

# Phasor Data Networks and Middleware

**Prof. Dave Bakken**

**School of Electrical Engineering and Computer Science**

**Washington State University**

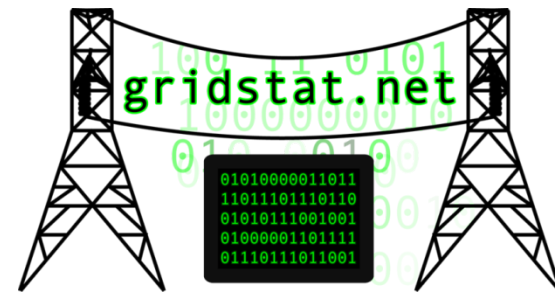
**Pullman, Washington, USA**

**[bakken@eecs.wsu.edu](mailto:bakken@eecs.wsu.edu)**

**NASPI Working Group Meeting**

**Chicago, IL**

**October 24, 2013**



# Background

- I am NOT a power person (IANAPP)! 😊
- Applied computer scientist: distributed computing, fault-tolerant computing, middleware
- BBN 1994-1999
- Working closely with WSU's power researchers since 1999
  
- My wise acre email .sig
  - “Since 1999, cheerfully and audaciously dragging the wide-area data delivery services of the electric power grid – kicking and screaming – into the mid-1990s. ETA: 2015-2020 (for 10% penetration...)”
  - ➔ That may have been optimistic 😞

# Outline

- **Definition of Relevant Terms**
- Middleware
- Common WAMS comms. deployment shortcomings

Following talk by Dan Lutter of Allied Partners dovetails with this very nicely...

# Definition of Some Terms

- **Computer networking**: gets bytes of data from point A to (multi-) point B with some delivery properties
- **Distributed computing**: a discipline above the network layer that ask how can we best use the network to help applications
  - Coordinate
  - Synchronize
  - Replicate
  - Higher-level building blocks for programmers (**middleware**)
- Distributed computing systems are very challenging for programmers. Harder even than concurrent programming, so programmers need all the help they can get!
  - Partial failures
  - Variable network delays

## Definition of Terms (cont.)

- **Point-to-point**: 1→1 communication where an application on one computer sends data to a single other application on another computer
- **Publish-subscribe**: 1→many middleware that lets subscriber applications **selectively** get information from publishers that they need at the data layer and according to subscription criteria
- **IP multicast (IPMC)**: a network level technique for efficiently sending data packets to multiple recipients. NOT pub-sub by definition
  - At packet layer, not data/middleware layer
  - Not selective: everybody gets everything or nothing
  - Note: **have to use carefully**; banned in many cloud datacenters because it can cause address instability and other problems

## Definition of Terms (cont.)

- **QoS**: quality of service. Usually thought of as network-level latency and jitter guarantees but translated up to the application layer (and managed at this level) by middleware
- **QoS+**: all “behavioral” properties: network performance and jitter, fault tolerance/availability, rate, security, etc.

# Closed Loop Wide Area Data Delivery Requirements

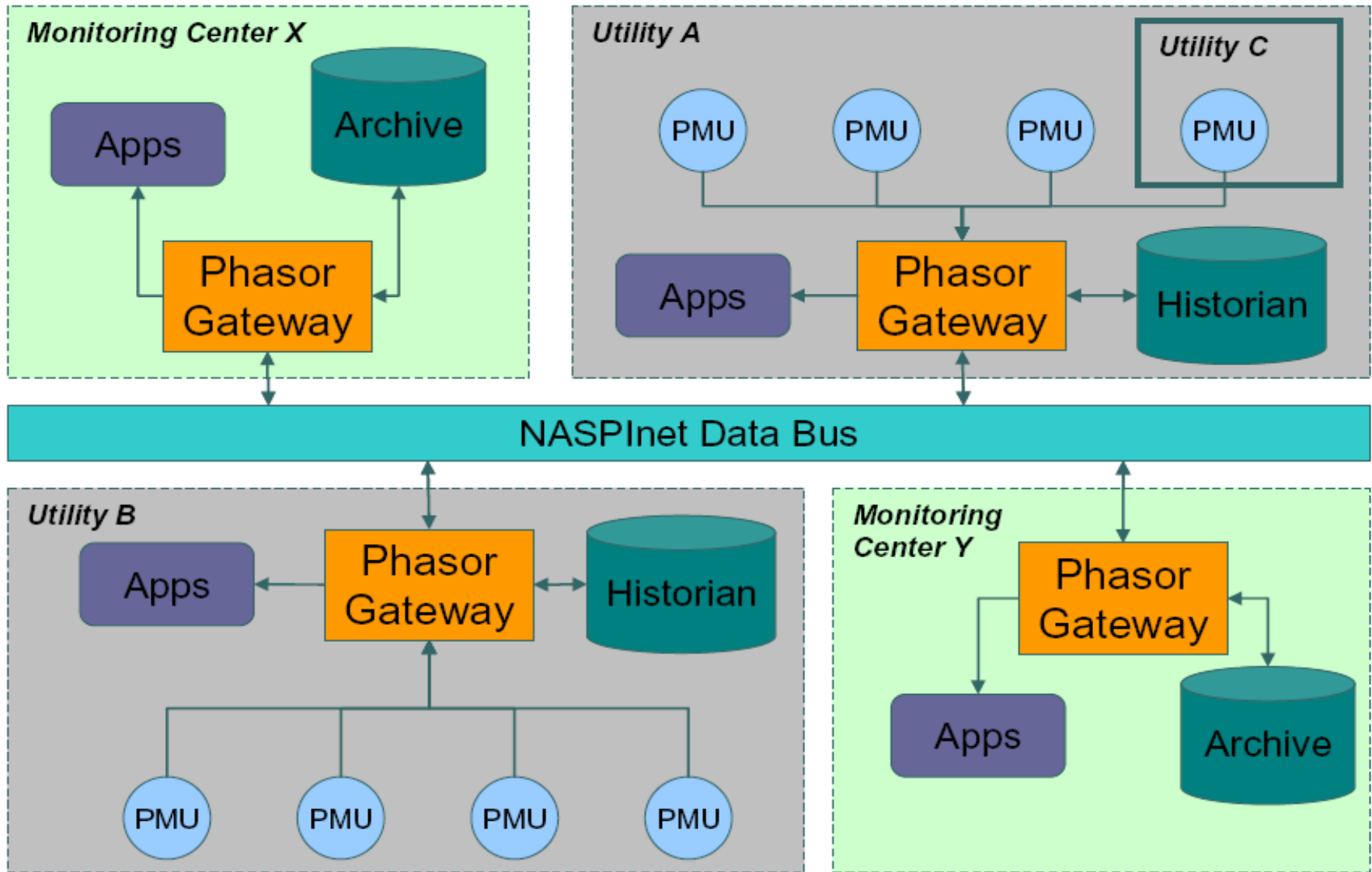
- Hard end-to-end guarantees on latency and jitter
- Future proofing: extend to use new net-level technologies, mechanisms for security or fault tolerance, etc.
- Provide a wide range of QoS+ guarantees
- Support extremely low latency guarantees
  - Predictably and **for each update** (not 30 minute aggregate)
  - Tolerating non-malicious failures
  - Tolerating malicious cyber-attacks
- Complete admission control at perimeters
- Management and instrumentation at app and data levels, not just network level
- Note: no commercial industry or military environment has anywhere these extreme requirements!
- Lots of ways to do the above wrong; see more info slide ...

