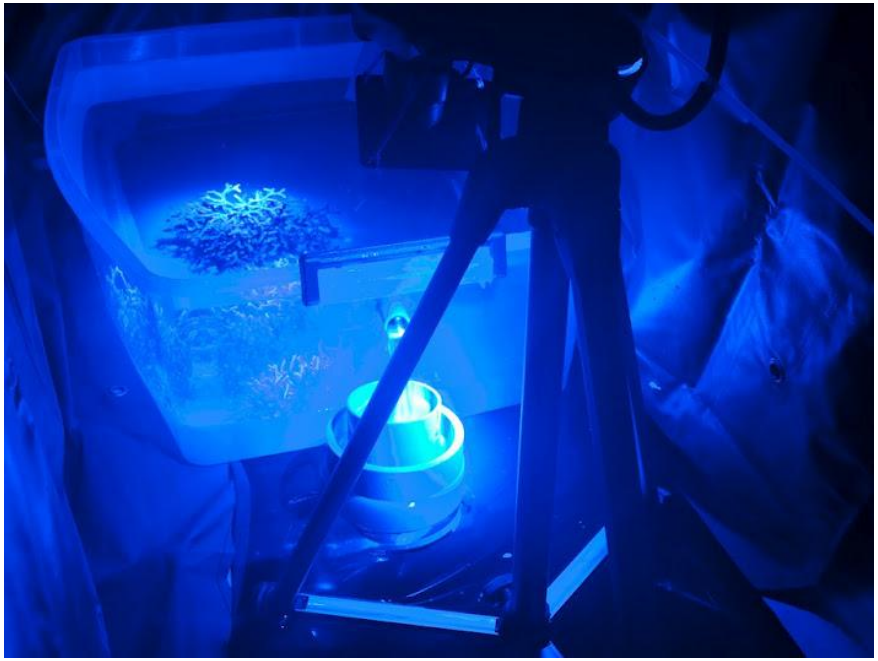
Open Source DataTurbine

Presented by Michael Nekrasov
University of California San Diego

THE PROBLEM

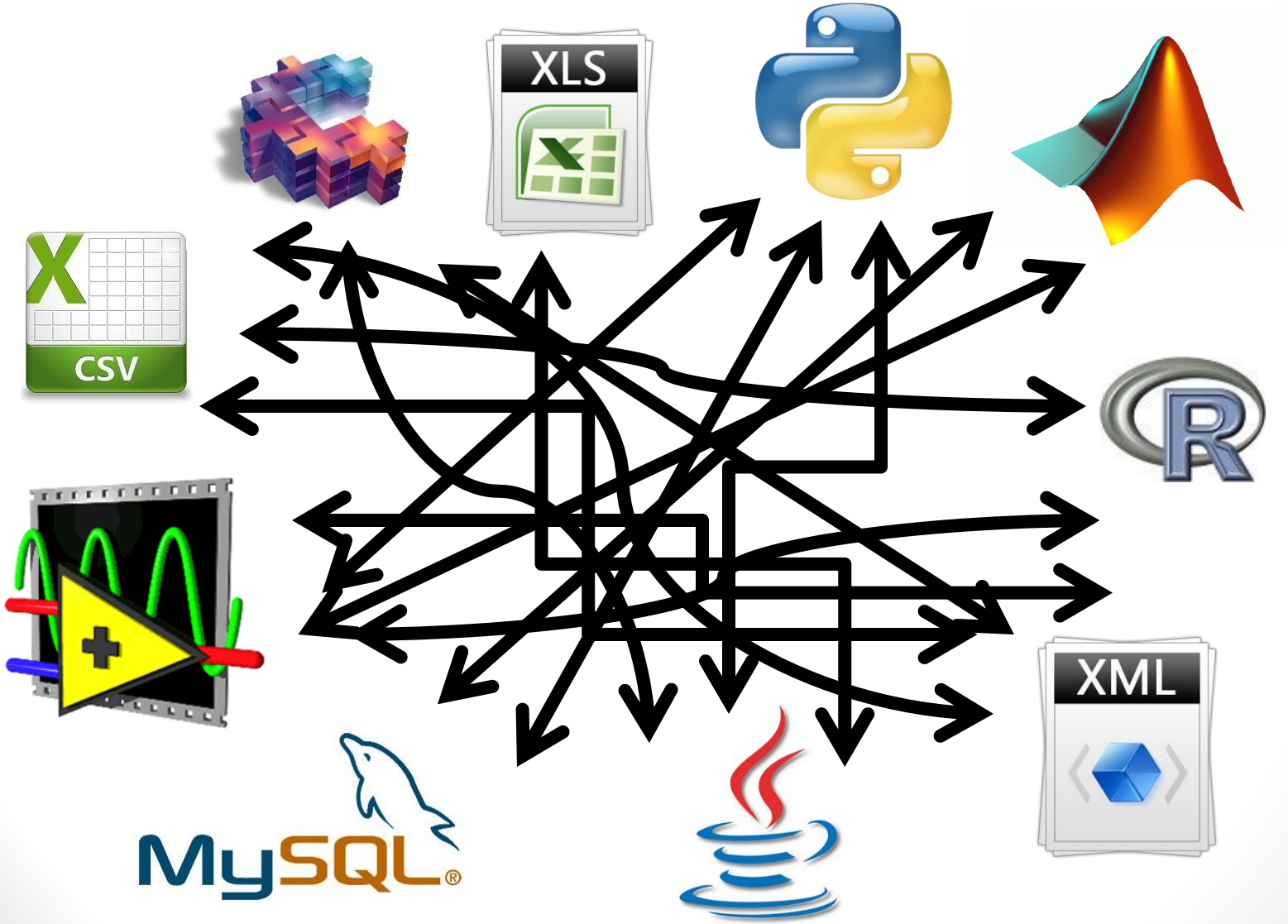
Heterogeneous Data



Data Delivery

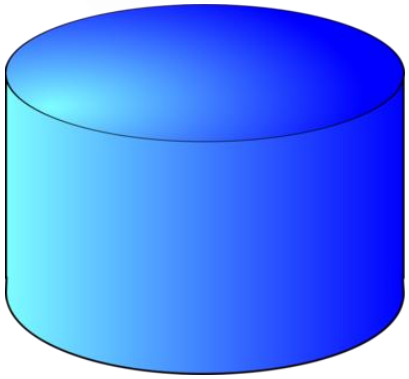
Portability

Multiple Tools

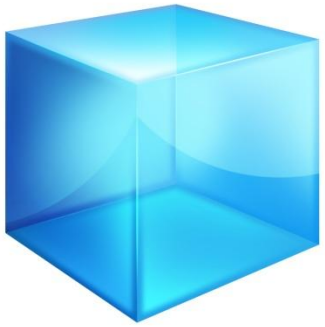


Interoperability

Site 1



Site 3

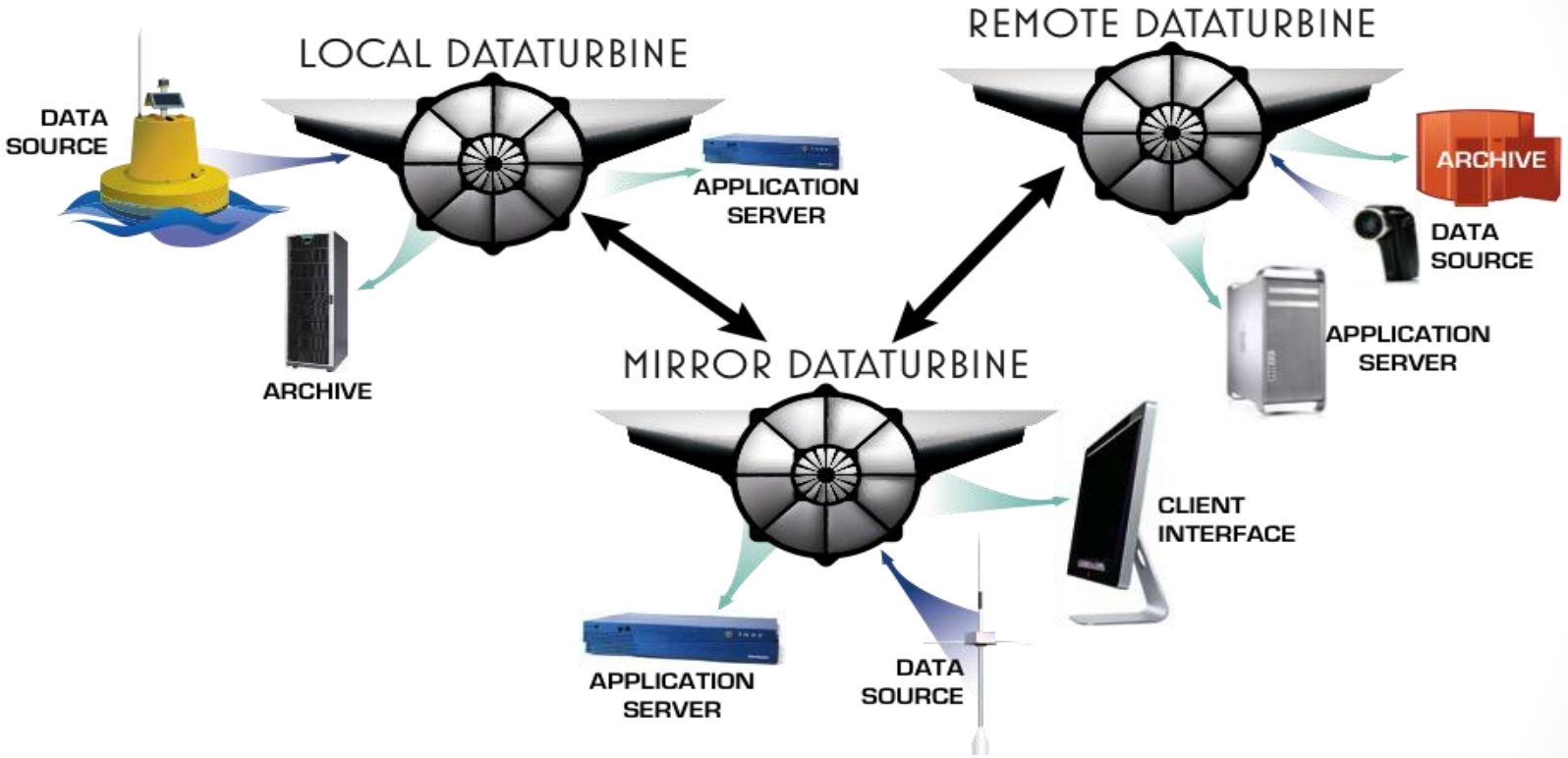


Site 2

IN A NUTSHELL

DataTurbine in a Nutshell

