EIOW – Exa-scale I/O workgroup (exascale10)

Meghan McClelland
Peter Braam

Lug 2013

xyratex

Problem Statement

- Large scale data management
  - *is fundamentally broken*
  - *but functions somewhat successfully as an awkward patchwork*

- Current practices
- Future needs
- What is wrong with current approaches?
- What framework can be built to handle this?

The Exa-scale IO Workgroup (EIOW) has has worked with application developers and storage experts and made exciting progress.

EIOW (exascale10) mission

- Let HPC ***application*** experts explain requirements for next generation storage

- Architect, design, implement an open source set of exa-scale I/O middleware

- So far around 40 participating organizations

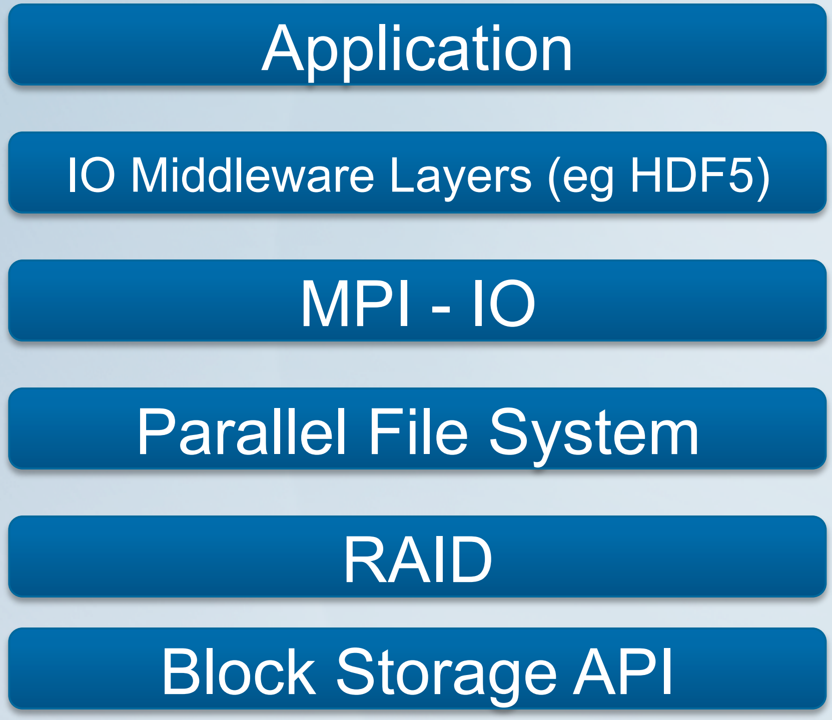
EIOW Participants (apologies – some probably omitted)

- University of Paderborn
- University of Mainz
- Barcelona Supercomputing (BSC)
- DDN
- Fujitsu
- TU Dresden
- University of Tsukuba
- Hamburg University
- TACC
- NCSA
- HDF group
- MPG/RZG
- Juelich
- Goethe Universitat Frankfurt
- ZIH
- DKRZ
- Netapp
- Tokyo Institute of Technology
- Micron
- Xyratex
- DSSD
- Sandia
- PNNL
- Cray
- DOE
- PSC
- LRZ
- HLRS
- CEA
- T-Platforms
- Partec-EOFS
- STFC
- Intel
- NEC

- EIOW is an open effort
  - European Open File System (EOFS) supported workgroup since inception
  - A core EOFS project (like Lustre) since Sep 2012
  - Everything is being published on the web
    - And actively being copied and amended
  - We will move in the direction of Internet Engineering Task Force (IETF) style controlled openness

- EIOW intends to be a ubiquitous middleware
  - An agreed, eventually standardized API for applications & data management
  - We hope to be an implementation of choice for researchers to study, amend, influence and change
    - Such research projects are now numerous
  - **A storage access API allowing storage vendors to bolt it onto their favorite data object and metadata stores**