# Outline

- Definition of Relevant Terms
- **Middleware**
- Common WAMS comms. deployment shortcomings



# NASPInet Conceptual Architecture

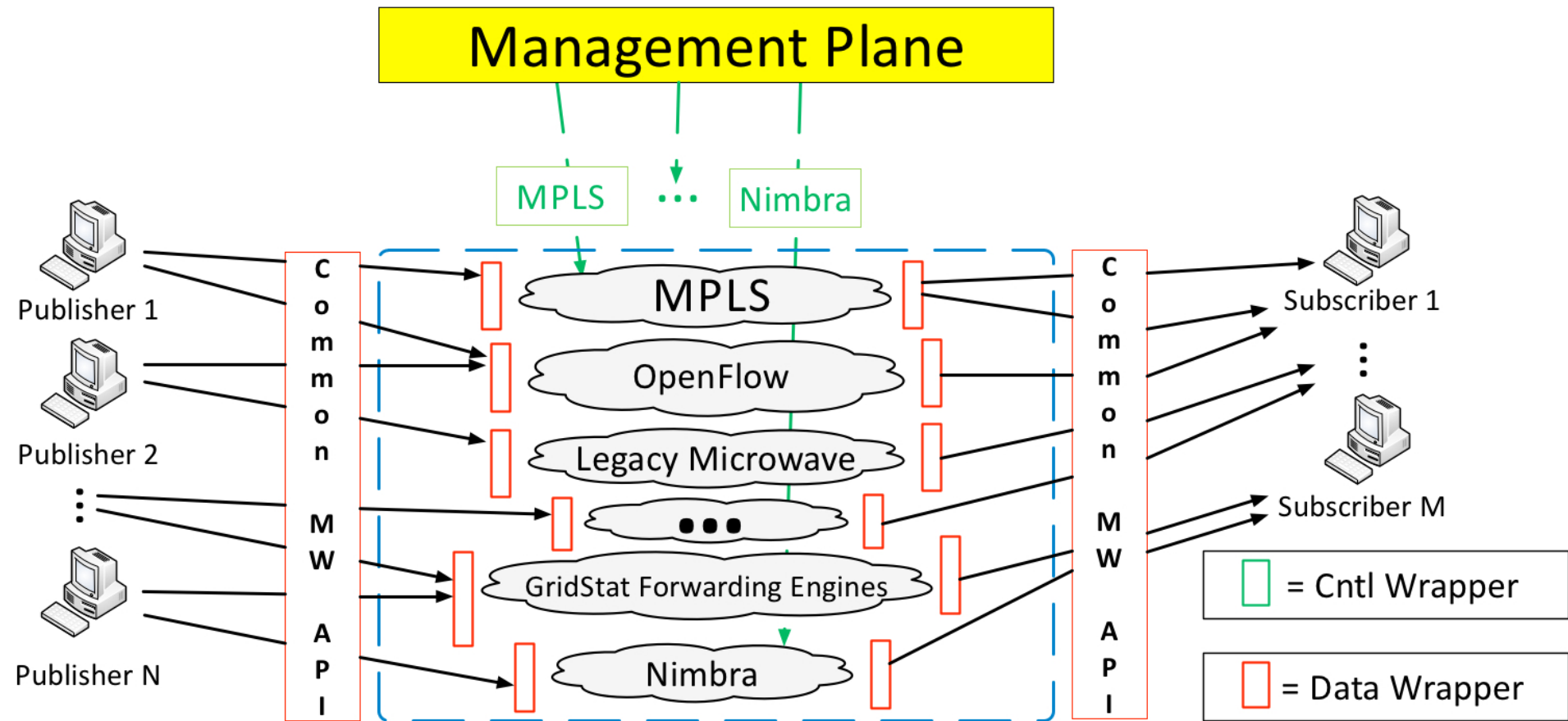


# Middleware in One Slide

- Middleware == “**A layer of software above the operating system but below the application program that provides a common programming abstraction across a distributed system**”
- Middleware exists to help manage the complexity and heterogeneity inherent in distributed systems
- Middleware provides **higher-level building blocks** (“abstractions”) for programmers than the OS provides
  - Can make code much more portable
  - Can make them much more productive
  - Can make the resulting code have fewer errors
  - **Programming analogy — MW:sockets  $\approx$  HOL<sup>1</sup>:assembler**
- Considered best practices in other industries for 15-20 years! (Ouch!)
- See resources at end for why needed for WAMPAC



# Middleware Integrating Legacy (Sub)Systems



- Manage as one coherent whole
- Future proofing (no tech lock-in)
- "... can be BPL/PLC, satellite links, 4G cellular, ...
- Tight admission control and network mgmt. w/data knowledge

# Outline

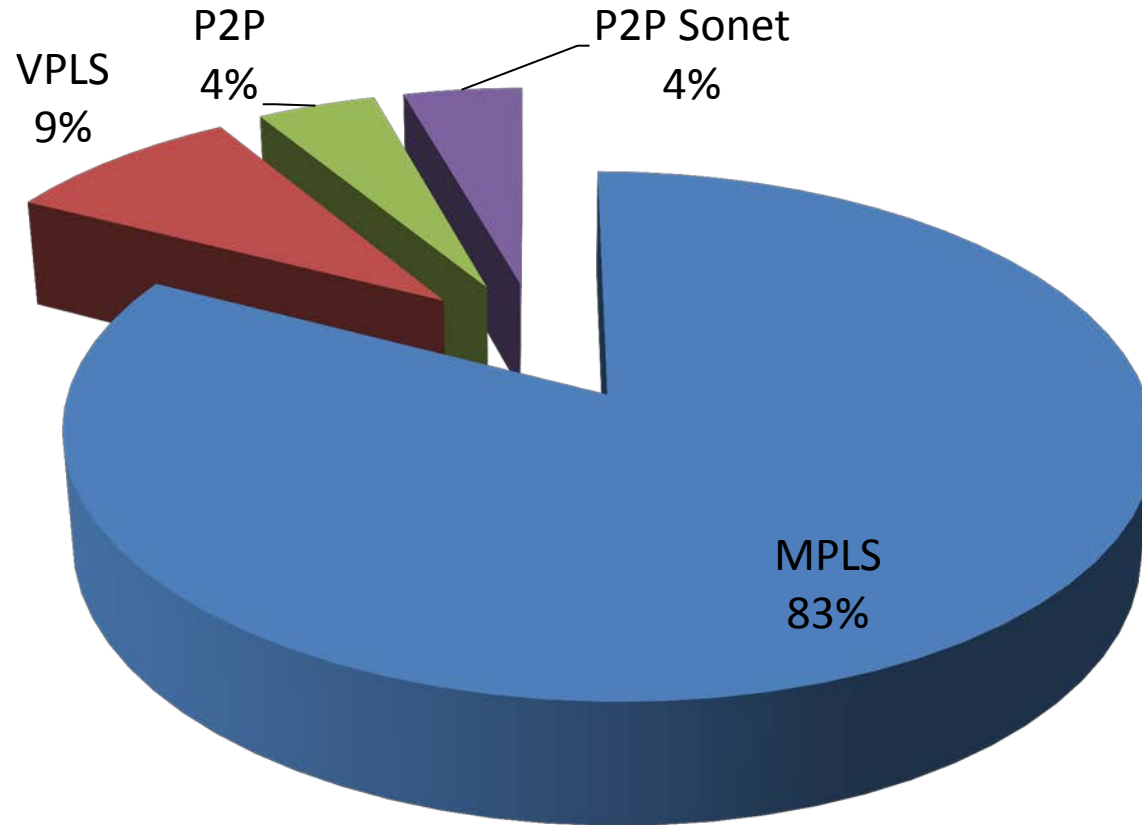
- Definition of Relevant Terms
- Middleware
- **Common WAMS comms. deployment shortcomings**

# Common Deployment Shortcomings: Misc

- Not even using technology that was proven by 1995!
- No admission control (no real “management” without)
  - W/middleware: (1) per variable not entire app (2) in app proxy not router → neither true with network-level admission control like RSVP
- No tracking all traffic at all locations in network
  - Should know what is supposed to be there!
  - Unless this done carefully
    - Not providing strong guarantees appropriate for closed-loop apps
    - Can’t detect spurious traffic (failures, denial-of-service attacks)
- No future proofing: locking into lower-level network and other mechanisms (rather than middleware)
- No real QoS mechanisms, just “**bandwidth over-provisioning**”
  - Can be adequate in (relatively calm) steady state: great availability!
  - But when you need it most, likely to be least available
  - No protection at all against spurious traffic: cyber-attacks, IT failures (jabbering HW or SW), node and link failures,
  - And then very over-cautious on adding more data to network
- **Note: all SGIG projects have the above shortcomings ☹️**

# From Jim/Dan Survey Presentation Tuesday

(1) What type of link layer technology did you deploy in your WAN to collect or distribute the data being collected (e.g. Frame Relay, Point to Point connections, MPLS, VPLS, some combination of items, etc..)?