- R*obust real-time streaming data engine*
- Stream live data from experiments, labs, web cams and even Java enabled cell phones
- Acts as a "*black box*" to which applications and devices send and receive data
- Open source and freely available


LOCAL DATATURBINE
REMOTE DATATURBINE
MIRROR DATATURBINE
DATA SOURCE
APPLICATION SERVER
ARCHIVE
ARCHIVE
DATA SOURCE
APPLICATION SERVER
CLIENT INTERFACE
APPLICATION SERVER
DATA SOURCE

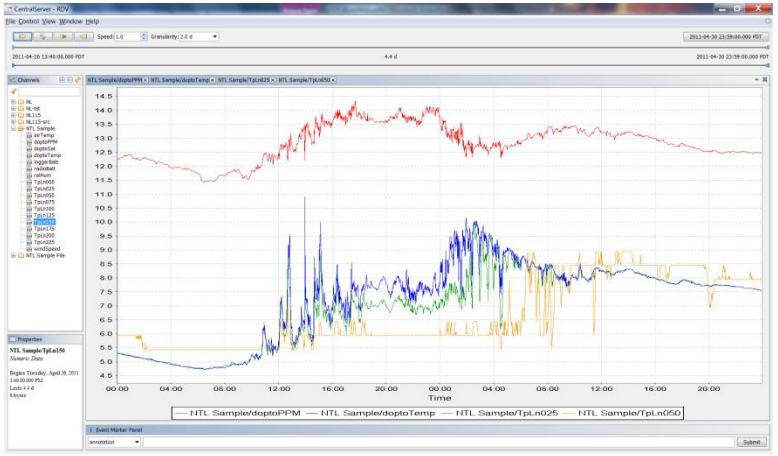
Why Use It?

