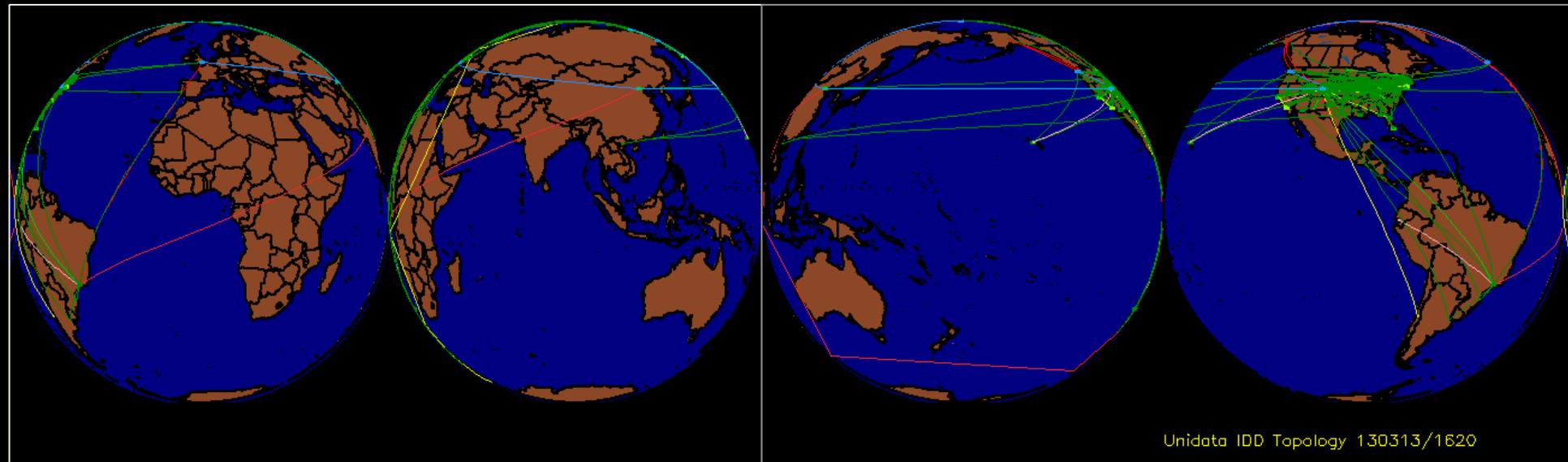


# Sensor Data Stream Aggregation System

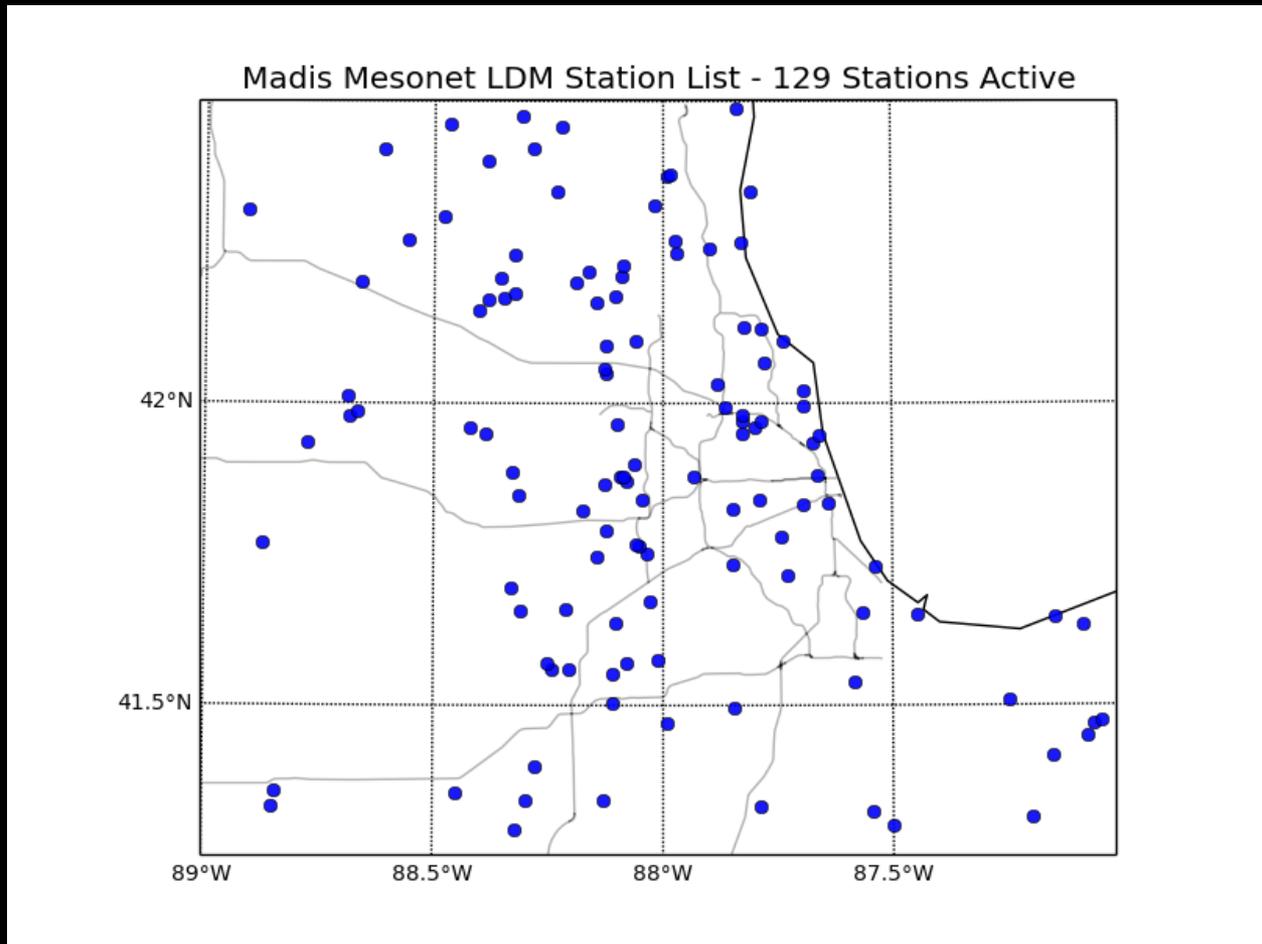
Constantly monitoring multiple streams, and event driven feeds to provide a unified parallel friendly data archive service.