# Common Deployment Shortcomings: MPLS+IPMC

- MPLS: Weak statistical QoS, not each update
  - Inappropriate for closed-loop applications, remote protection, etc.
    - BTW using separate lines is very wasteful: why not 3-4 well managed paths
  - This will be much more needed as grids get more stressed each year
  - **“DiffServ [and MPLS] does not guarantee that flows classified with a higher priority will really observe a better quality of service than lower priority ones”**
    - Robert Wojcik and AndreJ Jajszczyk, “Flow Oriented Approaches to QoS Assurance”, *ACM Computing Surveys*, 44(1), January 2012.
- MPLS: QoS granularity
  - Has only 3 bit tag for QoS
  - Not guaranteeing or managing at fine granularity (per-flow or subscriber) but putting into one of 8 categories
  - Even having 1000 priorities/classes is not a strong guarantee ☹
- IPMC: selective pub-sub better
  - Lock-in to lower level
  - But better than point-to-point
- Note: **WISP uses MPLS and point-to-point (not even IPMC)**
  - **Won’t scale to many more PMUs than they have now ☹**
  - **(Inadvertently?) ruling out closed-loop apps/remote protection**

# For More Info

1. D. Bakken, A. Bose, C. Hauser, D. Whitehead, and G. Zweigle. “Smart Generation and Transmission with Coherent, Real-Time Data. *Proc. IEEE*, 99(6), June 2011.
2. D. Bakken, H. Gjermundrød, and I. Dionysiou. “GridStat: High Availability, Low Latency and Adaptive Sensor Data Delivery for Smart Generation and Transmission. in D. Bakken and K. Iniewski, ed. *Smart Grids: Clouds, Communications, Open Source, and Automation*, CRC Press, 2014, ISBN 9781482206111.
3. David E. Bakken, Richard E. Schantz, and Richard D. Tucker. “Smart Grid Communications: QoS Stovepipes or QoS Interoperability”, in *Proceedings of Grid-Interop 2009*, GridWise Architecture Council, Denver, Colorado, November 17-19, 2009. Available <http://gridstat.net/publications/TR-GS-013.pdf>.

**Best Paper Award for “Connectivity” track.** This is the official communications/interoperability meeting for the pseudo-official “smart grid” community in the USA, namely DoE/GridWise and NIST/SmartGrid.

4. Slides from Monday: SmartGridComm workshop I led on “Closed-Loop Wide Area Applications, Communications, and Security” (email me or business card)
5. PES GM tutorial summer 2014 full-day tutorial “Overview of Middleware, Distributed Computing, and Fault-Tolerant Computing” (pending approval)  
→ Happy to do a one-day workshop the day before next NASPI



# Questions

- Questions?
- Middleware altar call...

# Backup Slides

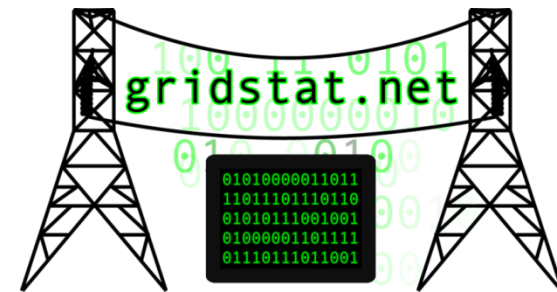


# Workshop 1: Closed-Loop Wide Area Applications, Communications, and Security

**Prof. Dave Bakken**

**School of Electrical Engineering and Computer Science  
Washington State University  
Pullman, Washington, USA**

**IEEE SmartGridComm 2013 Workshop  
Vancouver, BC, Canada  
October 21, 2013**



# Today's Schedule

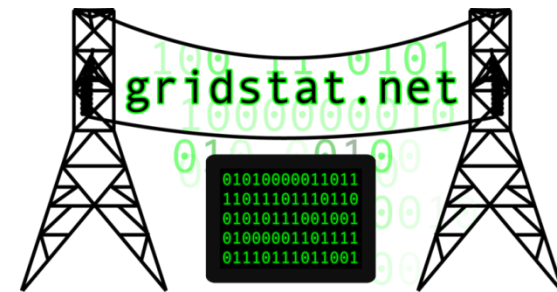
- 830-910** *Applications of Closed-Loop Wide Area Protection and Control.* Dr. Greg Zweigle, Principal Research Engineer, Schweitzer Engineering Labs.
- 920-950** *Communications for Closed-Loop Cyber-Physical WAMPAC.* David Bakken, Professor of Computer Science, School of Electrical Engineering and Computer Science, Washington State University.
- 1000-1030 Break**
- 1030-1050** *Security Issues and Tradeoffs for Closed-Loop WAN Applications.* Carl Hauser, Associate Professor of Computer Science, School of Electrical Engineering and Computer Science, Washington State University. Substituting for Prof. Thothitha Gamage (visa delays).
- 1100-1130** *The Security Fabric for Critical Infrastructures,* John Reynolds, Chief Architect, Security Fabric Alliance.
- 1140-1155** *Panel Discussion and Q&A*

# Communications for Closed-Loop Cyber-Physical WAMPAC

**Prof. Dave Bakken**

**School of Electrical Engineering and Computer Science  
Washington State University  
Pullman, Washington, USA**

**IEEE SmartGridComm 2013  
Vancouver, BC, Canada  
October 21, 2013**



# Outline

- A Brave New World
- Middleware and NASPINet
- GridStat
- Notes on commonly-used WAPMAC technologies