- **Extendable:** It is a free Open Source project with an extensive documented API.
- **Scalable:** It uses a hierarchical design that allows a network structure that grows with the requirements of your application
- **Portable:** DataTurbine runs on devices ranging from phones & buoys to multicore servers.
- **Dependable**: Using a Ring Buffered Network Bus, it provides tunable persistent storage at key network nodes to facilitate reliable data transport
- **Community Driven**: There is also an active developer and user community that continues to evolve the software and assist in application development.

BENEFITS AND LIMITATIONS

What DataTurbine Does Best

- Reliable Data Transfer
- Real-Time Data
  - Streaming
  - Analysis
  - Visualization
  - Publication



- Cleanly works with heterogeneous data types
- Separates data acquisition (sources) from data utilization (sinks)
- Seamlessly access historical and real time data
- Synchronized access across disparate data channels
- Can work atop existing infrastructure

Benefits of Real-Time

Interactive:

- **Failure:** Ability to respond to factors on the fly. If a sensor goes bad the system registers it immediately and can be fixed (before potentially months of data are ruined).

- **Important Event:** If an event of importance occurs a team can be dispatched immediately to gather additional samples and observe the occurrence first hand.

- **Sampling:** With a real-time system its possible to change sampling rates and activate and deactivate sensors based on the data they receive.
  Ex: If one sensor detects an important event perhaps the sensors in that region need to increase their sampling rate temporarily or a camera needs to be activated.

Benefits of Real-Time

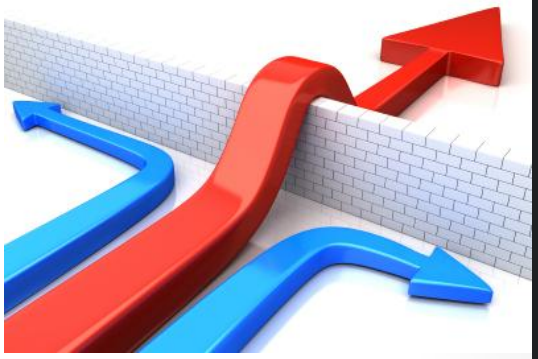
Analysis:

- There is a lot of analysis that can be performed on real-time data and in certain cases this is actually the more efficient route. Averages, correlations, and mathematical operations can be performed in real-time with ease.

- The *derived data* can be put back into DataTurbine and further utilized.

- The end result is that summary and derived data is available in in exactly the same way that the original data is.

Benefits of Real-Time



Portable:

- Adding destinations or applications is easy and transparent.

- Since data is contained as tuples (time, value, source) it is easy for any system to accept it and requires significantly less overhead then trying to read from a rigid structure such as a database.

- Once a streaming system is set up raw data, and automated analysis and quality assurance and quality control are available to any application and destination that the provider specifies the second it is available. .

Benefits of Real-Time

Public Consumption:

- The same sensor network that is monitoring an ecosystem for scientific research can display the tides and temperature of the water, the wind speed and direction, even a video feed showing the view of the forest.

- Can publish Data of multiple levels of QA/QC
  Ex: Publishing raw data in the context of approximating the weather for public as its collected and publishing the cleaned data for scientific consumption

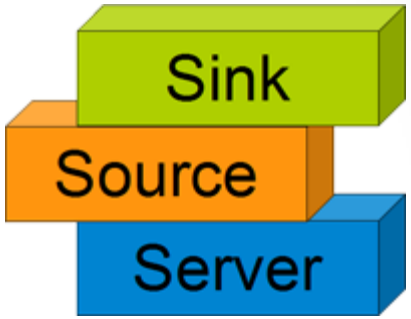
What DataTurbine Is Not Good At

- Replacing a database *(DataTurbine should be used with a database)*
- Certain tasks are better suited to real-time while other are easier to preform in a relational database.
- Out of order data (Data is accepted chronologically)
- Back-loading data

THE PARTS

The Basics Blocks

DataTurbine consists of:



- **Servers** accepting and serving up data
  Ex: rbnb.jar

- **Sources** that put data into the server
  Ex: Sensor, Camera, Application that generates data

- **Sinks** that pull data from server for visualization or analysis
  Ex: Real Time Data Viewer, Web Page, Database off-ramp

Each component can be located on the same machine or different machines, allowing for flexibility in the deployment.

DataTurbine Server

- The DataTurbine server is contained in **rbnb.jar** it is the core of DataTurbine and is used as a center point that applications (sources and sinks) interface with.

- The server is agnostic to the data it receives and can accept **heterogeneous** data types including numerical, video, audio, text, or any other digital medium.

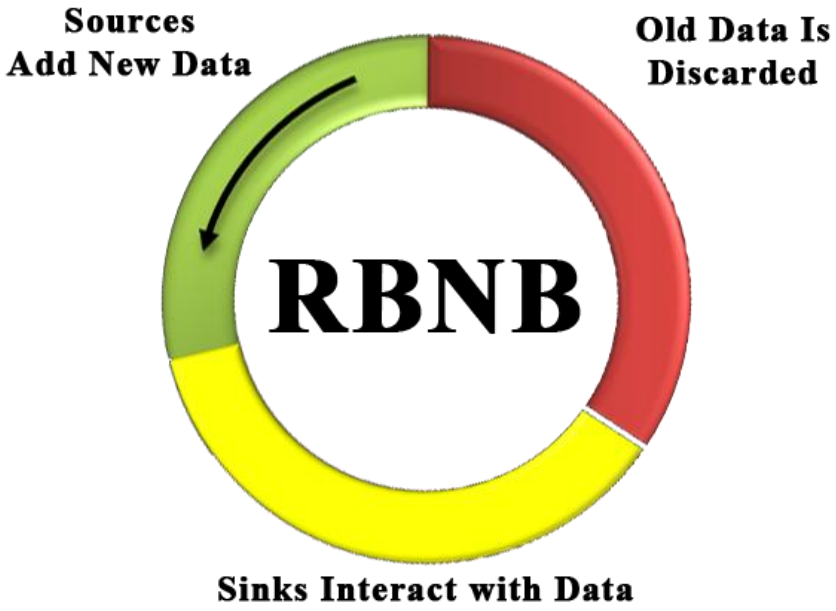
- It acts as a black box with sources adding in data and sinks reading the data out.