James Bittner, University of Illinois



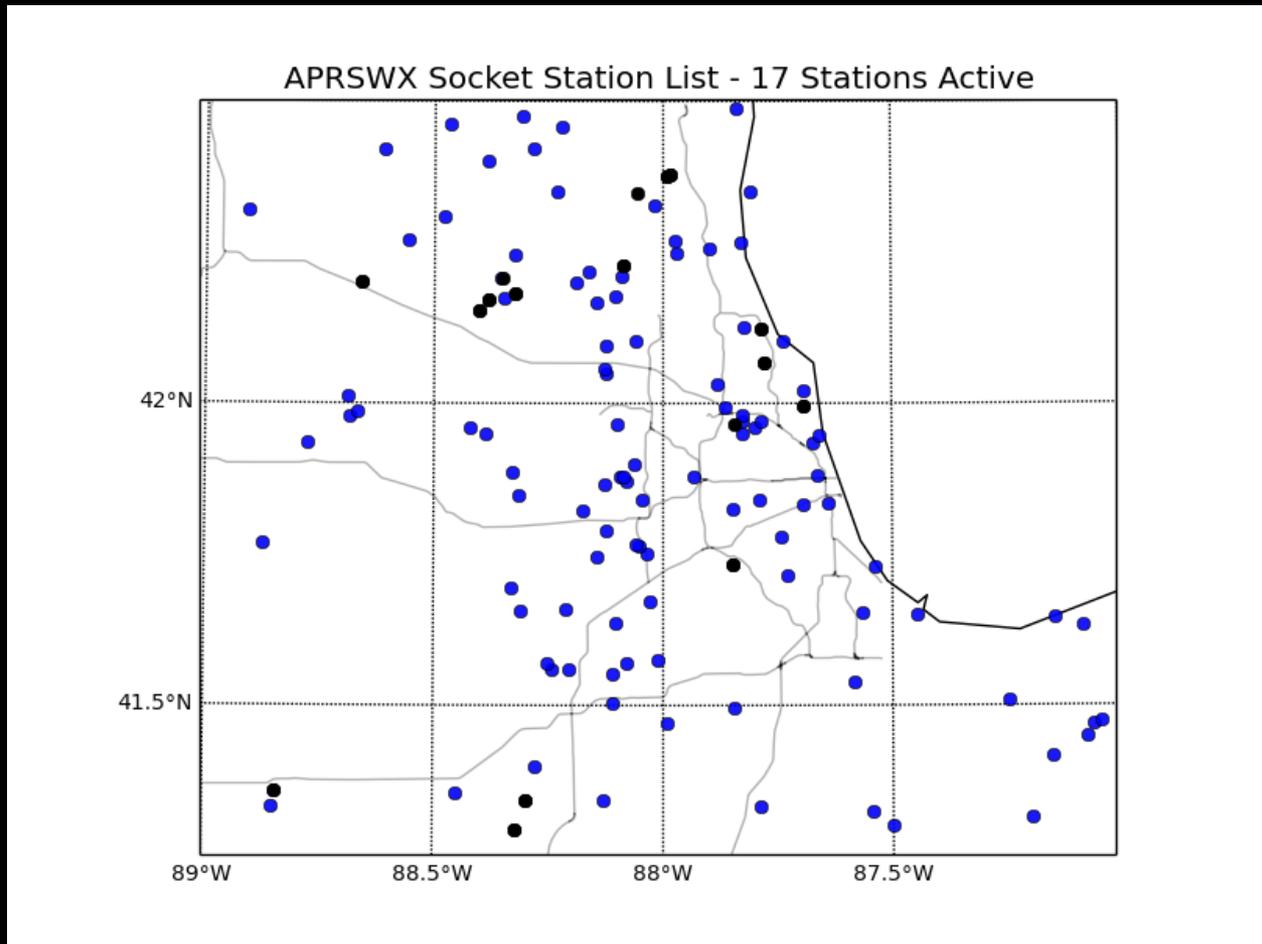
# NOAA MADIS Surface Observations

## LDM



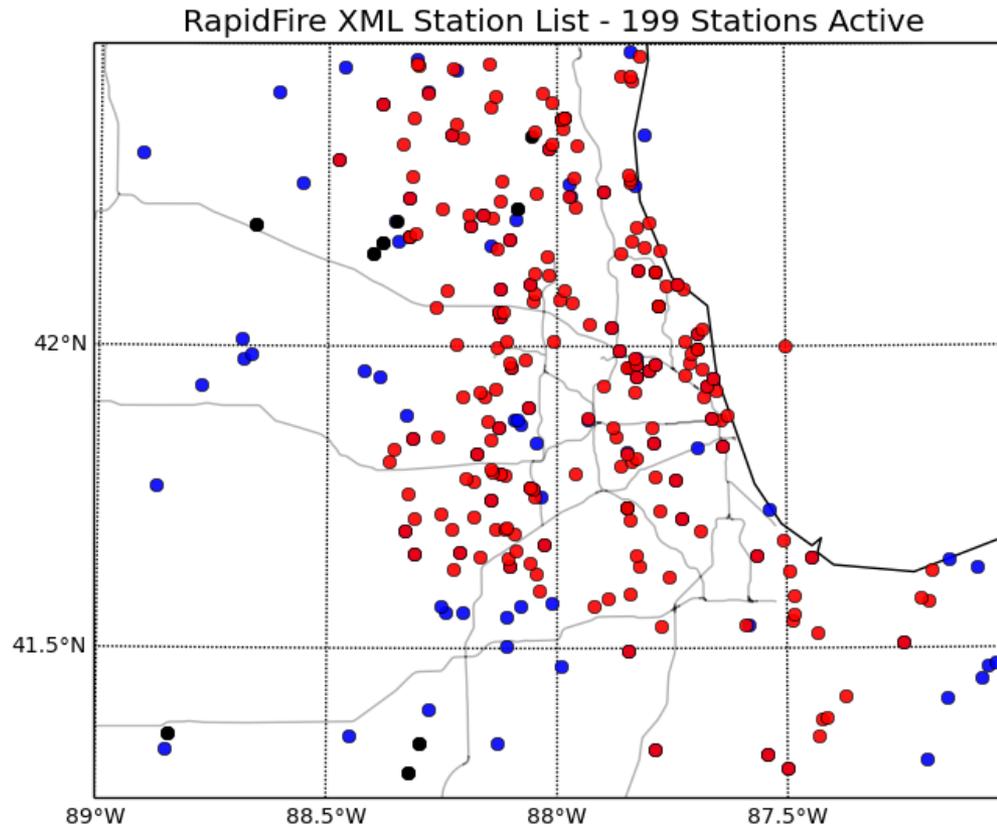
Delay ~10-15 Minutes

# Amateur Radio Packet APRS Socket



Delay ~1 Minutes

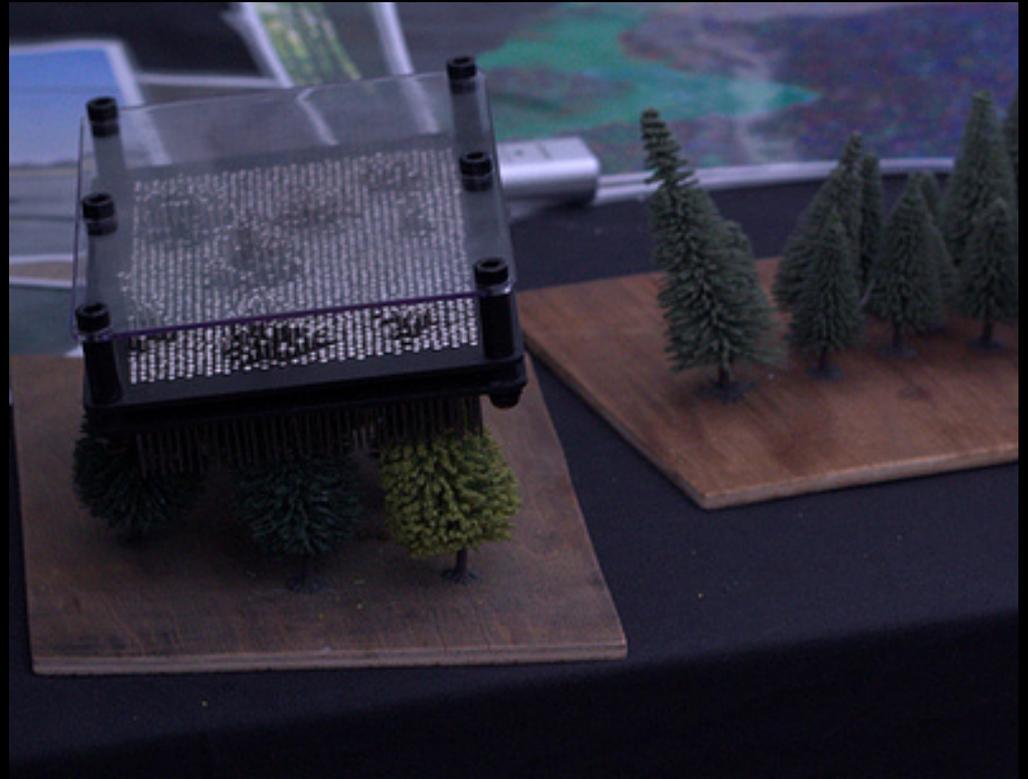
# WU Rapid Fire XML



Delay < 1.0 Minutes

# Overview

- Background
- Motivation
- System Design
- Metrics
- Use Cases
- Discussion
- Questions

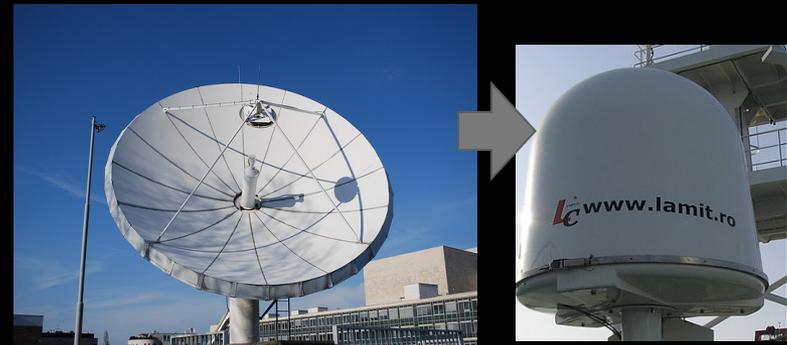