Middleware issues



Application

IO Middleware Layers (eg HDF5)

MPI - IO

Parallel File System

RAID

Block Storage API

- There are 100’s of middleware packages, sometimes layered
- Application developers regard them as very useful and convenient
- They generally are very difficult to get working well
- This is not ready for future hardware
- The stack isn’t working well

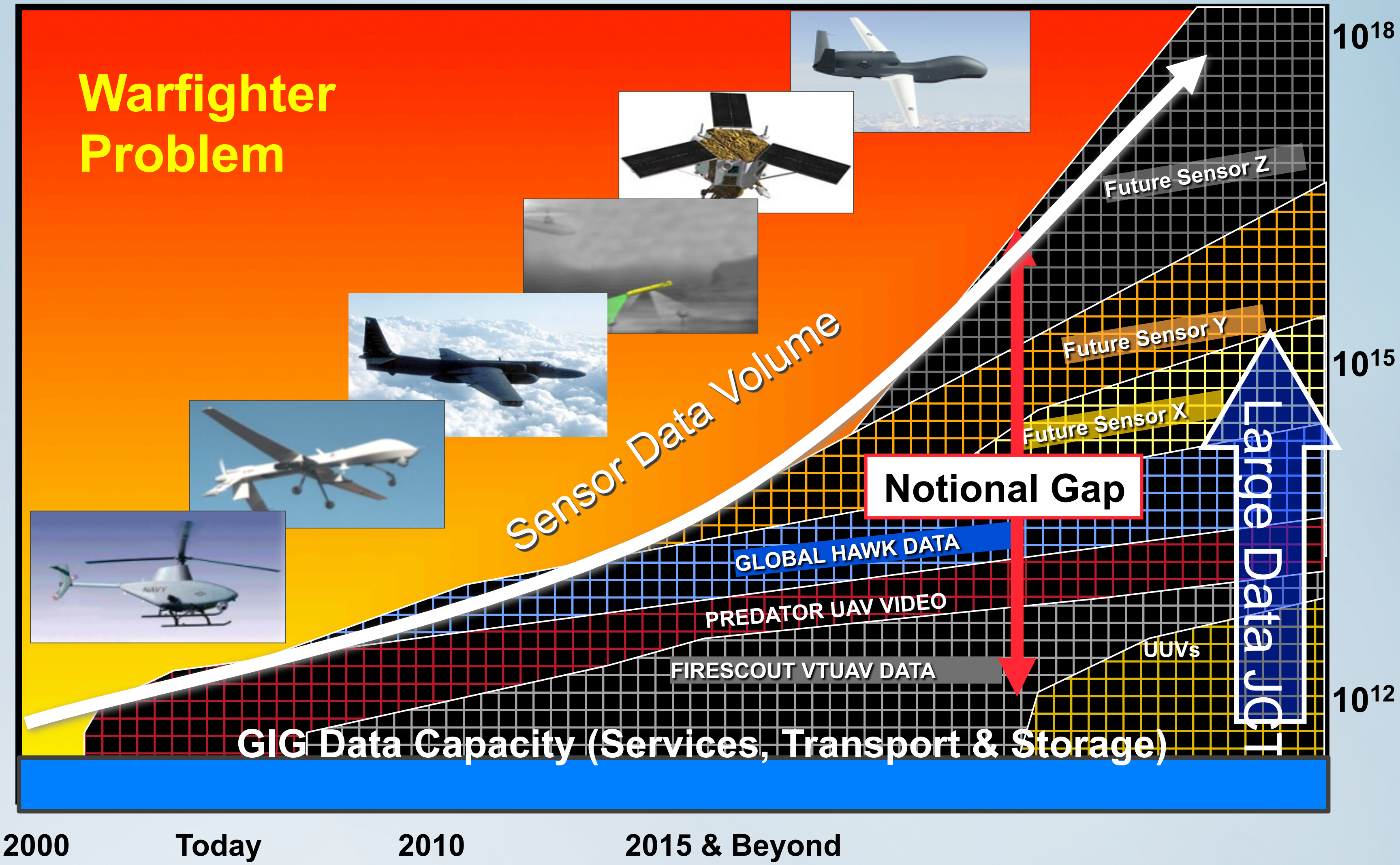
Middleware issues

- Proliferation of middleware packages – 100's
  - Many with a great deal of overlap
  - MPI-IO, PLFS, HDF5, NetCDF, Hercule, …….
- Many have strengths and weaknesses
  - E.g. HDF5 is very highly regarded
  - Because there is no stack they are nearly impossible to debug
- They re-implement major parts of file systems
  - Leads to inefficiencies, incorrectness, huge code bases
  - Nearly impossible to define HA properly