- Each source specify their own **archive sizes** and **cache size**.

- It can be thought of as a series of rotating disks (a ring buffer) with new data being added and old data removed when the archive becomes full.

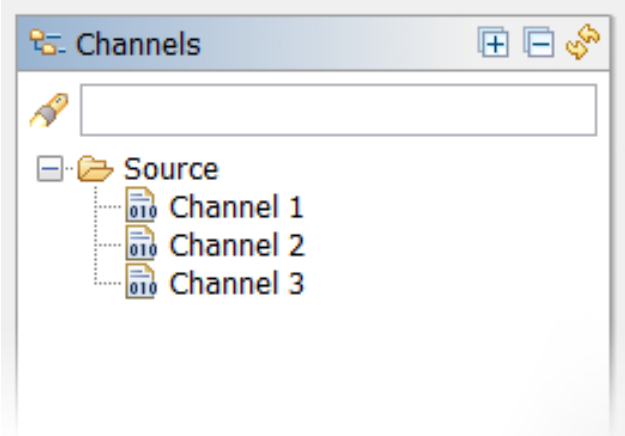
- The server expects an accurate timestamp for every data point.

- That means that each data point has to have a timestamp that is greater than the previous timestamp on record.



Source (on-ramp)

- Takes data from a target
  and puts it into a DataTurbine server
- Runs independently from the server as a separate application
  and uses the network to communicate.
- It can run on the same machine
  as the server or across the world.
- Can contain multiple channels each
  with their own data type.



- Controls its own memory and hard drive space allocation

Sink (off-ramp)

- Simply a program that takes data from a DataTurbine Server and utilizes it
  (for example brings it up in Matlab or Real-time Data Viewer or puts it into a relational database or file for permanent storage)

- Just like a source, a sink runs independently from the server as a separate application and uses the network to communicate.

- It can run on the same machine as the server or on a machine across the world.

The Sink's Perspective

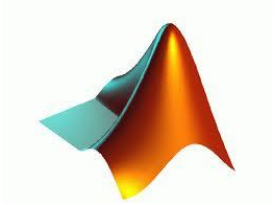
- From the sink's point of view it no longer needs to know where the data came from or how it got there.

- The data is heterogeneous and the sink could access any type of data seamlessly. It can make the decision on how to display and interpret the data via its data type (byte array, float, int, etc) as well as the **MIME Type** specified by the sink.

- A sink can issue a **request** to pull data from the server in a timeframe. A sink could also **subscribe** to a specific set of channels getting data as it becomes available.

Common Types of Sinks

- **Viewer:** An application that can be used to access and interact with the streaming data
- **Web Server:** An application that serves the data as web content for public display
- **Analysis:** Takes the data and performs some kind of manual or automated analysis
- **Export:** Exports the data into a file or set of files for distribution or integration
- **Storage:** Permanent storage in a database or as a series of files.
- **Other:** Easy to code any kind of sink that utilizes the data



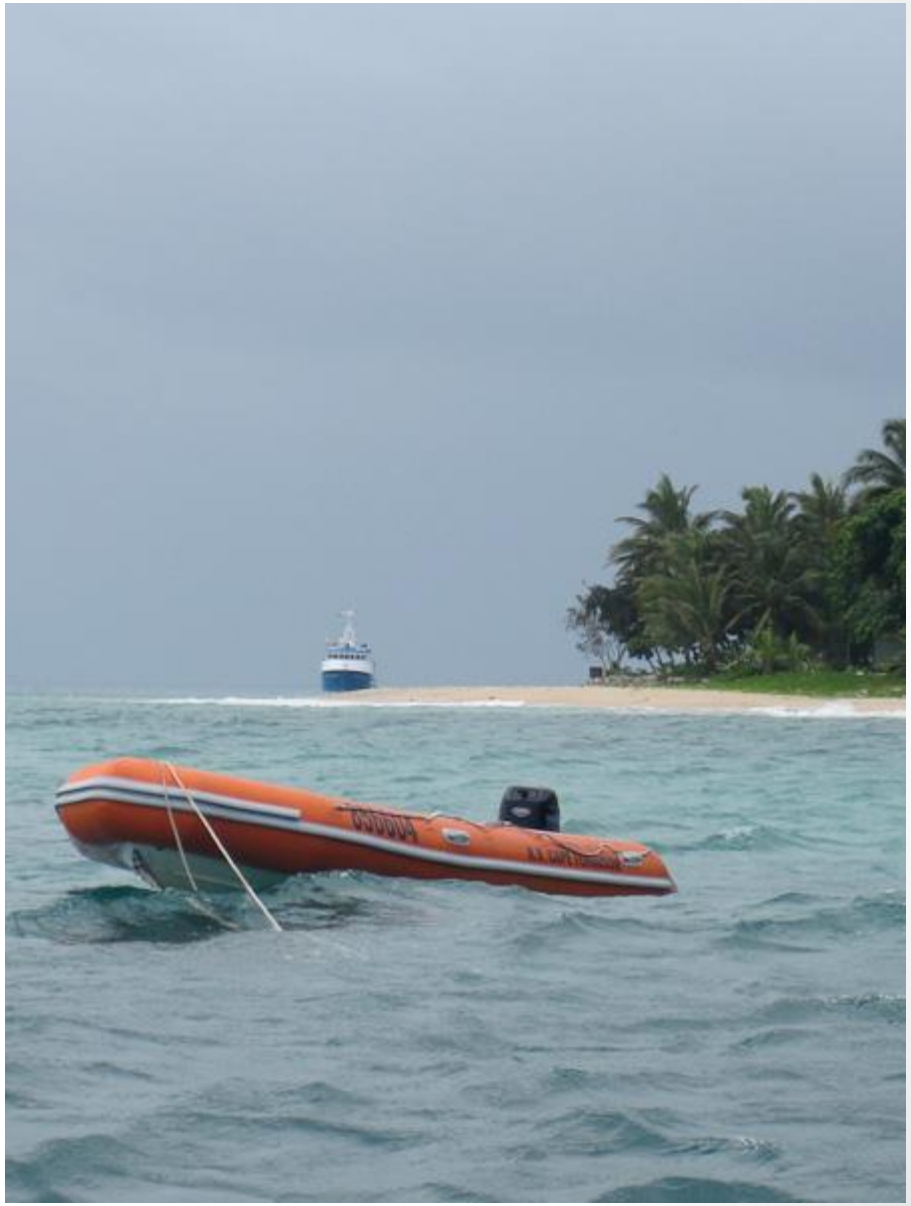
Connecting the Pieces

- The pieces are connected via local networks or internet using the IP address of the device they are running on.
- The default port for DataTurbine is *3333*, but this can be reconfigured (for example in order to run multiple servers).
- This allows each flexibility for where components reside on a network