➔ I will be moving **FAST** 😊 ... Q&A time plus can chat right at 1200 for a bit

# Baseline You Can Assume

- Data can be delivered (with GridStat or future sys):
  - Very **fast**: less than 1 msec added to the underlying network layers across an entire grid
  - Very **available**: think in terms of up to 5 9s (multiple redundant paths, each with the low latency guarantees)
  - Very **cyber-secure**: for long-lived embedded devices and won't add too much to the low latencies
    - E.g., RSA adds  $\geq 60$  msec so not for SIPS or closed-loop
  - Tightly managed for **very strong guarantees** (~~MPLS~~)
  - Adaptive: can change pre-computed subscriptions FAST

# Questions to Ask Yourself

- What rate and latency and data availability does my app really need for remote data?
  - Why fundamentally does it need that?
  - How sensitive is it to occasional longer delays, periodic drops (maybe a few in a row), or data blackouts for longer periods of time?
- Can I formulate and test hypotheses for the above?



# Beyond Steady-State-Only Thinking

- Previous is just for steady state: different in some contingency situations?
- How important is my app in that given contingency
  - E.g., simple “importance” number [0,10]
  - How much worse (latency, rate, availability) can I live with in steady state and in given contingencies?
    - But would still get strong guarantees at that lower quality
    - How much benefit do different levels really give me?
  - Can I program my app to run at different rates, or is there a fundamental reason it has to run at one?
- What extra data feeds (or higher rates etc) could I use in a contingency (could get in  $\ll 1$ sec)

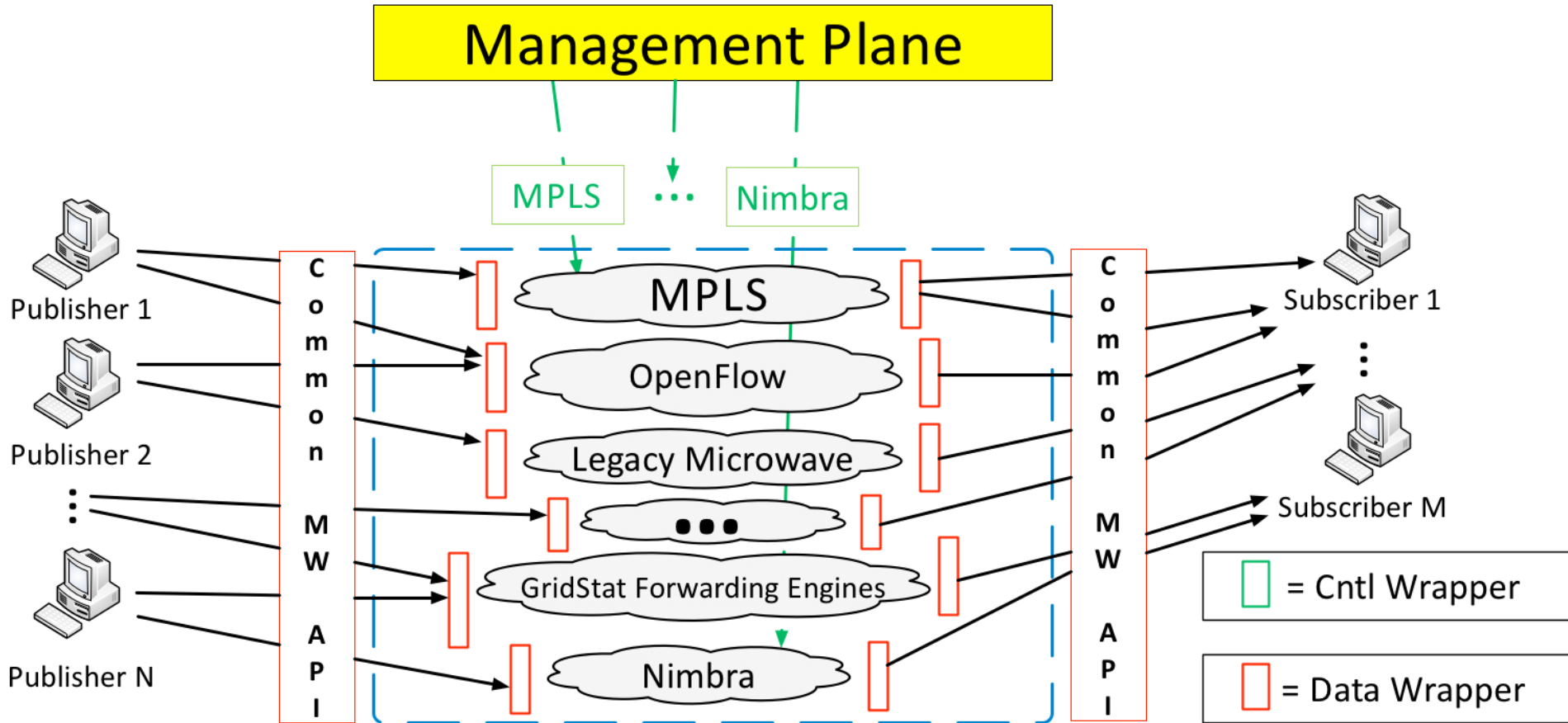
# Outline

- A Brave New World
- **Middleware and NASPINet**
- GridStat
- Notes on commonly-used WAPMAC technologies