# Background

- Background
  - Communications Platforms
    - GSM/LTE/WIFI/VSAT
- Devices
  - Weather Cameras
  - Low Cost Microcontrollers
- New Window into Processes

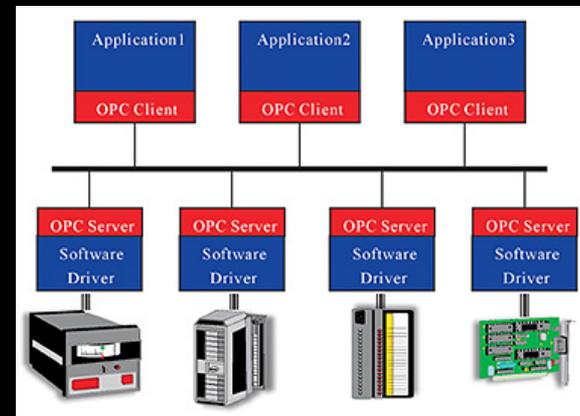
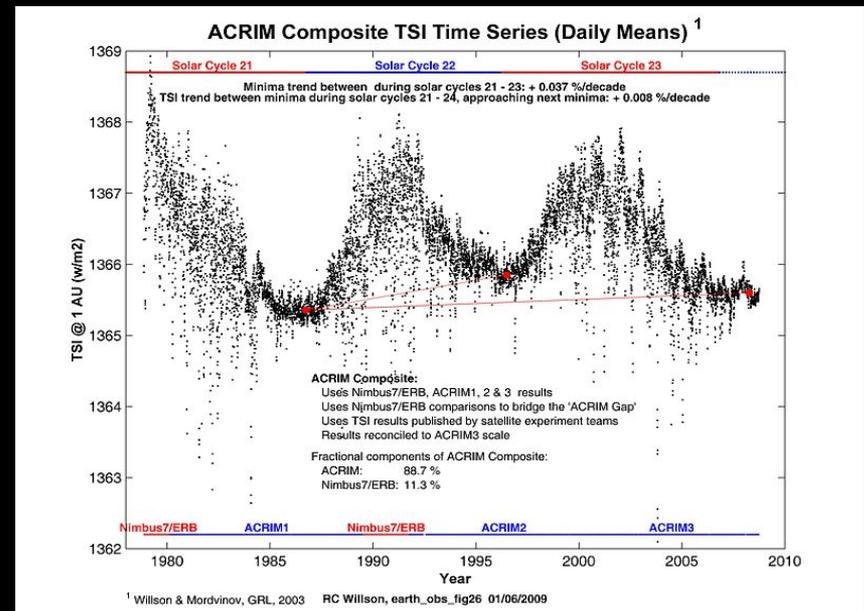


© USGS



# Motivation

- Why?
  - Dataflow Increasing
  - Scalability
  - Context (CSV)
  - Open Standards
- Who Else?
  - OPC, Win32 DCOM
    - Closed Spec.
  - Independent Scripting



