr{
                write_to_orbtag         outputorb
        }
        &Arr{
                read_from_queue         statusq
                write_to_orbtag         outputorb
        }
}
```

orb2orb: default orb2orb.pf



orb2orb: parameter file

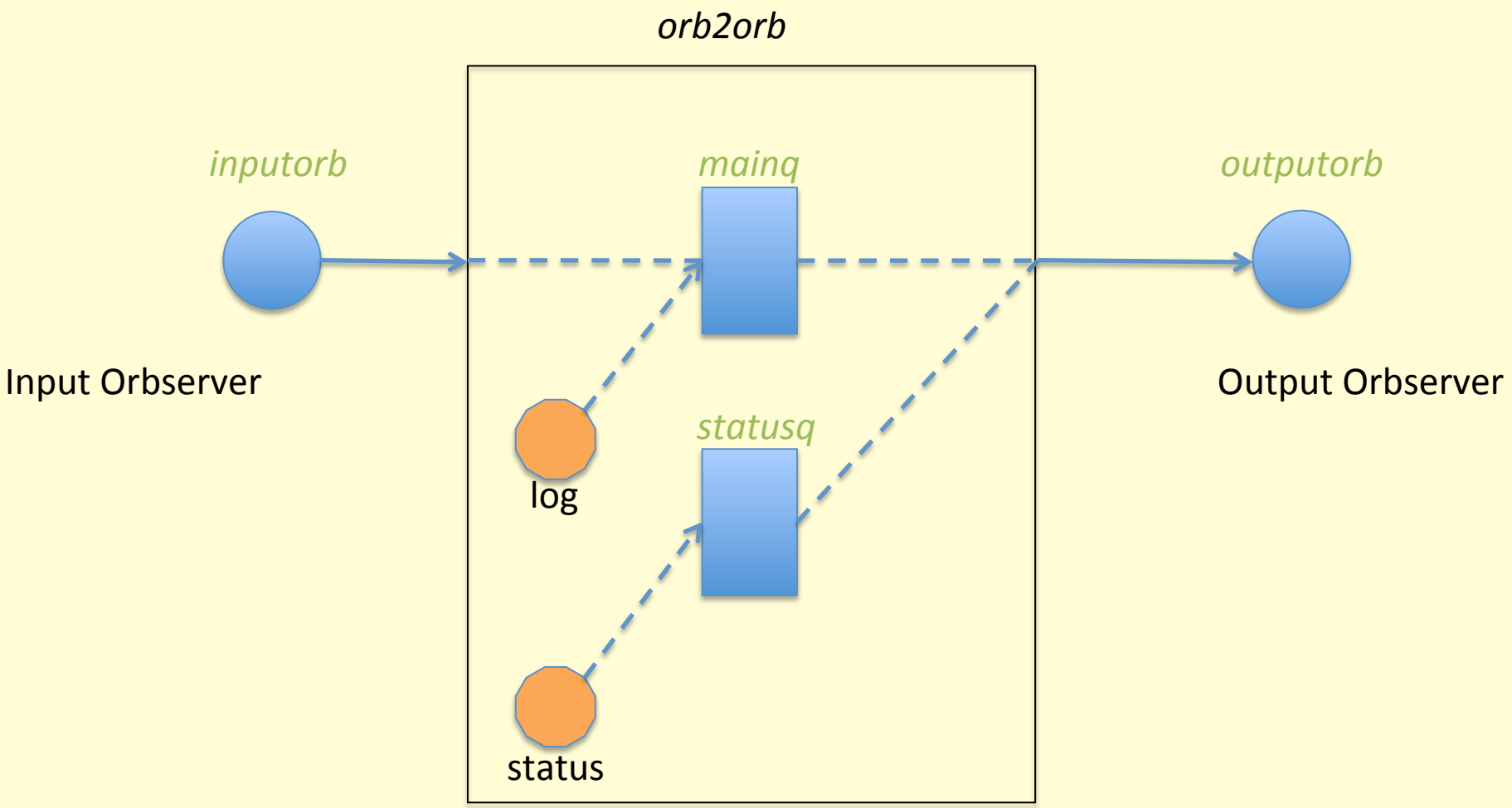
```
connections_defaults &Arr{
    read &Arr{
        read_from_orbname
        read_from_orbtag
        write_to_queue              mainq
        starttime
        endtime
        too_old
        too_new
        check_unstuff               false
        suppress_unstuff_errors     false
    }
    write &Arr{
        read_from_queue             mainq
        write_to_orbname
        write_to_orbtag
        max_queue                   100
    }
    shared &Arr{
        name                        auto
        run                         true
        match
        reject
    }
}
```

orb2orb: parameter file

```
connections_special &Arr{
    status_create &Arr{
        run                 true
        write_to_queue      statusq
    }
    log_create &Arr{
        run                 true
        write_to_queue      mainq
    }
}

time_intervals_sec &Arr{
    pfstatusreport          2
    internal_timeout        1
    shutdown_grace_period   15
}
```

orb2orb: default orb2orb.pf



orb2orb: switching advice -- options

1. Run in legacy mode
   – ***orb2orb*** *bbarray.ucsd.edu :*
2. Add orbtags
   – ***orb2orb*** *inputorb bbarray.ucsd.edu outputorb :*
   – (supported by default parameter-file)
3. As above, plus start adding other connections to parameter-file, adding more orbtags
4. Don't switch [*not recommended*]:
   – ***orb2orb_dep*** *bbarray.ucsd.edu:gsn :*

orb2orb: planning for next year

- Time and Multiplex repackaging
- Point-Of-Contact ("POC") Capability
- Command-and-control (*dlcmd*)
- Duplicate packet rejection
- Additional legacy option & parameter support
- Failover to alternate input orbservers

Thank You!

Questions?