Towards More Flexible and Robust Data Delivery for Monitoring and Control of the Electric Power Grid

David E. Bakken, Carl H. Hauser, Harald Gjermundrød, Anjan Bose

Technical Report EECS-GS-009
School of Electrical Engineering and Computer Science
Washington State University
Pullman, WA USA

{bakken,chauser,harald,bose}@wsu.edu

May 30, 2007

URL for this paper: www.gridstat.net/TR-GS-009.pdf

***Abstract*—With the increase in the monitoring of status data at very high rates in high voltage substations and the ability to time synchronize these data with GPS signals, there is a growing need for transmitting this data for monitoring, operation, protection and control needs. The sets of data that need to be transferred and the speed at which they need to be transferred depend on the application – for example, slow for post-event analysis, near real-time for monitoring and as close to real-time as possible for control or protection. In this paper, we overview the requirements for the next-generation power grid's communication infrastructure in the areas of flexibility and quality of service, with extensive citations of power industry practitioners and researchers, and analyze implementation options. We also overview technologies in the computer science field of distributed computing that can be brought to bear to help meet these requirements, yet to date have not been discussed in the context of grid modernization. Additionally, we argue against the industry trend of using either TCP/IP or web services for real-time data exchange for fast controls. We then describe GridStat, a novel middleware framework we have developed that is suitable for the power grid and its application programs. Test results demonstrate that such a flexible framework can also guarantee latency that is suitable for fast wide-area protection and control.**

***Index Terms*—Computer control of power systems, energy management systems, SCADA systems, power system measurements, power system monitoring, power system protection, wide-area control, distributed control, quality of service (QoS).**

***Context*—Much of the content of this report, which is intended for computer science laymen in the power industry, will end up in a journal paper on GridStat and in a magazine article calling for more flexibility in the next-generation communication system for the electric power grid. We are releasing this report with the contents of these papers plus additional material because we believe this material needs to be in one coherent document in order to achieve its goal: to help power researchers and those from industry understand both the need and opportunity for more a flexible and robust data communication system for the power grid, leveraging advances in distributed computing technologies that have occurred in the last decade or two. This document thus serves as both a rationale and a manifesto for GridStat, as well as a call for more flexibility and robustness for the grid's next-generation communication system.**

CONTENTS

1. INTRODUCTION

The communication systems for the electric power grids in North America were developed in response to the 1965 North-eastern US blackout. Today's SCADA (Supervisory Control and Data Access) systems, which form the core of the communication system for monitoring and controlling the wide-area power grid, are based on the requirements and technology of that time period. In the years since then, the data collection capabilities of devices in substations, such as IEDs, and Energy Management Systems' computing capacity in control centers (EMSs) have grown enormously. The communication links between them, however, largely retain the characteristics imposed by the technology of 30 years ago: they are slow (a few kilobits per second) and they provide no flexibility to adapt to changing needs. As a result, little operational use can be made of the available wealth of data that is locked up in substations. Utilities have augmented this slow infrastructure with some higher-speed networking technologies. These additional communication facilities do not support the flexibility and robustness which we argue below is crucial for the grid.

These limited capabilities of the power grid's communication system in turn limit the kinds of protection and control that can be done, [HBB05,TBV+05]. This paper reports on an interdisciplinary effort called GridStat, which since 1999 has involved both computer science and power engineering researchers to design and implement a better communication infrastructure for the electric power grid [BBB00,BEB01,GDB+02,BBD+02,HBB05,TBV+05]. GridStat is meant to augment (not replace) deployments of IP, SCADA, ATM, SONET, and other networking technologies to provide a managed layer on top of them. This allows data delivery services to span different network technologies in different utilities, exploiting their underlying characteristics such as quality of service guarantees while providing great flexibility and portability yet incurring very little additional overhead. It is a much more reasonable alternative for delivering critical operational data than using either TCP/IP or web services, both of which are used or proposed for this in the power industry, as we show below.

In the following sections we: further address the limitations of today's power grid communication systems (beyond what is in [HBB05]); elaborate on the requirements for improving the grid's data communication system (most of which are not met by communications products offered by utility vendors today); describe technologies developed in recent decades in the computer science field of distributed computing which can be brought to bear to provide better communication services; describe the GridStat architecture and the design of its principal components; report the results of an experimental evaluation of GridStat's performance, including comparing it to performance requirements of today's grid; describe advanced GridStat mechanisms that provide more flexibility and adaptability along with ongoing and planned research; provide detailed examples of some advanced capabilities that a more flexible and robust communication system enables. We finish with an overview of related work on both the electric power and computer science sides and some conclusions about the feasibility and potential for improved power grid communication systems.

2. Problems with Today's Grid Communications

In this section we overview the limitations of today's communication system for the electric power grid. These limitations have had severe consequences. As noted in [Cle07] (and by others):

With the exception of the initial power equipment problems in the August 14, 2003 blackout, the on-going and cascading failures were almost exclusively due to problems in providing the right information to the right place within the right time.