# Middleware in One Slide

- Middleware == “**A layer of software above the operating system but below the application program that provides a common programming abstraction across a distributed system**”
- Middleware exists to help manage the complexity and heterogeneity inherent in distributed systems
- Middleware provides **higher-level building blocks** (“abstractions”) for programmers than the OS provides
  - Can make code much more portable
  - Can make them much more productive
  - Can make the resulting code have fewer errors
  - **Programming analogy — MW:sockets  $\approx$  HOL<sup>1</sup>:assembler**
- Considered best practices in other industries for 15-20 years!
- See resources at end for why needed for WAMPAC

# Middleware Integrating Legacy (Sub)Systems



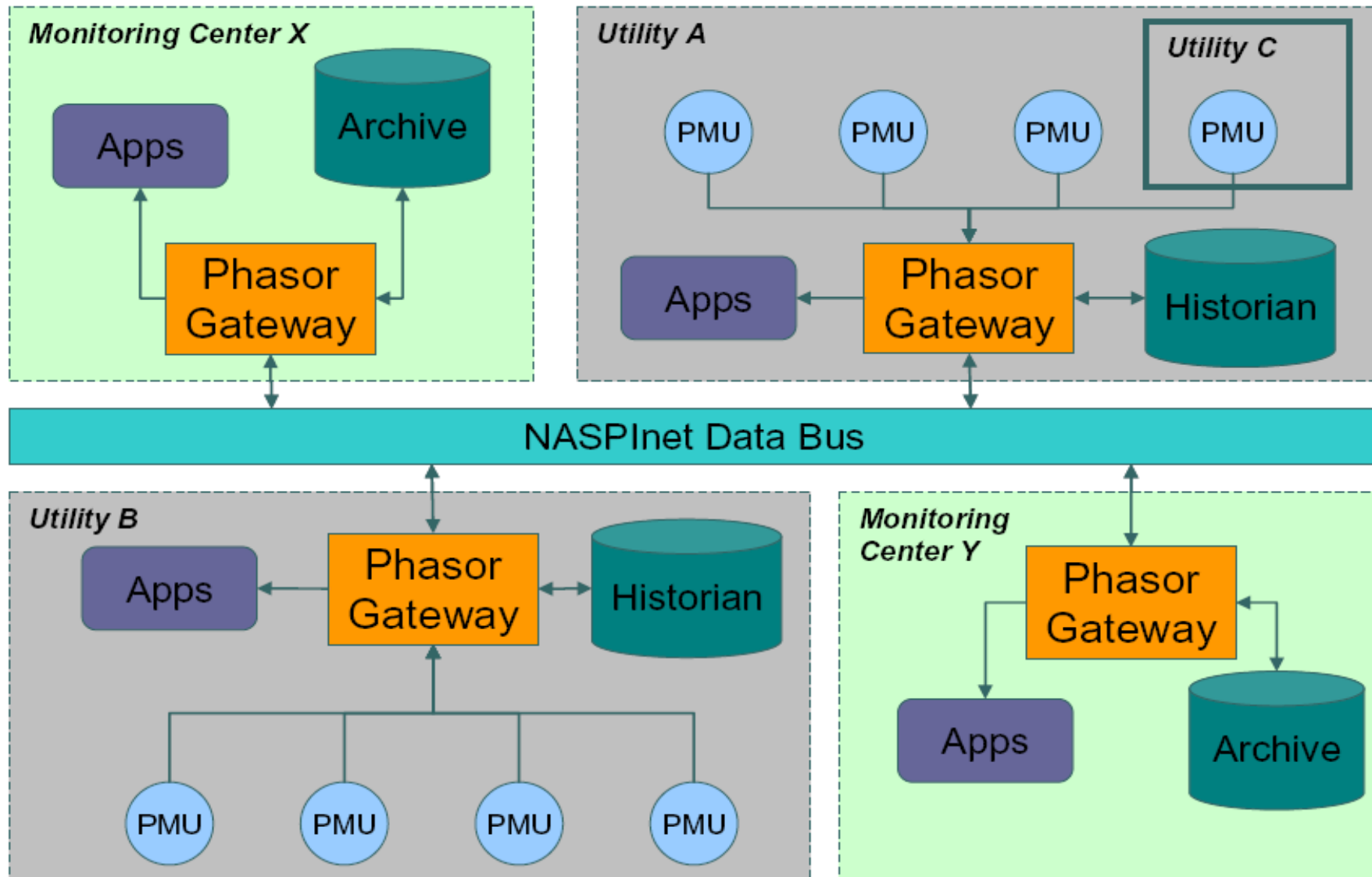
# Req. for Closed Loop Wide Area Data Delivery

- Hard end-to-end guarantees on latency and jitter
- Future proofing: extend to use new network-level technologies, mechanisms for security or fault tolerance, etc.
- Provide a wide range of QoS+ (latency, rate, redundancy/availability) guarantees
- Support extremely low latency guarantees
  - Predictably and for each update
  - Tolerating non-malicious failures
  - Tolerating malicious cyber-attacks
- Note: no commercial industry or military environment has anywhere these extreme requirements!
- Lots of ways to do the above wrong; see more info..