EXAMPLE SCENARIOS

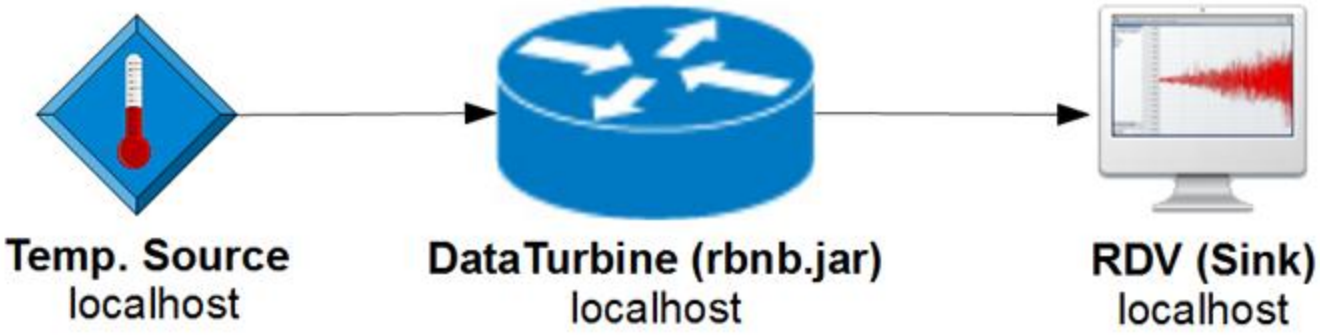
Scenario 1: Single Laptop on a Boat

- *An investigator wants to go out on the ocean for 2 hours and record the temperature there using a laptop and a thermistor hanging off the boat. He wishes to observe the data as it is collected and visualize it.*



Scenario 1: Infrastructure

- The experiment will use a DataTurbine to temporarily collect and store data.
- The thermistor will interface with DataTurbine via a source.
- The investigator will monitor and visualize using the NEES Real-time Data Viewer (RDV).


Temp. Source
localhost
DataTurbine (rbnb.jar)
localhost
RDV (Sink)
localhost

Scenario 2: Field Station

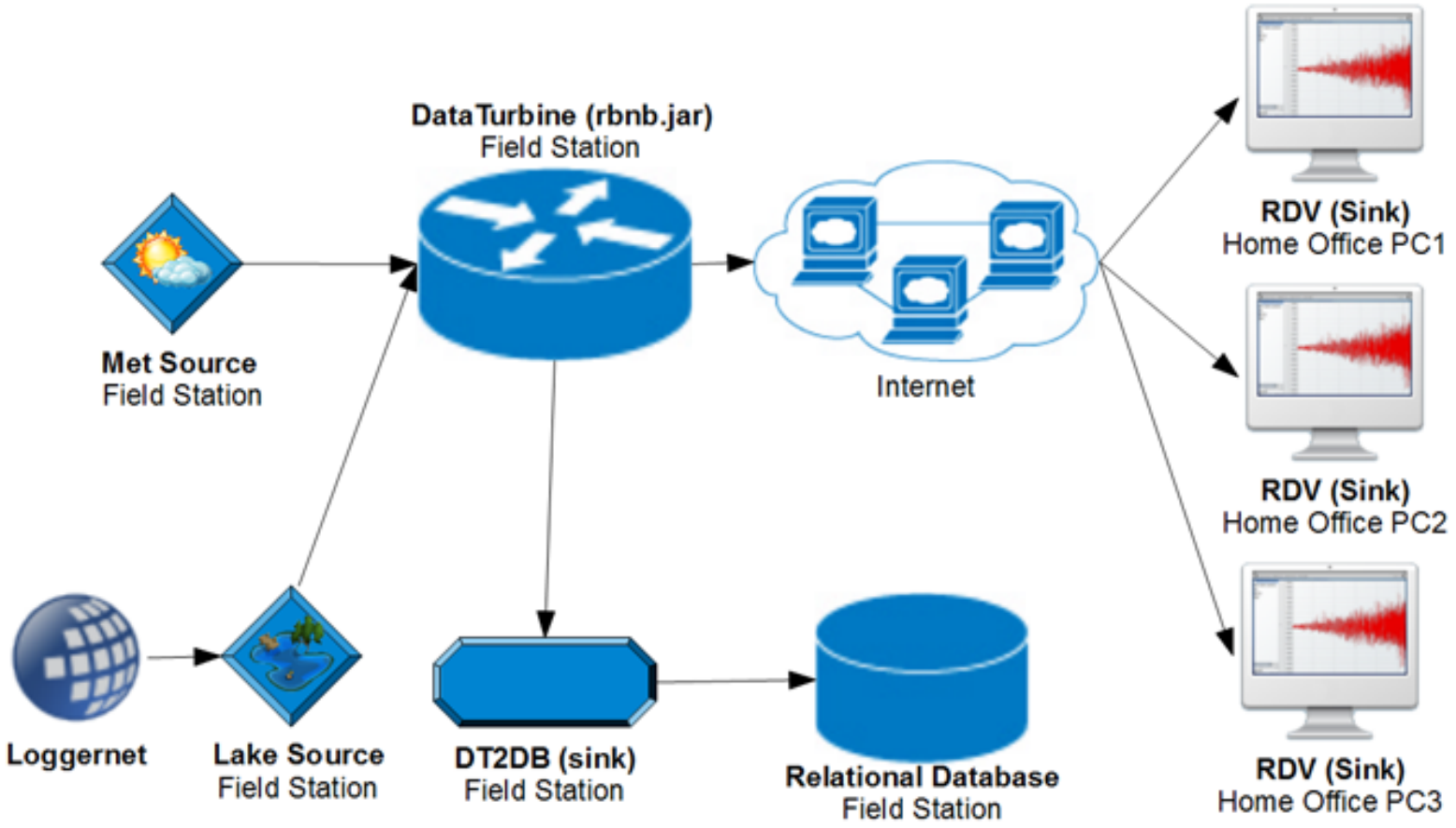
At a field station investigators want to gather data about the lake.