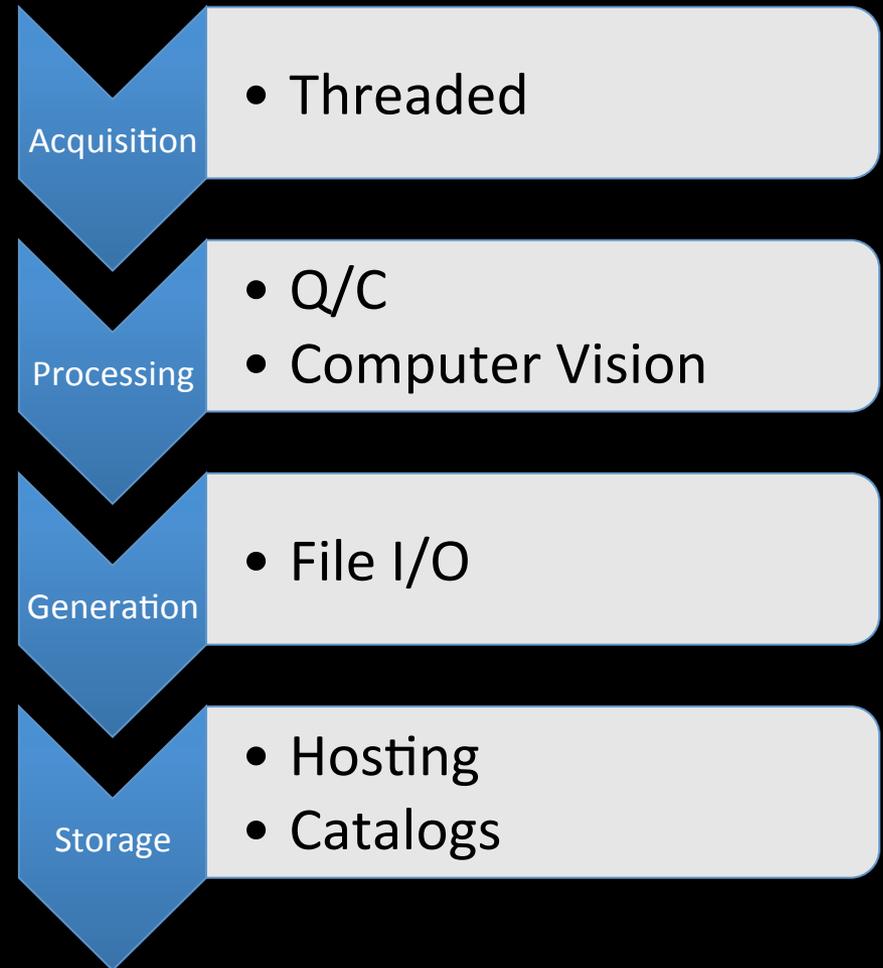
# System Design Goals



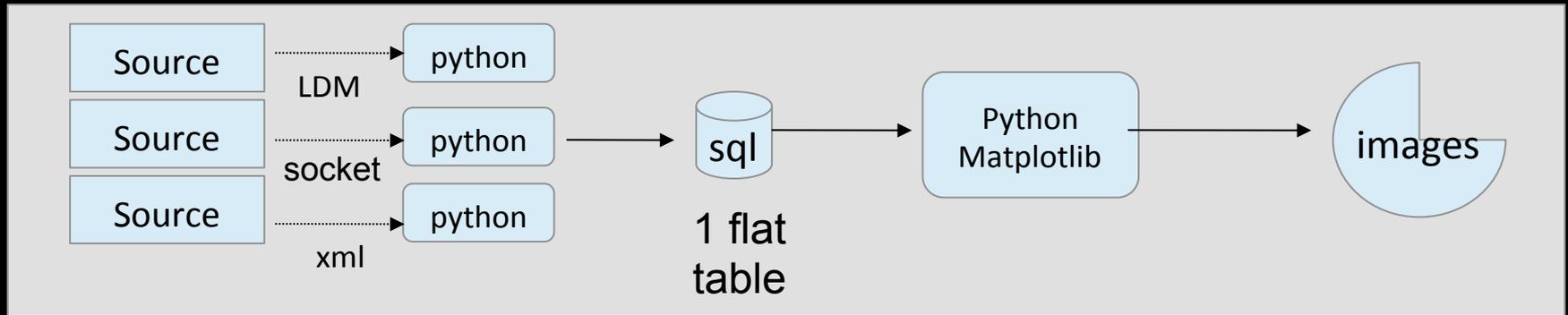
- Support near 'Real-time'
- Open Transparent Process and Procedures
- Wide Support for Product Platforms
- Low Overhead Maintenance
- Long Term Project Scope

# System Layout

- Acquisition
  - Threaded
- Processing
  - Quality Control
  - Computer Vision
- Generation
  - File I/O
- Storage
  - Hosting and Catalogs



# Prototype 1 Design Map



## Pros

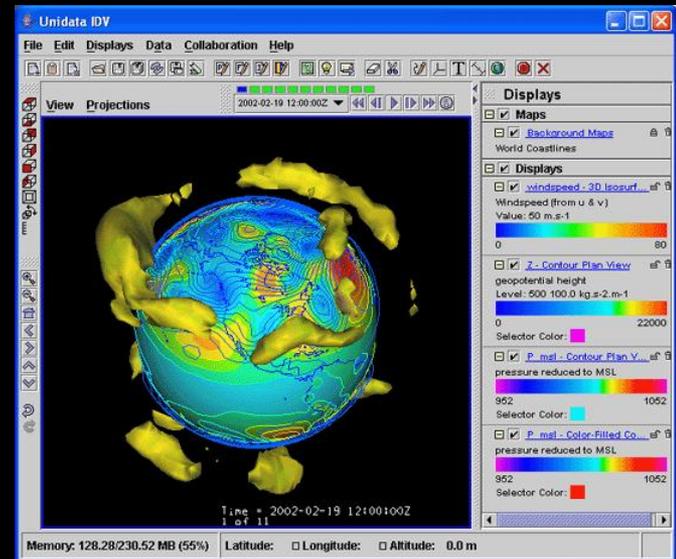
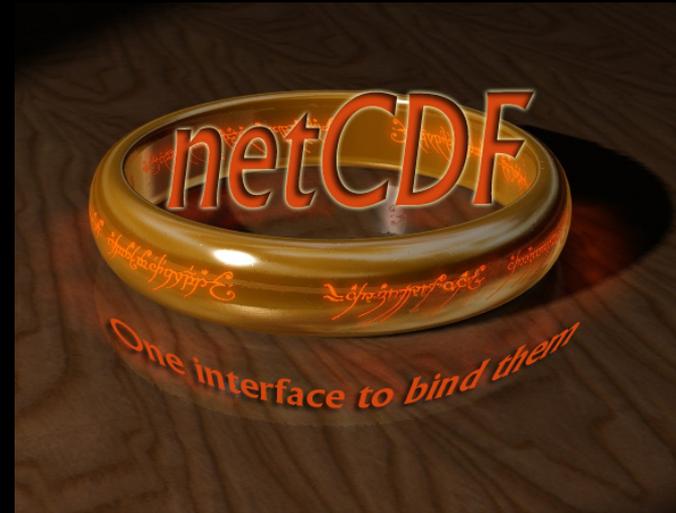
- Data acquisition threaded
- SQL was fast

## Cons

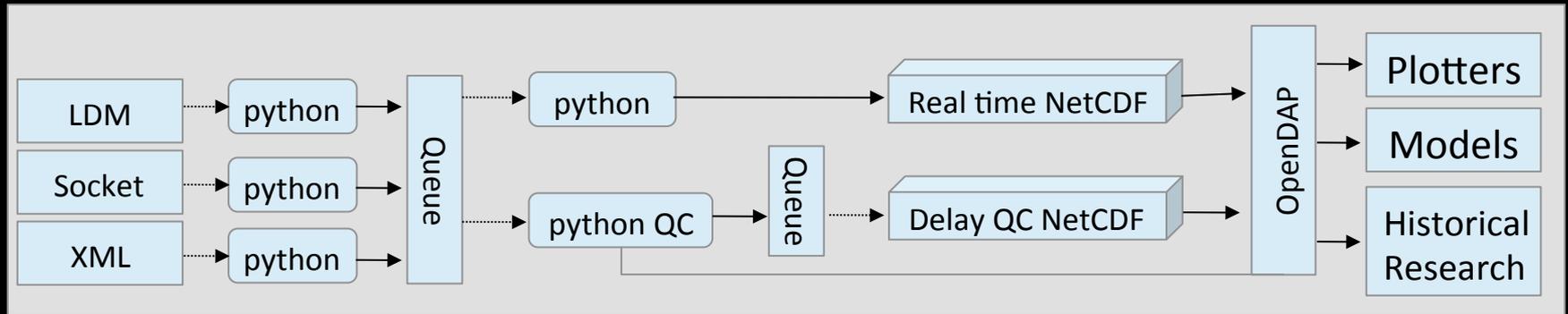
- Threading handled by SQL server
- SQL table fixed size
- Indexing and partitioning is based upon data access
- No clear QC stage
- Direct network SQL access required by users