# NASPI

- Vision: “The vision of the North American SynchroPhasor Initiative (NASPI) is to improve power system reliability through wide-area measurement, monitoring and control.”
  - Synchrophasor: a sensor with a very accurate GPS clock...
  - Becoming much more deployed in US, Europe, ...
- Great need for much better data delivery services
  - Can no longer send “all data to control center at the highest rate anyone might want to”
- Very involved with development of “NASPInet” concept
  - Many requirements come from GridStat research (cited)
  - GridStat (most full featured) NASPInet Data Bus framework

# NASPInet Conceptual Architecture



# Outline

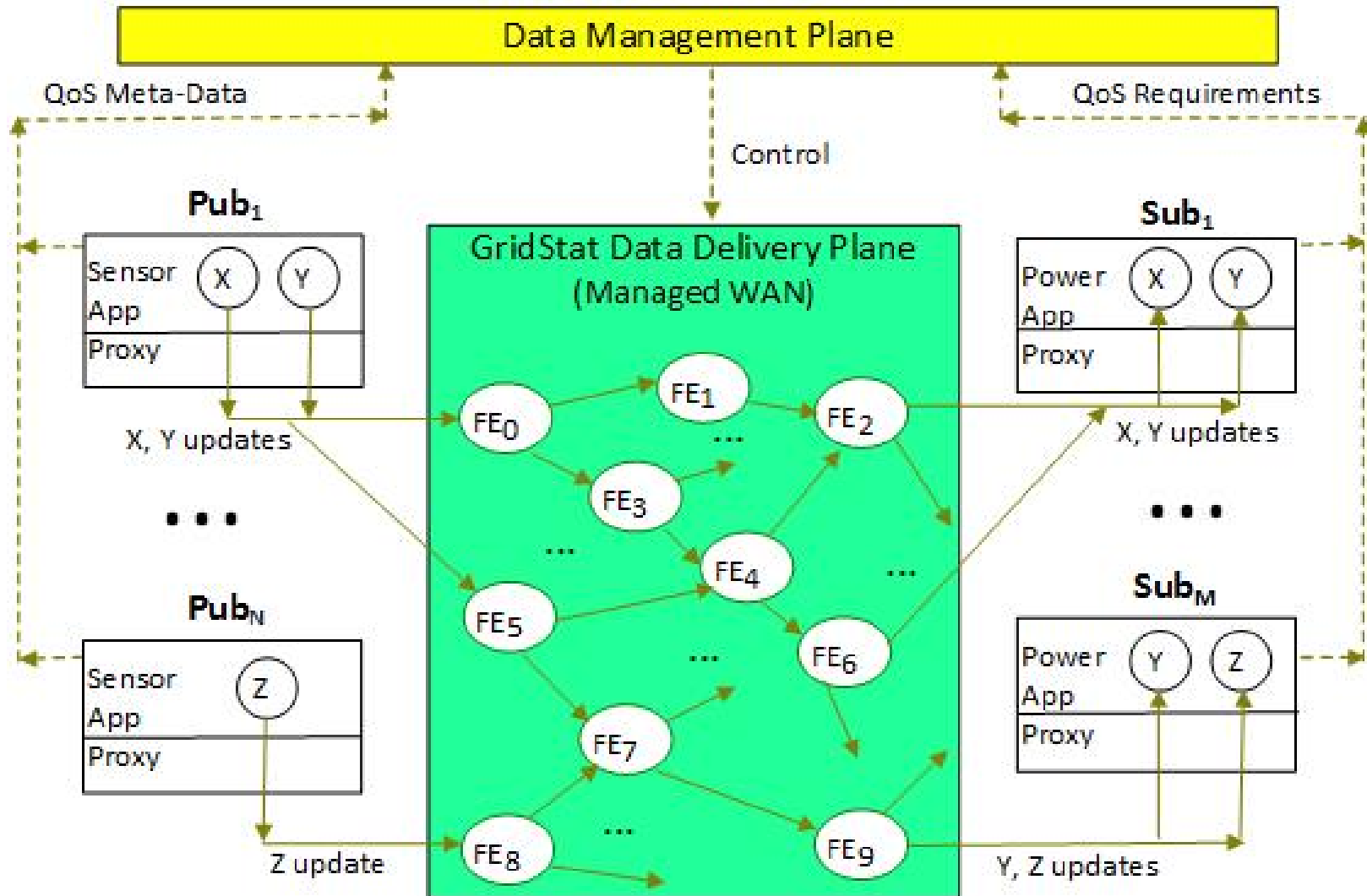
- A Brave New World
- Middleware and NASPINet
- **GridStat**
- Notes on commonly-used WAPMAC technologies



# What is GridStat?

- Bottom-up re-thinking of how and why the power grid's real-time data delivery monitoring services need to be
- Comprehensive, ambitious data delivery software suite in coding since 2001
  - Rate-based pub-sub with
    - Predictably low latency
    - Predictably high availability
    - Predictable adaptation
  - Different subscribers to same variable can get different **QoS+ {rate, latency, #paths}**
- Influencing NASPInet effort

# GridStat: Rate-Based Forwarding



# Overview of GridStat Implementation & Perf.

- Coding started 2001, demo 2002, real data 2003, inter-lab demo 2007-8
  - But power industry moves very, very slowly.....
    - “Utilities are trying hard to be first to be second” D. Chassin
    - “Utilities are quite willing to use the latest technology, so long as every other utility has used it for 30 years” unknown
  - And NASPI is pretty dysfunctional in a number of dimensions
- Implementations
  - Java: < 0.1 msec/forward, 300k+ forwards/sec
  - Network processor: 2003 HW ~.01 msec/forward, >1M fwds/sec
    - Current network processors are ~10x better, and you can use >1 ...
  - Near future: FPGA/ASIC
    - Should be competitive with IP routers in scale
      - Doing much less, on purpose!
- Note: no need to use IP for core ..... (ssshhhh!): less jitter and likely more bullet-proof (no IP vulnerabilities)

# Outline

- A Brave New World
- Middleware and NASPINet
- GridStat
- Notes on commonly-used WAPMAC technologies

# MPLS

- **Weak statistical guarantees** over {location, user, long time}
  - Meant to help ISPs coarsely provision bandwidth w/QoS, not for providing specific QoS for given data variable
  - E.g., Harris' FAA network has 30 minute statistical guarantees
- **Only 8 categories (3 bits) of QoS treatment**, yet many (hundreds, ?thousands) of QoS combinations very useful
  - Its not one size (or 8 sizes) fits all!
- But widely used (with IPMC) by utilities lately, because you can buy it from a router vendor
  - QoS and 1 → many superficially similar to what is needed!