They are interested in dissolved oxygen and temperature from sensors deployed in the water and meteorological data from a weather station at the site.

This is a long term study and the data needs to be archived in a database. Scientists back at the head office want access to the data as it is collected.

Scenario 2: The Infrastructure

- The sensors in the lake are deployed using serial connections that run back to the base station.
- They are using a Campbell data logger and Loggernet to communicate with the lake sensors.
- The data needs to be permanently archived and stored into a database
- Multiple users will access the most recent data from multiple locations across the world


DataTurbine (rbnb.jar)
Field Station
Met Source
Field Station
Internet
RDV (Sink)
Home Office PC1
RDV (Sink)
Home Office PC2
RDV (Sink)
Home Office PC3
Loggernet
Lake Source
Field Station
DT2DB (sink)
Field Station
Relational Database
Field Station

Scenario 3: Build on Existing System

A field site already has an existing archival system.

The site wishes to add on DataTurbine to provide real-time capabilities.

They want to expose the data to the public and other field sites.

Scenario 3: Infrastructure

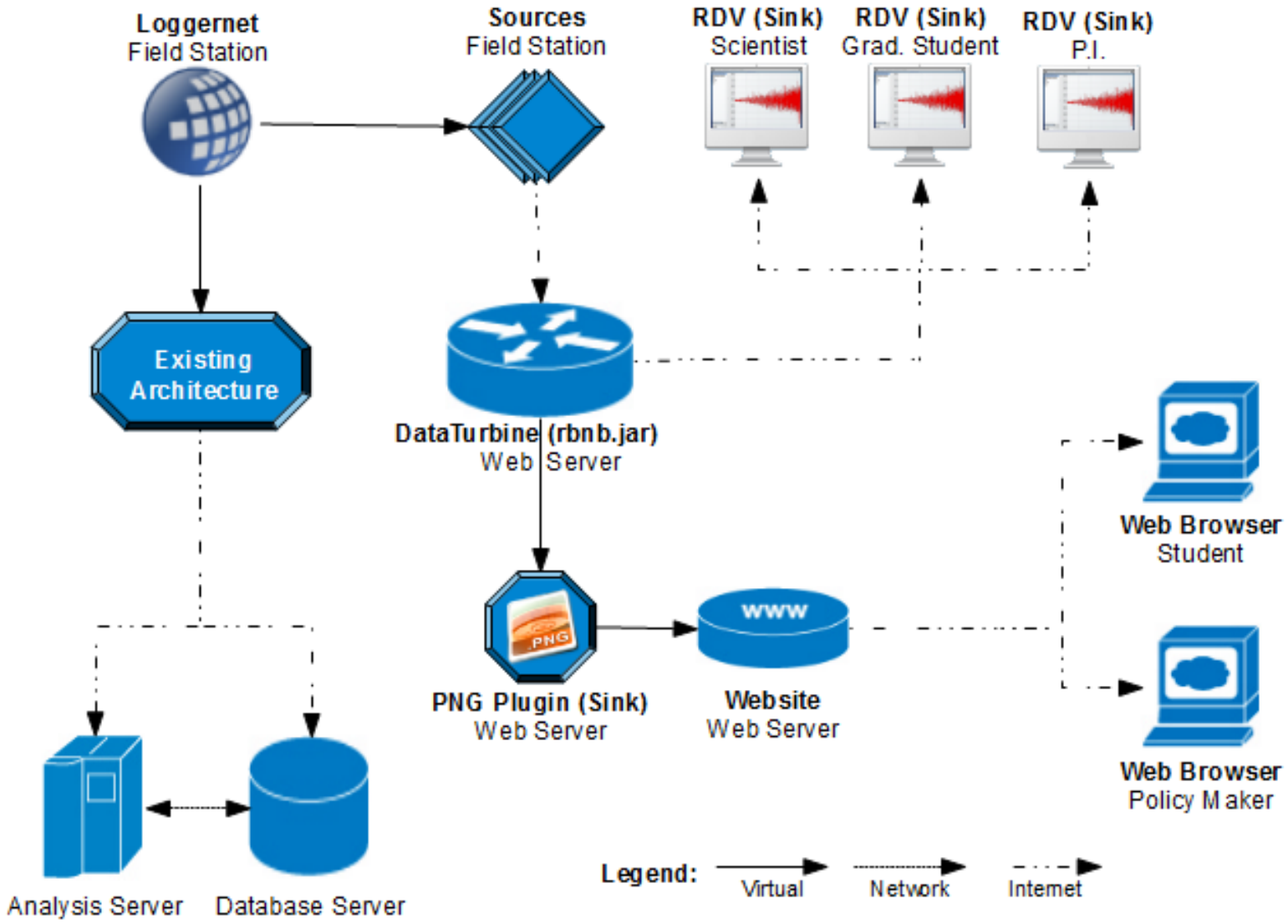
- The field station uses a Campbell data logger and Loggernet to collect data and a custom architecture to put that data into a database, perform QA/QC and analysis.
- Will use a DataTurbine in parallel with the existing architecture.
- This will be accomplished by running multiple sources on the field station machine to load the data.

Scenario 3: Infrastructure

- DataTurbine will expose the real-time streams and provide a service to continuously generate PNG charts to include on a website. DataTurbine will not be responsible for permanent storage or analysis.

- Some clients will connect to the real-time data using the NEES Real-time Data Viewer (RDV). Others through a public web site.