- Neither file systems nor middleware are ready for new hardware – particularly memory class storage

10PF – 100PF – 1EF

- 10PF
  - handled by large (mostly Lustre) storage systems – 1TB/sec
  - several billions of files
- 100PF
  - Flash cache approach – 10 TB/sec
  - Flash takes the bursts / Disks more continuously used
  - Takes ~ 20,000 disks (0.5MW / lots of heat / lots of failed drives)
  - Probably a metadata server becomes a scalability limit
- 1EF – the ***gap***
  - The paradigm appears to break: 100K drives is not acceptable
  - Most data can no longer make it to disks
  - What data management can help?

Big Data in the Military



xyratex

Future Needs

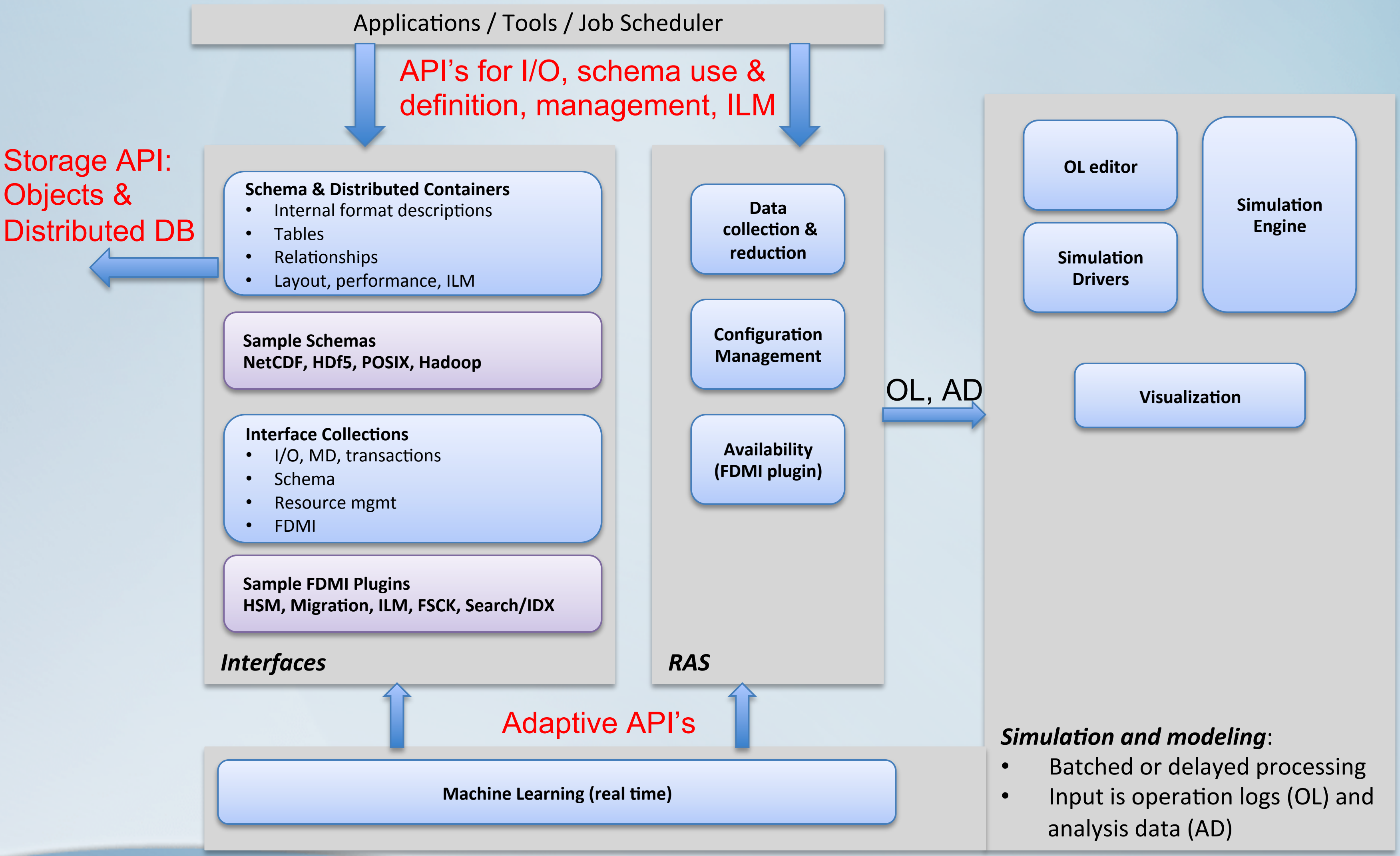
- Technology revolutions
  - File system clients will have ~10,000 cores
  - Architectures will be heterogeneous
  - Flash and/or PCM storage leads to tiered storage
  - Anti revolution – disks will only be a bit faster than today