# NetCDF vs. CSV vs. SQL

- Speed
  - File structured binary
- Low Overhead
  - File based
- Support
  - Most programming languages (C/java/ Fortran/Matlab)
- Existing Architecture for product datasets
  - IDV Java based data explorer
- Self Describing
  - Supporting content of sensors and data
- Parallel solutions exist



# Prototype 2 Design Map



## Pros

- Open standard data file
- Opportunity to parallel at each and every stage (QC, NetCDF Creation, OpenDAP, Plotters)
- Open design allows operation in resource limited environments (queue→HD)
- Network access not required

## Cons

- OpenDAP does not support contextual queries by default on NetCDF Files
- Speed of python

# Source Example

- Event or Timer Based Loading Resource
- Parse resource based on standards (Metar, NetCDF, APRS)
- Map source sensor names to global sensor names and units
- Create observation array for each sensor unit
- Serialize and send to Queue Broker

```
In [11]: obs
Out[11]:
{'addtime': 123156859,
 'dsource': 1,
 'elev': 486,
 'key': 'Temp',
 'lat': 41.38,
 'long': -88.3,
 'stn': 'KORD',
 'timestamp': 123156852,
 'value': 292.15}
```

# Quality Control

- Event Driven on available packet present with broker, message passed as callback
- Last 5 Points for Station-Sensor Loaded
- New Reading Valid Range Checked – Q1
- New Reading Temporal Rate Checked – Q2
- Potential for Nearest Neighbor.. Etc
- Q1, Q2 Tags added to array
- Returned to Queue broker as Final Message

# Write to NetCDF

- Two version exist: 'Real-Time' and Quality Controlled
- Receive callback from broker with array
- Open/Create Hourly NetCDF File
- Add/Update Station Spatial Dataset
- Add Observation
  - Create sensor if not existent
- Sync NetCDF File with disk

# System Web Panels

- Queue System Front End

RabbitMQ™ User: guest

Overview Connections Channels Exchanges **Queues** Users Virtual Hosts

## Queues

▼ All queues

Overview				Messages			Message rates		
Name	Exclusive	Parameters	Status	Ready	Unacked	Total	Incoming	deliver / get	ack
Final		D	Active	0	946	946	7/s	11/s	70/s
ToBeQc		D	Active	0	0	0		7/s	7/s
rtqueue		D	Active	0	609	609		11/s	84/s

▼ Add a new queue

- OpenDAP Web Portal

ldm1.mcs.anl.gov:8080/opendap/

OPeNDAP

## Contents of /

Name	Last Modified	Size	DAP Response Links	Webstart
<a href="#">LDAD/</a>	2013-07-22T17:48:56	-	- - - - -	

# OpenDAP Web Portal

## OPeNDAP Server Dataset Access Form

**Action:**

Get ASCII

Get as NetCDF

Binary (DAP) Object

Show Help

**Data URL:**

**Global Attributes:**

```
NC_GLOBAL.title: Aggregation of Data
NC_GLOBAL.summary: IL Region
NC_GLOBAL.keywords: point
NC_GLOBAL.Conventions: CF-1.6
NC_GLOBAL.standard_name_vocabulary: CF-1.6
NC_GLOBAL.description: Aggreqation of Data
```

**Variables:**

**lon:** Array of 32 bit Reals [station = 0..699]

station:

```
least_significant_digit: Attribute edlided: Unsupported attribute
type (NC_INT64)
standard_name: longitude
long_name: station longitude
units: degrees_east
```

**lat:** Array of 32 bit Reals [station = 0..699]

station:

```
least_significant_digit: Attribute edlided: Unsupported attribute
type (NC_INT64)
standard_name: latitude
long_name: station latitude
units: degrees_north
```

# Data Access Methods

- Python : NetCDF4
- C++/C: LibNetCDF
- Fortran: NetCDF Fortran
- ArcGIS: EDC Module
- Web: <http://ldm1.mcs.anl.gov:8080/opendap>

Input: 

```
import netCDF4
ds = netCDF4.Dataset('http://ldm1.mcs.anl.gov:8080/opendap/aggregate/qualitycontrol/anl-qc-2013-08-09-1300.nc')
print "\n".join(ds.variables.keys()[1:5])
print ds.variables['station_name'][1]
```

Output: 

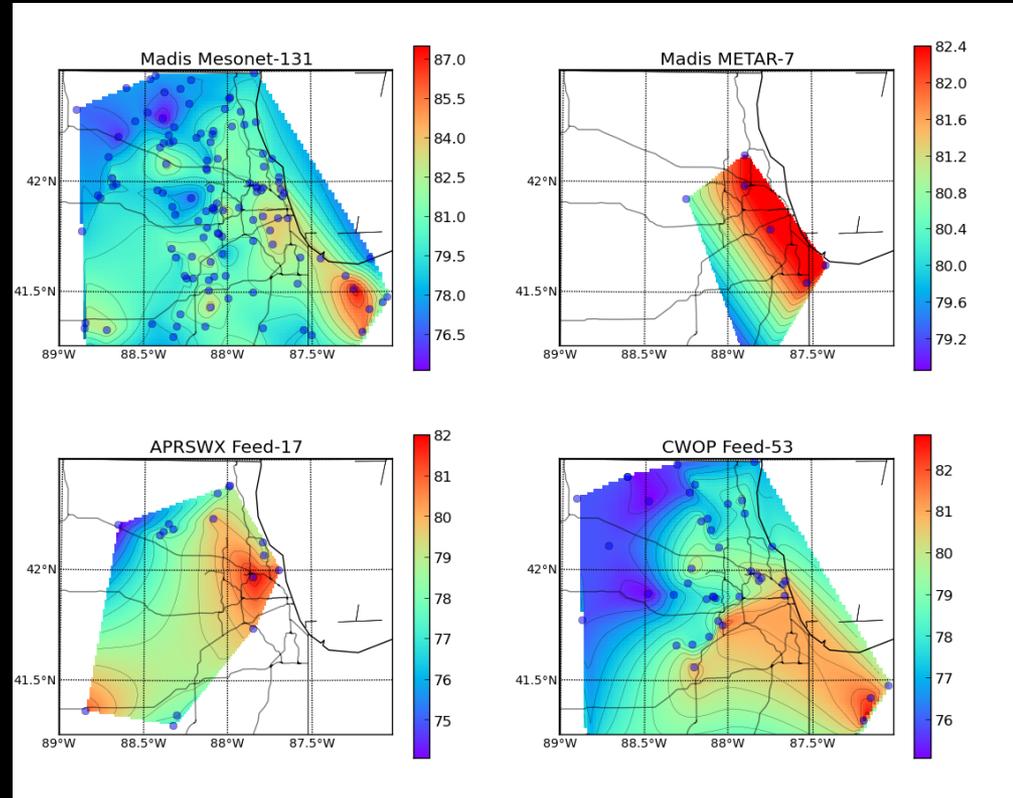
```
lat
alt
station_name
station_info
['A' 'S' '9' '7' '7' .....
.....
.....]
```