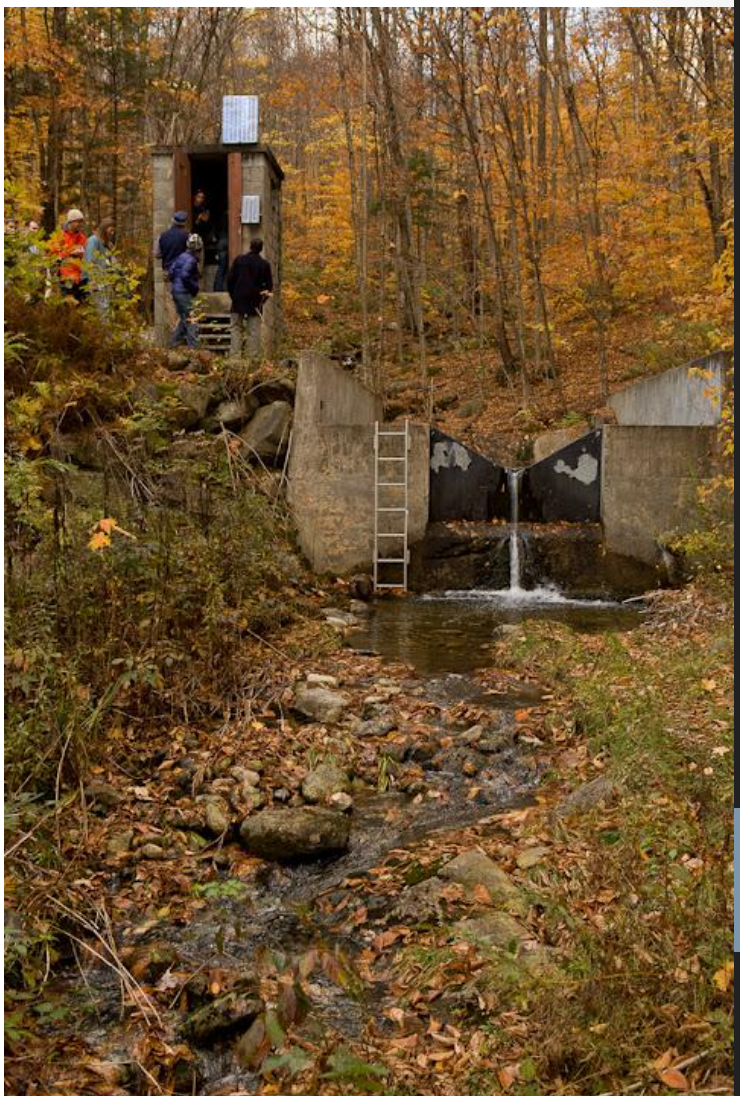
- To reduce load the DataTurbine server and the web server will run on a separate machine.


Loggernet
Field Station
Sources
Field Station
RDV (Sink)
Scientist
RDV (Sink)
Grad. Student
RDV (Sink)
P.I.
Existing
Architecture
DataTurbine (rbnb.jar)
Web Server
Web Browser
Student
PNG
PNG Plugin (Sink)
Web Server
www
Website
Web Server
Web Browser
Policy Maker
Analysis Server
Database Server
Legend:
Virtual
Network
Internet

Scenario 4: Multiple Sites

An institution with multiple field sites wants to use DataTurbine to reliably buffer and stream data to a central database.

Several partner institutions are interested in accessing this data.