- Tiered storage, in part memory class storage
- Data management to move less data to drives
- Scale performance 100x from today

- Pre-SQL (1972) databases were in this situation
  - We need manageable API's for unstructured data

- EIOW is an emerging framework
  - Providing rich I/O and management interfaces
  - Platform to build layered I/O applications efficiently, correctly
    - E.g. HDF5 metadata without layering it on other file systems
    - Logging and analytics through the stack
    - Transactions, data integrity through the stack
    - Not a 1980's approach to availability
- What we've seen is that most requirements can be addressed as adding plugins to a base system

Component Decomposition



Applications / Tools / Job Scheduler

API's for I/O, schema use & definition, management, ILM

Storage API: Objects & Distributed DB

Schema & Distributed Containers

- Internal format descriptions
- Tables
- Relationships
- Layout, performance, ILM

Sample Schemas
NetCDF, HDf5, POSIX, Hadoop

Interface Collections

- I/O, MD, transactions
- Schema
- Resource mgmt
- FDMI

Sample FDMI Plugins
HSM, Migration, ILM, FSCK, Search/IDX

Interfaces

Data collection & reduction

Configuration Management

Availability (FDMI plugin)

RAS

OL, AD

OL editor

Simulation Engine

Simulation Drivers

Visualization

Simulation and modeling:

- Batched or delayed processing
- Input is operation logs (OL) and analysis data (AD)