This failure of delivering sensor and other data was not unique to this US blackout, in fact it was a major contributor to the 2003 blackout involving the Swiss-Italian border, and other major blackouts. In most of them, serious problems occurred an hour or more before the blackout, but were not acted upon due to inadequate situational awareness of the operators, which was inevitable given the poor data communication infrastructure. [PKT06] concludes with a list of the 4 root causes of major blackouts (especially those in 2003), and the first root cause given is "a lack of reliable real-time data". [AF06] reports that a NERC study of 117 power system disturbances by NERC showed that "In 56 cases, problems were due to monitoring and control inadequacies".

For more details of these communication limitations, see [HBB05]. We now provide further description of the consequences of these limitations.

A. Consequences of Limited Communications

SCADA, the main communication technology used for monitoring and controlling the electric power grid today, features star-connected, point-to-point links connecting substations to control centers. Compared to today's networks, the links are slow—only a few hundred to a few thousand bits per second. The control center polls each substation for data once every 2–4 seconds so the control center's picture of the operational state of the power grid can fall behind when the power grid is undergoing disruptive events. SCADA protocols are also inflexible in that they can only support data delivery from substations to a single location. SCADA's limitations are well known [Dag06], and its limitations contribute to lack of situational awareness by operators in neighboring control areas and precludes better coordinated automated responses to crises.

Because of the communication limitations, automated real-time controls on today's power grids are almost all *local* in scope, meaning that they are located in a single substation where the input data is observed and the control actions are taken. Examples include protective relays, switched capacitors/inductors (and loads), transformer taps (including phase shifters), FACTS devices, exciters (including power system stabilizers), and governors. Existing automated wide-area controls are of only two types: Automatic Generation Control (AGC) runs on a slow 2–4 cycle using SCADA data; Special Protections Schemes (SPS), also known as Remedial Action Schemes (RAS) require very fast (low-latency) data delivered between substations using customized, dedicated communication links. Unlike AGC, SPS designs are not standardized: each design is unique, and they tend to be very expensive both to design and to implement. Most wide-area control actions today are taken by human operators acting through SCADA systems.

There is thus a large gap in the time scale between real-time controls (mostly local with latency of a few milliseconds), and the capabilities of human operators, [Bos03]. Data that operators receive through their SCADA systems are already several seconds out of date, to which must be added the time for reaction and decision. With the power grid becoming more dependent on power transfers between control areas in recent years there is increased concern about dynamic phenomena, such as oscillations resulting from instabilities, which can lead to major outages. Automated controls that can respond to these phenomena require inputs from many points in the grid and faster response times than can be provided by human operators observing SCADA data.

Computer networking technology has changed enormously since the basic SCADA architecture for power grid monitoring and control was developed. Some SCADA systems take advantage of this, if at all, mainly by layering the old architecture and protocols over new low-level technology such as optical fiber and Ethernet. However, very few, if any, of the results coming out of distributed computing systems research and development in recent decades have been utilized in the power grid. Advances in systems architecture, naming and discovery, synchronization, consistency and fault tolerance all are relevant for power grid communication as is the idea of *middleware* that embodies these results in software artifacts. We elaborate on these further in Section 4.

On the power side, [Mack06] notes, “literally thousands of analog and digital data points are available in a single Intelligent Electronic Device (IED).” Each upgrade of old substation equipment with IEDs enables collection of large quantities of data, not previously available, that are potentially useful for control and monitoring. Indeed, the stranded value of these unconnected IEDs, in terms of their ability to help make the grid smarter, is huge [IE07]. In addition, synchronous Phasor Measurement Units (PMUs) are increasingly being deployed, some even as part of location protection systems (*e.g.,* the SEL421 relay). Using GPS-based time synchronization the synchro-phasor measurements provided by these devices multiple times per second (and even per AC cycle) have great potential to improve wide-area monitoring and control, [MKO96, TEM+05], but require much greater communication bandwidth and lower latency than SCADA systems provide. For the limited number of PMU deployments so far, these needs have been met by installation of special-purpose networks to distribute and collect synchro-phasor data. However, this approach is not economically sustainable for large-scale deployment of PMUs. Furthermore, it does not make good use of the specially-installed network capacity. For example, other high-rate substation data may be even more useful in the wide area but this is not considered in the current PMU distribution network design. Similarly, SPS/RAS implementations today rely on dedicated communication links that are costly to design and implement, are not readily adapted as needs change, and cannot be used for any other purpose. Given recent advances in optical wave division multiplexing and other ways to isolate different traffic flows in hardware, such dedicated networks are no longer necessary or even advisable.

B. Towards More Complete Grid Communications

The essential goal in designing a communication system for the power grid is to make data available where they are needed, when they are needed: these data should go “from all substation devices to any client” [MCG+07]. Some data may be needed in multiple locations, such as control centers (and their backups), regional security[1] coordinators, neighboring control centers, ancillary service partners, or even in other substations in support of special protection schemes. For example, one utility has investigated the use of both operational and non-operational data across all of its entities [MRW+07]. They noted:

[1] We note that the meaning of the term “security” here is in the traditional power sense, involving how stable the power grid presently is and would be in the face of a single foreseen contingency. When possible, we try to call this “grid security” or “power security”, and use the term “cyber-security” when referring to protection from hackers etc.