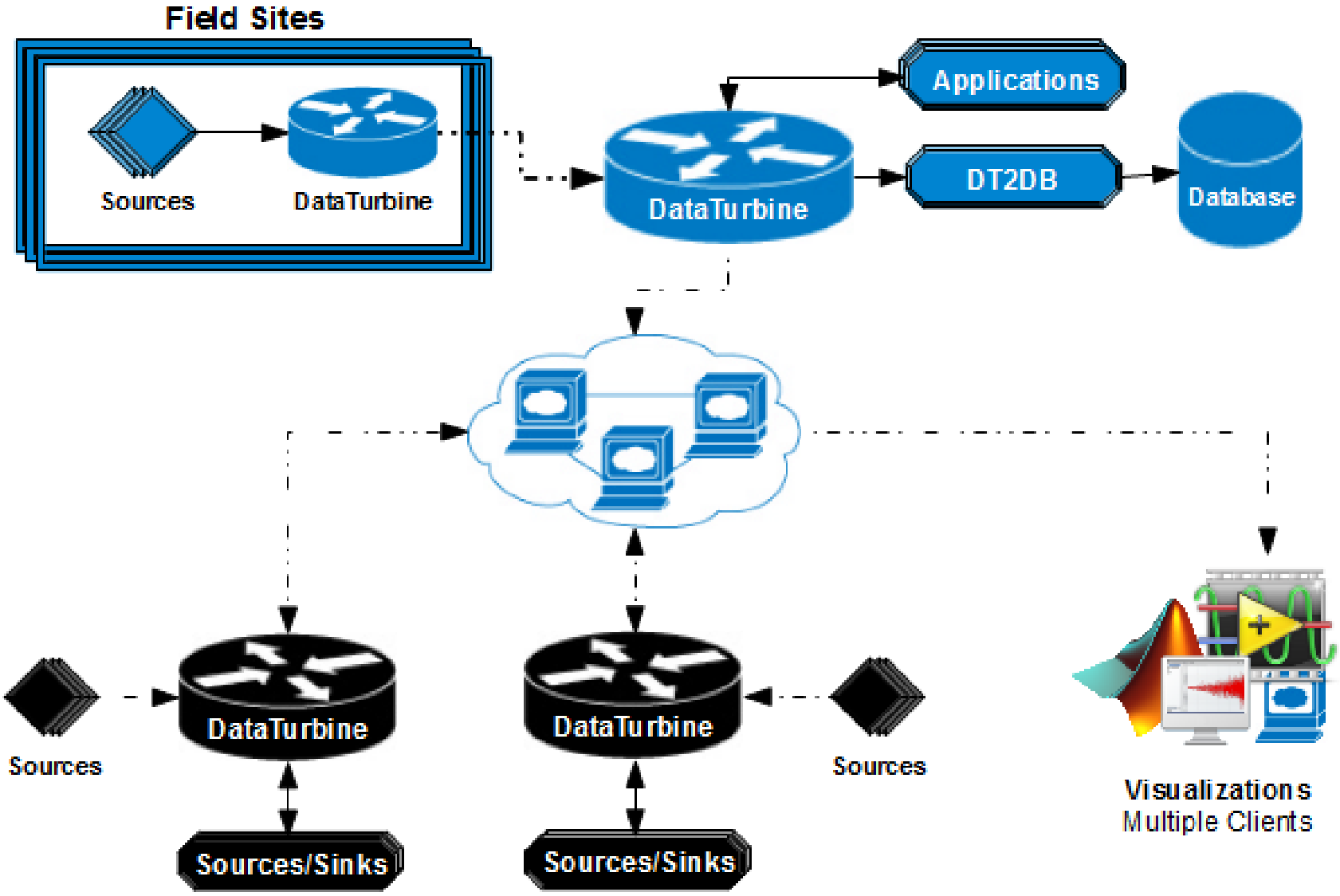


Scenario 4: Infrastructure

- Multiple field stations, all with internet access run complex instrument packages.
- The link between them and the central server is unreliable and prone to downtime.
- Each site runs its own DataTurbine Server to buffer the data and mirrors that data to a central server also running Data Turbine.
- That server runs a suite of analysis applications and stores the data in a database.

Scenario 4: Infrastructure

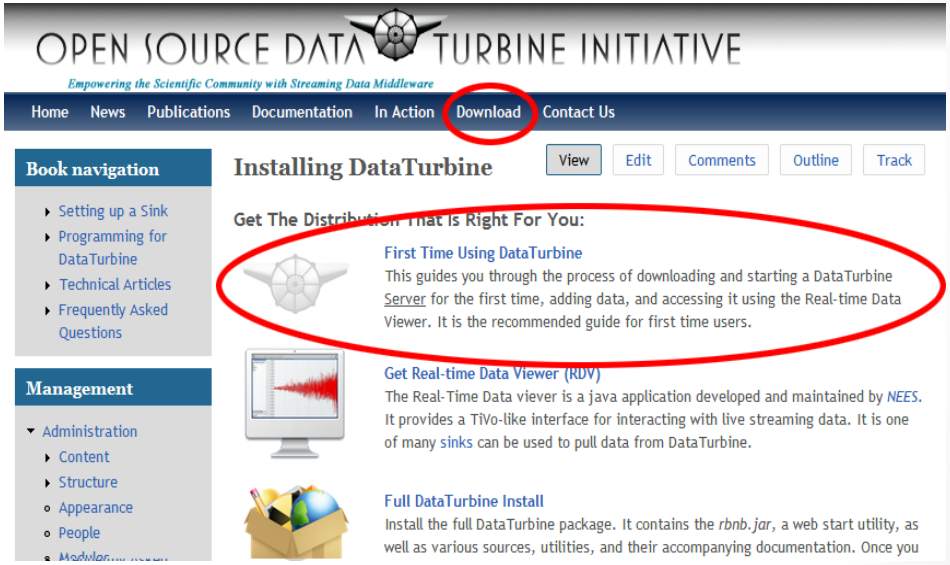
- Several partner institutions also utilize this data.
- They run a multitude of sources to get data and use analysis routines that utilizes information from multiple institutions and sites.
- They store the data and the result of the analysis in their own database.
- Each institution has users that visualize the data via MatLab, LabView, Website, RDV, Google Earth, and other applications


Field Sites
Sources
DataTurbine
DataTurbine
Applications
DT2DB
Database
DataTurbine
DataTurbine
Sources
Sources
Sources/Sinks
Sources/Sinks
Visualizations
Multiple Clients
Partner Institution 1
Partner Institution 2

GETTING STARTED

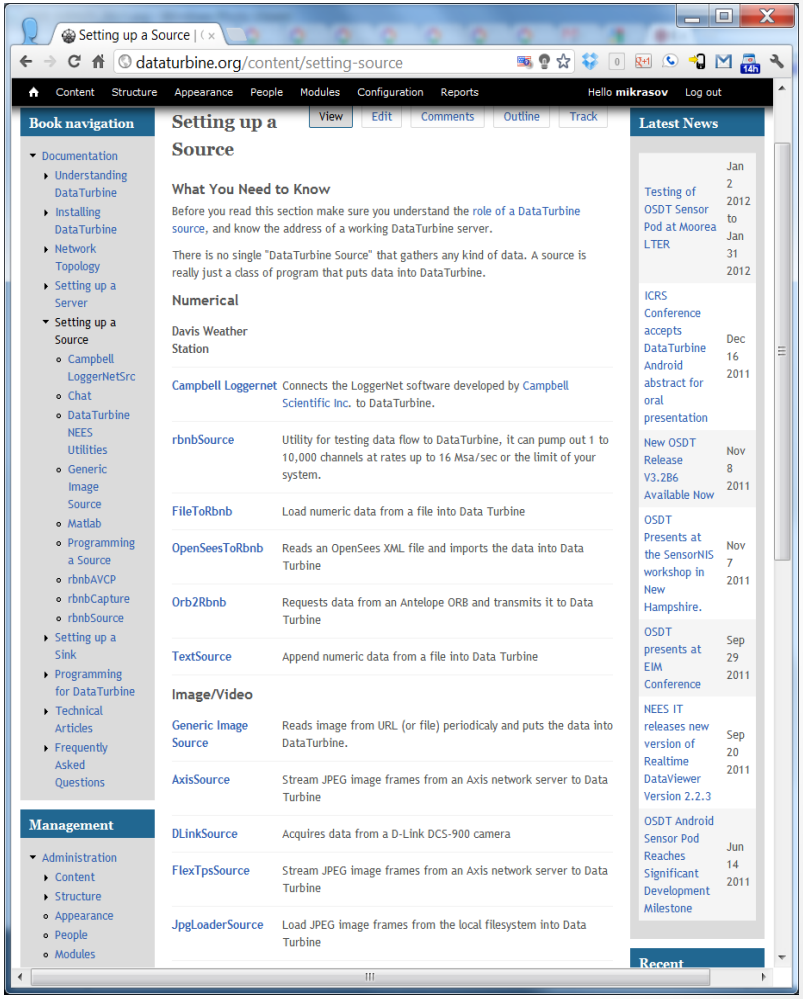
How to get Started

1. Go to dataturbine.org
2. Click the *Download* Tab
3. Click *First Time Using DataTurbine*



Getting a Sink/Source

- On the website there is a list of commonly used sources and sinks



Create Your Own

- Online documentation and sample code
- Can program in Java or Matlab