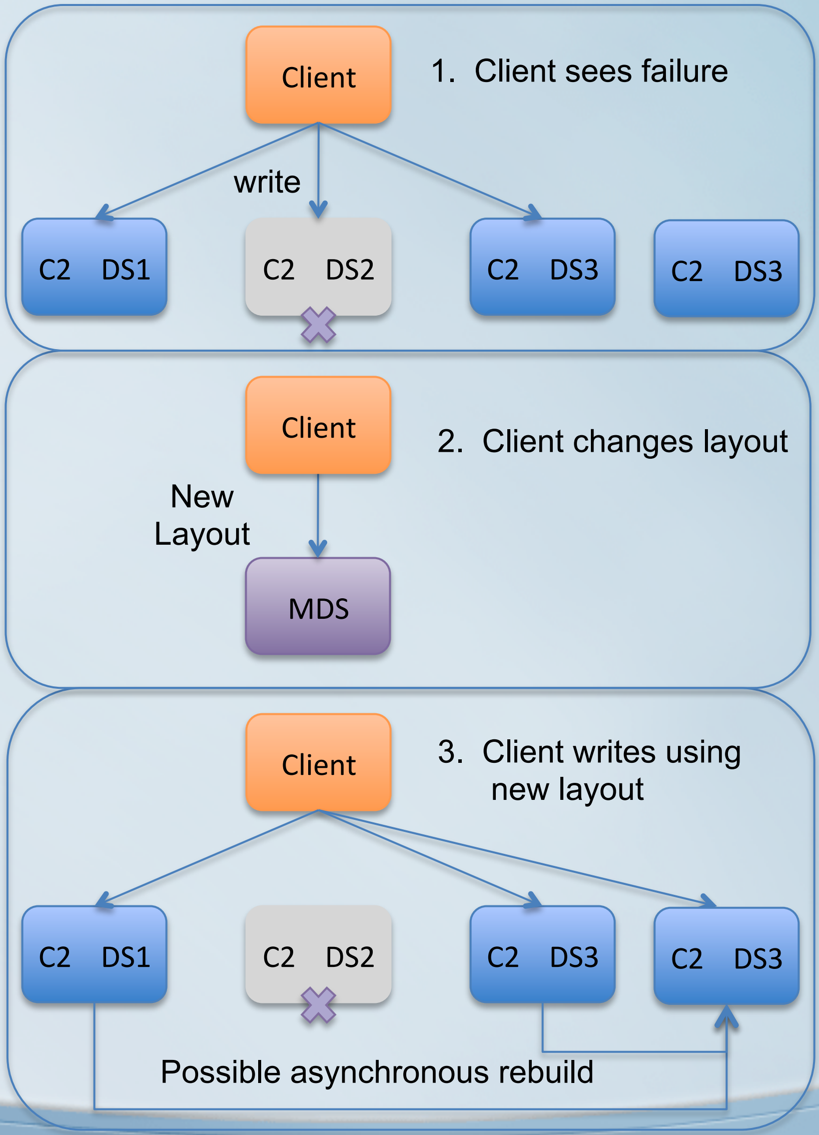
Adaptive API's

Machine Learning (real time)