# Use Cases:

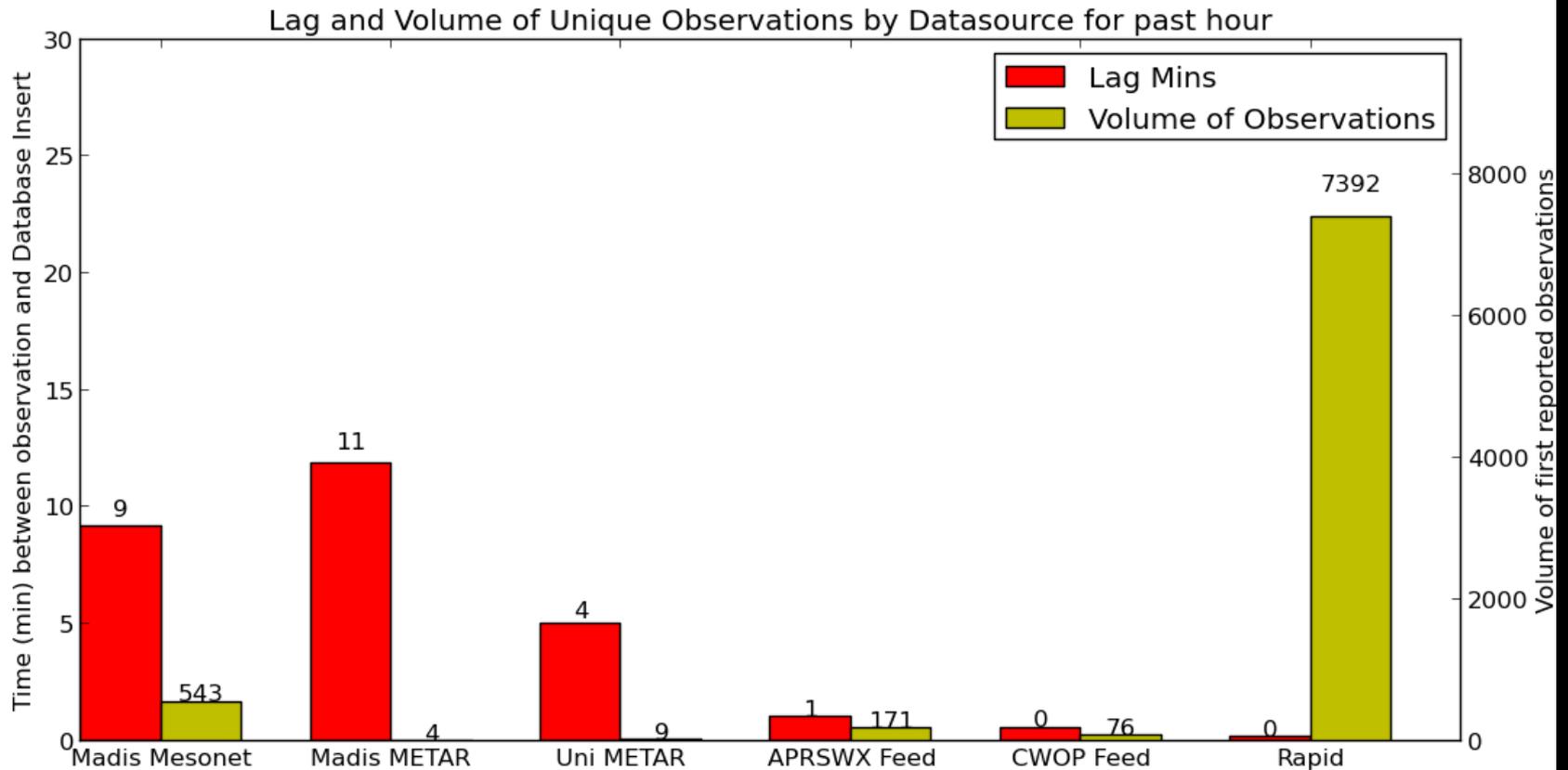
- Weather Modeling
  - Increased observation density/frequency
- Event Based Systems
  - Continuous bridge monitoring
- Production Optimization
  - Integration with near real-time plant measurements with cloud fueled models to generate operational decisions

# Use Case Example:

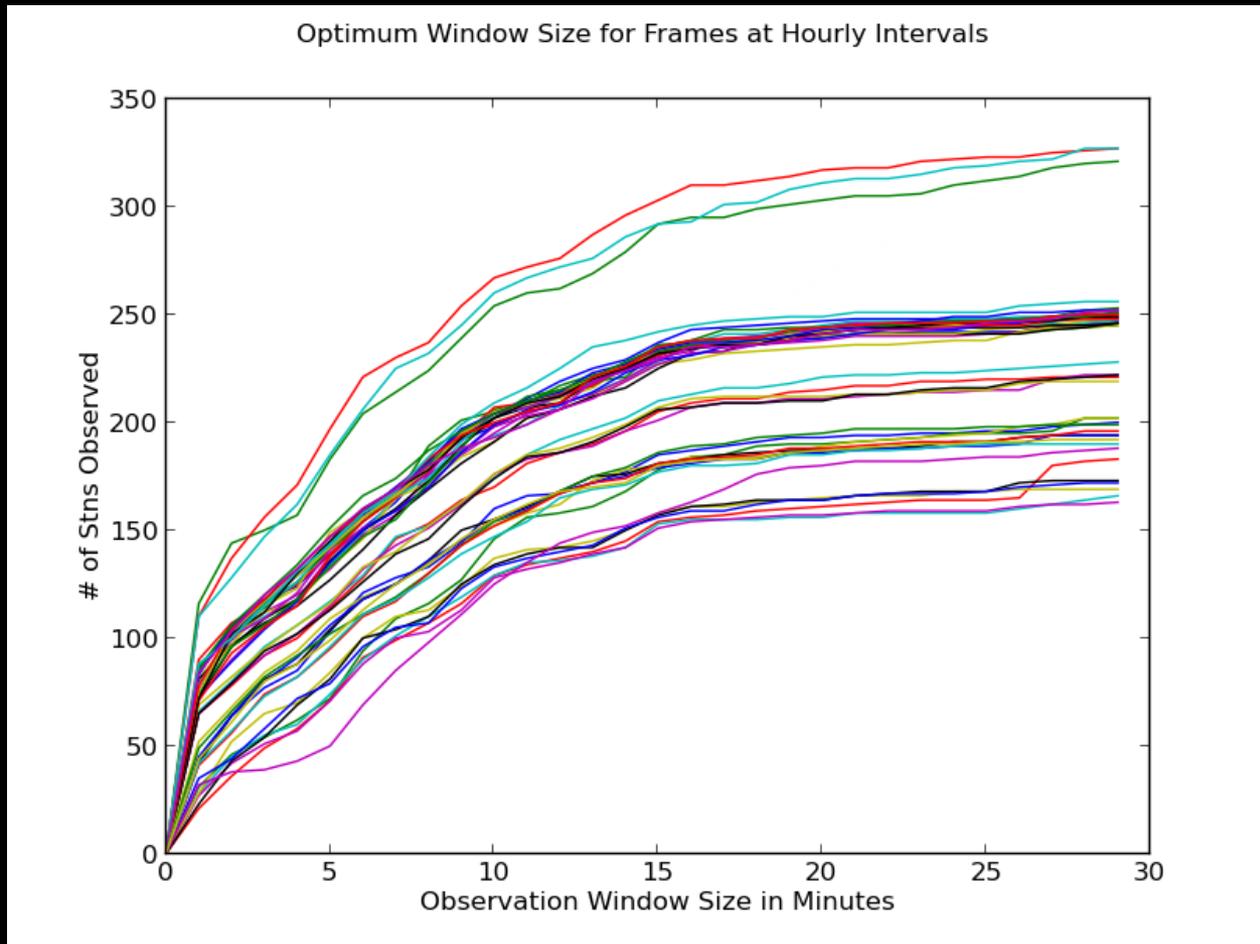
- Weather Modeling
  - Multiple Integration
    - LDM
    - Socket
    - XML
  - Quality Control
    - Valid Range
    - Temporal
  - Products



