Continuous Observation Platforms.
Common Data Products.
QA/QC.

MAINTAIN a common sensor suite across class.
OPERATE resident sensors to run continuously.
PERFORM near real-time QC to promote QA
The datapresence problem space – where do we fit?

**High Complexity / Unique:**

**Problem**
Too many techs not enough shipboard science

**Solution**
connect shoreside science with A/V tech

**Lower Complexity / More Common:**

Traditional seagoing technician/scientist ratio – science outnumbers tech support

**Hidden Problem**-
Awash in data, logistics, etc.

**Solution** –
Turn data to information off the ship
The Datapresence Problem Statement: In Other Words, Why?

By nature seagoing research is resource limited:
- Time at sea – you’ve got the time you’ve got and only one chance
- Active participants – you’ve only got so many bunks
- Technology on hand – can’t easily scale up and out
- Connectivity – information, social, other..

Potential Impacts:
- Reduced situational awareness
- Reduced data quantity & quality
- Unrealistic expectations & workload
- Impaired ability to act adaptively
- Reduced access to traditional support networks
Needs Assessment:

AUDIENCE - Who are we targeting?  YES.


PRODUCT – What are we offering?  YES...

SERVICE – How are we providing it?  YES....
RCRV Datapresence Functional Model

SHIP SIDE
- Sensors
- Data Network
- External Data Sources
- Data Acquisition
- Private Data Store

Sat Link
512kbps - 2 mbps typical
User Space: Mirror Content & Service Delivery on Ship and Shore

SHORE SIDE
- QA/QC and Bundling
- Repositories
- External Data Sources
- Private or CDN Data Store

Tools & Services Implemented On Web Framework For Distributed Access
- Scheduling & Data Access
- Data System QA/QC Visualization
- Geospatial Visualization Map Services
- Event Notification
- Data Replication Services
- Telepresence Services

Indicates Existing Datapresence Components

Indicates New Datapresence Integration
Service Requirements

• **Data Discovery** – UI has “portal like” capabilities
• **Data Access** – Erddap and other data services (map services, file shares..)
• **Chart/Plot Data Visualization** – UI time series visualizations
• **Map Data Visualization** – Sikuliaq like mapserver implementation with GMRT base layers
• **Data Replication** – mirror full resolution content to shore
• **Event Notification** – Users can create custom notifications
• **Shipboard QA/QC** – Flagging and notification
• **Shoreside QA/QC** – FTE for sensor technician oversight
Synchronizing the data store

Target Requirements:
- Synchronize all “simple” vector time series data at full resolution
- Synchronize continuously instead of episodically
- Use COTS solution if possible, don’t roll-your-own
- Use a reliable or consistent method

Options:
- File Transfer (rsync) - simple (but you need to roll your own mgmt. logic), episodic
- **Shared Database** – pub/sub model, asynchronous (store and forward changes)
- Messaging – many models, some do guarantee reliable delivery, message-oriented-middleware (MOM), again some considerable assembly required.
Database Replication

Currently Testing EDB PostgreSQL’s xDB Replication Server
- Write Ahead Logs are used to protect against data loss
- You can ship the logs to remote db and play them forward
  - Performance is great in a local area network
    - Can push logs as frequently as once per second
  - Out of the box performance isn’t so great over high latency (RTT = 850ms) and high packet loss networks.

OPTIMIZATION
- Change TCP send and receive window size, disable slow start, selective acks, etc.
- During tests on Endeavor (April 2017) we replicated 12 sensors collecting at 2 Hz
  - Utilization - Ship to Shore ~59 kbps (175 kbps spikes)
  - Gracefully handles outages: 20 minute outage (over 28,000 transaction backlog)
Data Services:

User Interface components built on the Django Rest Framework.

- Data is serialized as JSON & geoJSON
- Integrates easily into javascript plotting libraries like Highcharts, D3, etc.
- Modify URL with query parameters to: Window/Filter/Order/etc.
Data Services: For Users

Dataset Title: 
Institution: 
Information: 

Variable: 
- datetime (UTC) 
- psp (Long W) 
- pir (Long W) 

Server-side Functions: 
- distinct() 
- orderBy 

File type: 
Just generate the URL 

Submit: (Please enter update time)

Data Download

Sensor: Anemometer

Parameter(s)
- Wind Direction
- Wind Direction
- Wind Speed
- Wind Speed

Select/De-select All

Data Options
- Temporal Resolution: Full Resolution
- Date Range: Last 7 Days
- Data Format: OPeNDAP CSV (.asc)

Show ERDDAP URL

Download Data

http://sardinops.coas.oregonstate.edu:8080/erddap/tabledasp/anemo_mmast.asc?datetime,direction,direction_flags,direction_flag,speed,speed_flag,speed_flag,speed_flag,speed_flag,speed_flag,datetime>=TZ&datetime<=TZ&orderBy="datetime"
Web User Interface: Sensor Specific Time Series Plots
Next Steps:

1. Collaborate:
   - I used to hear things like “you can’t do that” and “good luck”
   - Now I’m hearing “I want to do that also”

2. Fork:
   - Fork off branches to develop alternate methods for:
     • Synchronizing the data store (file or message models)
     • Data delivery to clients (web sockets)

3. Human engineering:
   - Noting beats demonstrated success and performance
Acknowledgements

Special thanks to the shipboard scientific support teams:

*Oceanus* – Andrew Woogen, Croy Carlin, Brendan D’Andrea
*Endeavor* - Bill Fanning, Lynn Butler, and Erich Gruebel
*Sikuliaq* Science Support - Steve Hartz, John Haverlack, Ethan Roth & Bern McKiernan

..and to our other science support heroes:

URI Inner Space Center – Dwight Coleman & Derek Sutcliffe
UHDAS – Jules Hummon & Toby Martin
MGDS @ LDEO – Vicki Ferrini & Rose Anne Weissel
R2R & SAMOS

...and finally to those brave enough to share with us their ship time:

OSU – Miguel Goni, Angel White, Clare Reimers
URI – Jamie Palter
OOI – Ed Dever, Jonathan Fram, Johna Winters, and Steve Lambert