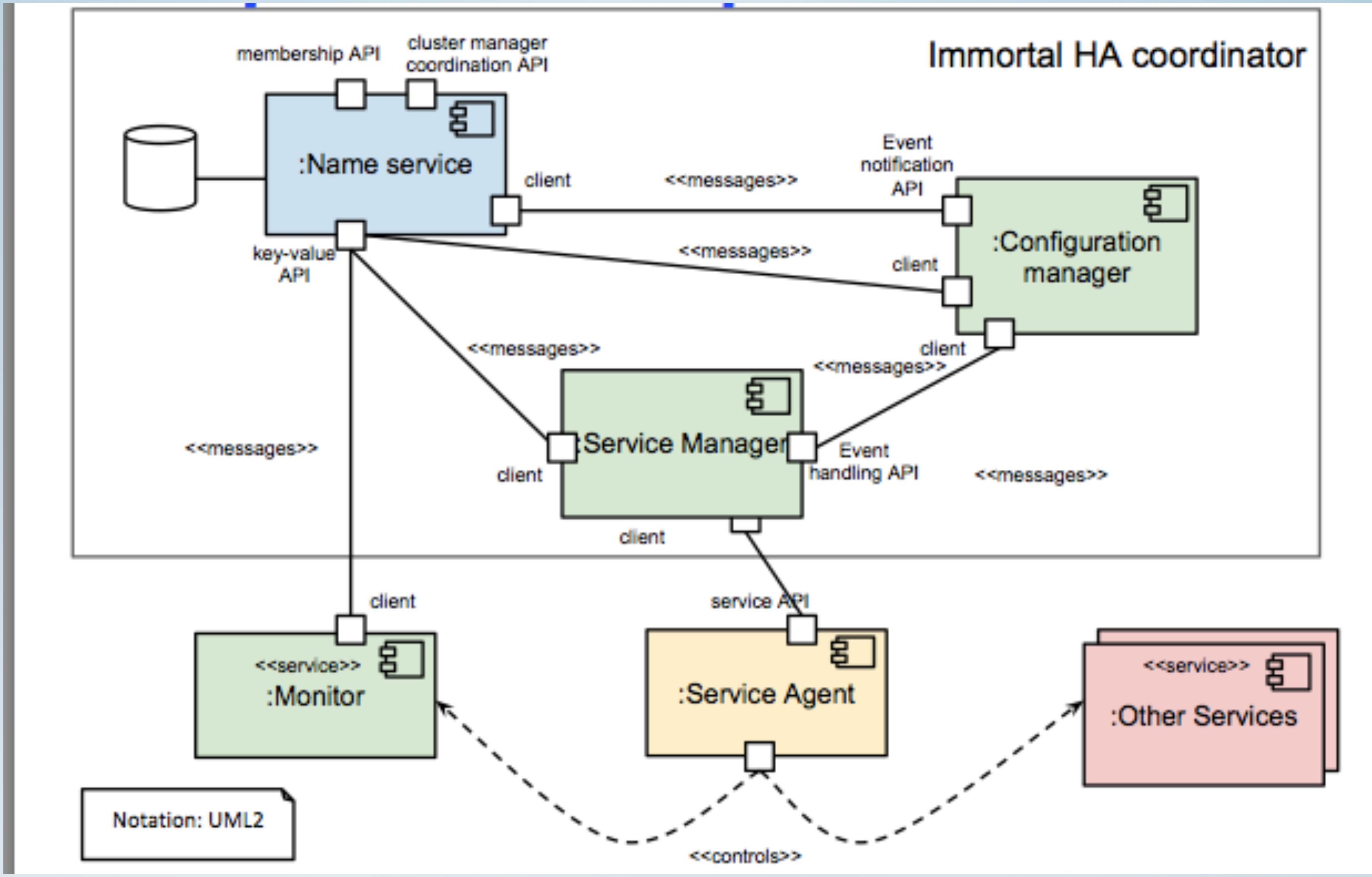
xyratex

Non Blocking Availability

- Failures will be common
  - in very large systems
- Failover
  - Wait until resource recovers
  - Doesn't work well
- Instead: focus on availability
  - No reply: change resource
  - Adapt layout
  - Asynchronous cleanup



HA


Immortal HA coordinator
membership API
cluster manager coordination API
:Name service
client
<<messages>>
Event notification API
:Configuration manager
key-value API
<<messages>>
client
<<messages>>
client
<<messages>>
<<messages>>
:Service Manager
client
Event handling API
<<messages>>
client
client
service API
<<service>>
:Monitor
:Service Agent
<<service>>
:Other Services
Notation: UML2
<<controls>>

Workshops

- Requirements Gathering
  - 1st workshop (Munich 02/12)
  - 2nd workshop (Portland 4/12)
  - 3rd workshop (Tokyo 5/12)
- Architectural Design, Funding
  - 4th workshop (Barcelona 9/12)
- Alternative Approaches
  - 5th workshop (Salt Lake City 11/12)
- Design Discussion of Code Components
  - 6th workshop (San Jose 2/13)

- Next workshop – Leipzig Germany June 20th 2013
- Implementation Level Design, Future Efforts

Current Efforts

- Community – phone calls, new web site

- Prototype code is being developed
  - Core system (schemas, interfaces, HA)
  - Simulation / monitoring

- Evaluate ideas with prototypes
  - Research proposals
  - Evaluation in next generation systems

Conclusion

A framework like SQL for HPC data / big data is 40 years overdue

- We aim to change that….

Thank You

Meghan_Mcclelland@xyratex.com

EIOW Website:

https://sites.google.com/a/eiow.org/exascale-io-workgroup/