# Use Case Example Products

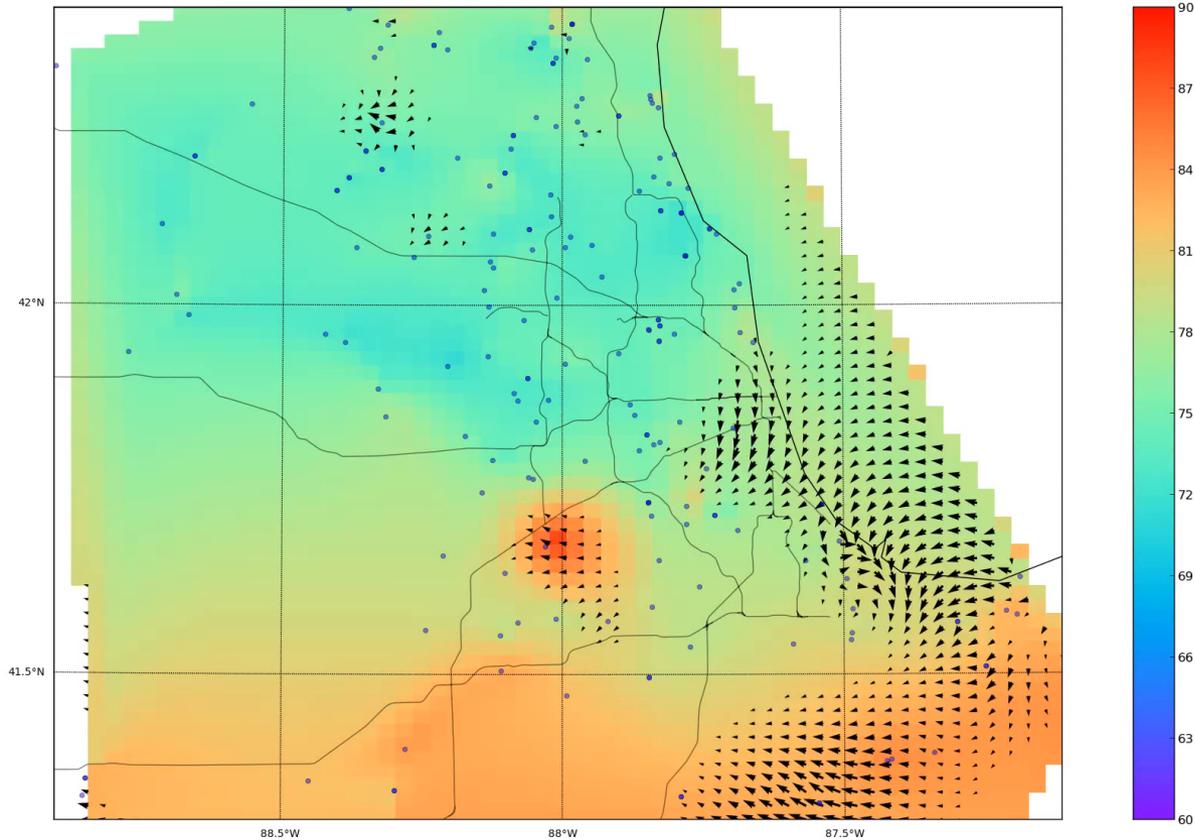


# Use Case Example Product

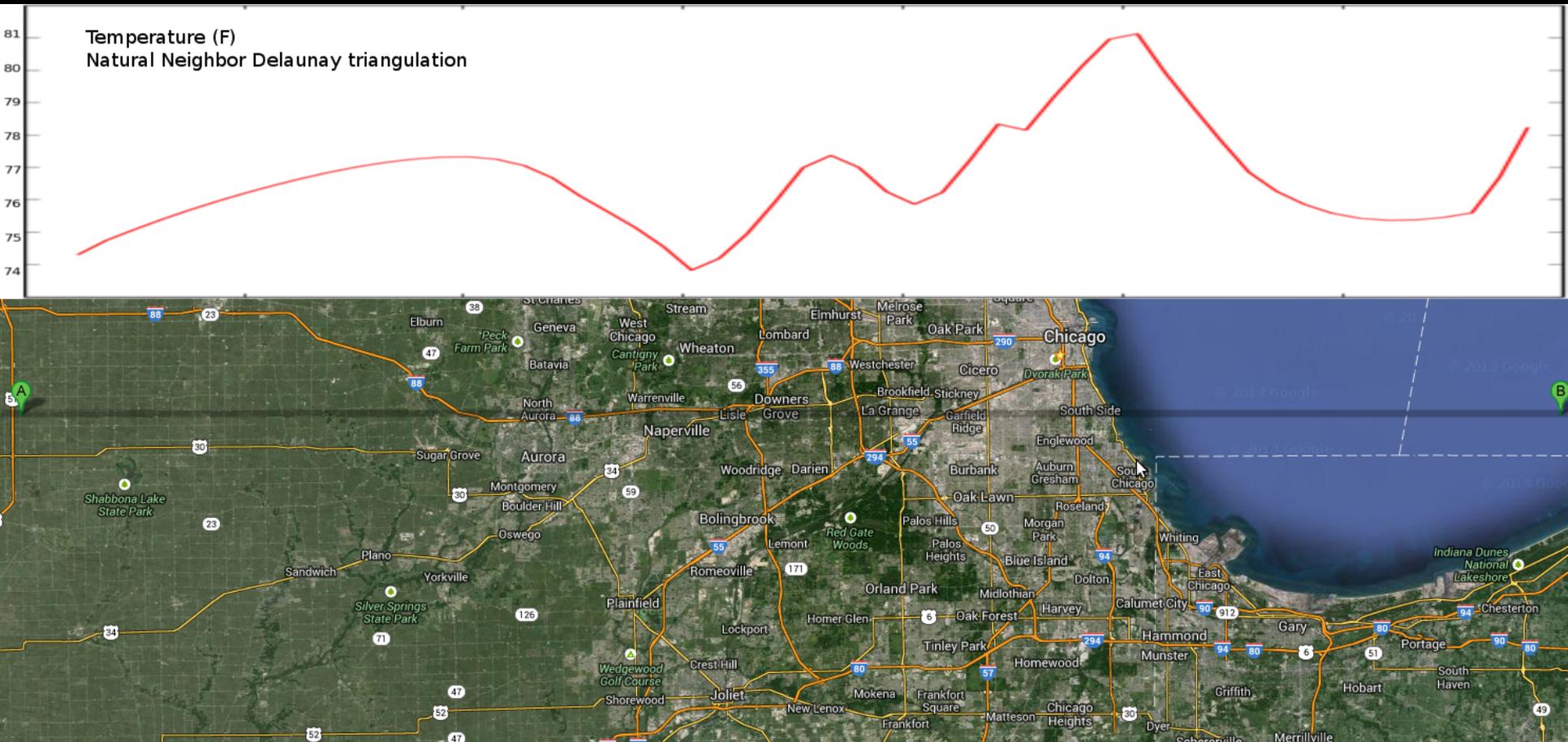


# Use Case Example Product

Temp Contours by Data Source: Start:Sat Jul 20 00:05:00 2013 End:Fri Jul 19 23:55:00 2013

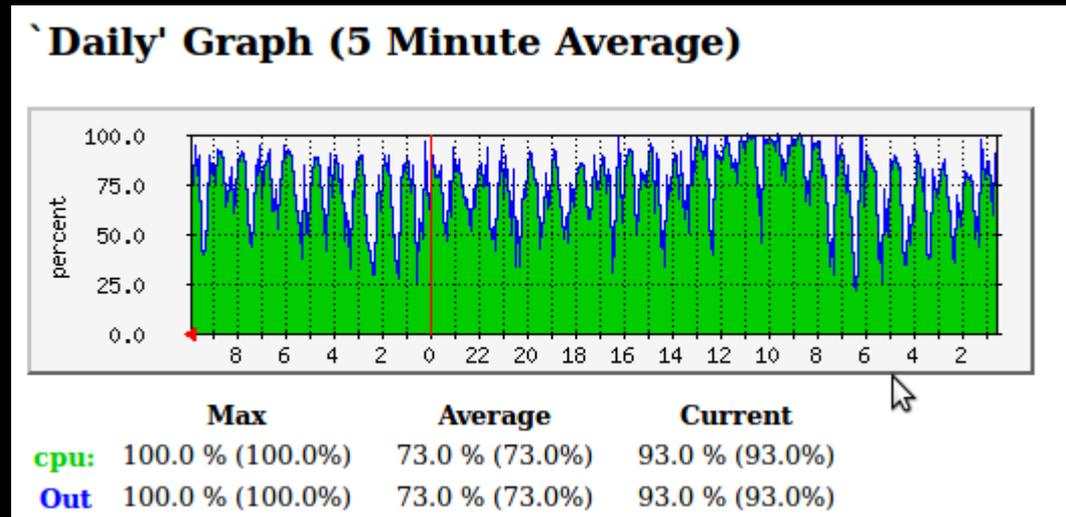


# Use Case Example Product



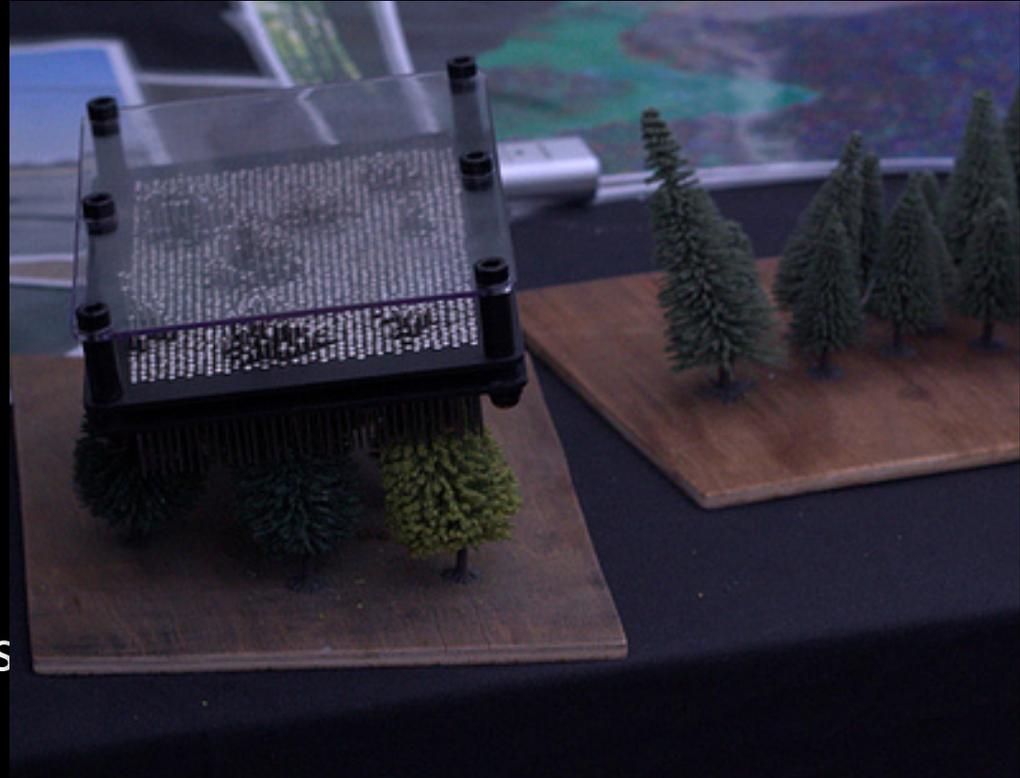
# Lessons Learned

- Python Excellent for acquisition phase
  - Lots of module support
- Throughput
  - Intel Core2 Duo E8400 @ 3Ghz
  - 160 Obs/second
- Bottlenecks



# Conclusion

- Platform Results
  - Modular acquisition
  - Open Formats and architecture
  - Potential for massive deployment
  - Low maintenance and shared data
- Use Case Results
  - Highlighted bottlenecks
  - Generated interesting products



# Acknowledgements

- Rajesh Sankaran, Kate Keahey, Rao Kotamarthi
- Scott Collins, Jonathan Helmus
- The Forest Team
  - Pete Beckman, Nicola Ferrier, Yuki Hamada, Nick Bond, Erik Keever, Jim Adamiec, Xiang Zhu, Gavin Strunk
- Unidata
- Noaa Madis Project
- Weather Underground
- Civilian Weather Observer Program



© COBOL Rube Goldberg by Phil Manker

# Questions

?