# IP Multicast (IPMC)

- IPMC is not publish subscribe
  - Pub-sub is at middleware layer, dealing with data structures in programs
    - IPMC deals with network packets
  - Pub-sub is more selective: not everyone gets spammed with everything
- IPMC address space must be managed very carefully else instabilities happen
  - Banned in many cloud datacenters

# IEC 61850 & family: The Good

- HUGE benefit compared to wires in substation
- Data model elegant
- Substation Configuration Language (SCL) elegant

## 61850: The Bad

- **Complexity**

- Far more complex than it has to be given the problem it is tackling
- Double the size/bandwidth of IEEE C37.118 with no extra useful info
- Feels to me like a spec doc by a 1975 Mechanical Engineer specifying HW not a 1995 (or later) SW Engineer specifying SW

- **Hype**

- Almost sounds like it will cure cancer at times
  - PJM engineer: 4 substations (ISO has ~30% of the USA footprint)



## 61850: The Bad (cont.)

- **Performance**

- Subscriber apps have to be able to detect missing and duplicates (no sophisticated fault-tolerant multicast)
- GOOSE authentication via RSA signatures (initial version): way too expensive for many embedded devices
  - UIUC paper (Jaiianqing Zhang and Carl Gunter, IEEE SmartGRidComm 2010)
  - WSU paper (Hauser et al paper from HICS 45 (2012))
- Shared-key multicast authentication flavor allows subscribers to spoof a publisher
- GOOSE messages very CPU-intensive with ASN.1 integer fields etc, expensive for many embedded devices
- Have to be careful that the multicast (Ethernet broadcast) does not overload small embedded devices

## 61850: The Bad (cont.)

- **Misc**

- \$3K just to read the spec
- Design by Committee before Full Implementation

- Way better way: IETF and OMG

- David Clark, Internet pioneer (1992)

*We reject: kings, presidents, and voting.*

*We believe in: rough consensus and running code.*

- Rickard Schantz, father of middleware (mid-90s)

*Any time you standardize beyond the state of the practice you are in trouble.*

## 61850: The Bad (cont.)

- **Misc (cont.)**
  - PMUs often need many:one (to a PDC) not 1:many communication
  - Lack of a reference implementation and reference test suite
    - Have to test devices pairwise
- Standard so huge many vendors don't implement all of it; most vendors violate the standard in some way
- No tools (configuration, administration, etc) that work across multiple vendors
- WANs are very different from LANs: partial failures & widely-varying performance (incl. network jitter)

## 61850: The Ugly

- 61850-90-5 is the WAN extension
  - Dec 2010 draft says communications redundancy is “crucial”
  - But the draft has less than one page on it (Sec 8.8) that has no meaningful details 61850-90-5 (cont.)
    - IETF RFC 2991 it relies on has nothing about end-to-end latency, availability, exploiting a more controllable utility infrastructure, tradeoffs below, etc
- **Advanced multicast is hard, fault-tolerant is harder, real-time is harder yet, with security (not ruining performance) is extremely hard**
  - Wide range of properties could trade off, incl. latency, jitter, consistency, throughput, resource consumption, availability, ...
  - Do implementers (or drafters) know what this space of possible properties is, what tradeoffs their given implementations make? Very unlikely...
  - Do utilities/ISOs know what tradeoffs they are being sold, and how appropriate they are for them? Unlikelier!

## 61850: The Ugly (cont.)

- Bottom line: a lead control engineer from a large utility to me
  - 2009: “No way in hell am I letting it [61850] outside my substations”
  - 2011: (ruefully) “I was overruled from above, because its ‘a standard’.”
    - But a standard for doing what? With what properties traded off?

# For More Info

- [bakken@wsu.edu](mailto:bakken@wsu.edu)
- D. Bakken, A. Bose, C. Hauser, D. Whitehead, and G. Zweigle. “Smart Generation and Transmission with Coherent, Real-Time Data. *Proceedings of the IEEE*, 99(6), June 2011.
- D. Bakken, H. Gjermundrød, and I. Dionysiou. “GridStat: High Availability, Low Latency and Adaptive Sensor Data Delivery for Smart Generation and Transmission. in D. Bakken and K. Iniewski, ed. *Smart Grids: Clouds, Communications, Open Source, and Automation*, CRC Press, 2014, ISBN 9781482206111.
- David E. Bakken, Richard E. Schantz, and Richard D. Tucker. “Smart Grid Communications: QoS Stovepipes or QoS Interoperability”, in *Proceedings of Grid-Interop 2009*, GridWise Architecture Council, Denver, Colorado, November 17-19, 2009. Available <http://gridstat.net/publications/TR-GS-013.pdf>.
  - **Best Paper Award for “Connectivity” track.** This is the official communications/interoperability meeting for the pseudo-official “smart grid” community in the USA, namely DoE/GridWise and NIST/SmartGrid.