> *The utility identified more than 20 user groups for whom data would be collected, integrated, and delivered. These groups include personnel involved in distribution planning, transmission planning, substation operations, load dispatch, relay maintenance, power supply operations, substation design, and other functions.*

The usefulness of all these data depends, in part, on the ability of the communication system to deliver them at the rate and latency needed by control and monitoring applications, as we describe further in Section 3. For these and similar reasons, Clark Gellings of the official research arm of America's power utilities, Electric Power Research Institute (EPRI), has stated, [EPRI03] (emphasis ours):

> *In order to create this new power delivery system, what is needed is a national electricity-communications superhiway that links generation, transmission, substations, consumers, and distribution and delivery controllers…. The ultimate challenge in creating the power delivery system of the 21st century is in the development of a communications infrastructure that allows for universal connectivity.*

At GridWeek 2007, a recent gathering in Washington, DC of those interested in modernizing the power grid, it was noted in a plenary presentation that, [Hof07] *(emphasis is the presenter's):*

> *The reliability of the nation's bulk power system can be improved by developing an interconnection-wide real-time monitoring system to give system operators a near-instant picture of the transmission system's health.*

The crux of the matter is this: continued piecemeal expansion of the power grid's communication system is unnecessarily expensive, and does not even meet today's requirements. As noted previously, lack of situational awareness, partially due to inadequate communications, has been a major contributor to inadequate operator responses to circumstances leading up to most recent major blackouts, such as those in the US and Italy in 2003.

We believe that it is imperative that the overall effort to improve the power grid be judicious about how the power grid's communication system is modernized, and avoid the one-of-a-kind engineering approach, such as used in SPS/RAS deployments and other hardwired approaches that are used today. Indeed, the systems engineering issues involved may well be the hardest part of grid modernization, as noted in [Gun07]:

> *Any competent engineer can glue two systems together and make it work. But it takes sound architecture and good systems engineering practice to make the resulting system scaleable, manageable, secure — and oh, yeah … address the business case.*

What is needed to support this is a more flexible alternative to today's patchwork of communications in order to meet the evolving communication requirements of the power grid over many decades. The improved communication system must be accompanied by focused R&D that exploits it. This is something of a chicken-and-egg problem: without cheaper and more flexible communications, power researchers are unlikely to experiment with new communication topologies and control/protection schemes utilizing them. And without better control and protection schemes, the investment in improving the grid's communications cannot be justified. This points to the great need for coordinated research between computer scientists and power engineers.

Some in the electric power industry have recognized the need for a much better communication infrastructure. [Gal07] notes that the intelligent grid will need ubiquitous communications: it will have many sensors and “these data will be communicated wherever they are needed,” and smart digital controls will be “based on the continuous flow of information from the sensors.” [Uti07a] notes that “advanced networking from the substation to the premise creates the fundamental platform on which smart grid initiatives are built,” thus showing that an improved communication infrastructure is also needed for distribution, not just generation and transmission. (Of course, if done right, such an infrastructure can be shared between these fundamental roles in the grid, which will be blurring somewhat with the advent of demand response and some other smart grid applications.) Others have noted the need for a better communication infrastructure (not just more bandwidth), including [Ene06, Eco04, Taf06, PJM07, Uti07b, BPA06, IntelliGrid04].

C. The Need for More Reliable Grid Communications

The power community is beginning to recognize that the communication network needs to be reliable, something that seems to have simply been assumed in the past. [XMV+02] contains an analysis of 162 disturbances between 1979 and 1995 and indicates that information systems have an impact on power grid reliability, and points out major deficiencies in the current communications scheme. As noted in [Cle07] *(**emphasis** is the author’s):*

> *The reliability of the power system is increasingly affected by any problems that the information infrastructure might suffer, and therefore **the information infrastructure must be managed to the level of reliability needed to provide the required reliability of the power system infrastructure**.*

[BPA06] notes that one of the challenges in creating an enhanced wide-area measurement system is “preventing engineers from wasting valuable time by tracking down and fixing data problems when they should be studying the system.”

The above uses of better communication in the grid are largely for online control and protection. [HP06] stresses that control and protection must be integrated and must have very good communication infrastructure:

> *In any realistic scenario, the total protection, operation, and control package must be considered as an integrated system. It is being recognized that the advent of new technologies has provided two of the most valuable elements of this total package: system integrity protective schemes (SIPSs) and wide-area-based protection systems (WAPSs)…. To implement effective, intelligent SIPS appropriate for the prevailing power system condition, **it is essential that a realtime, fast-acting, system-wide data collection system be available** (**emphasis** ours).*

In addition to improving control and protection, an improved communication infrastructure can also be important to support postmortem investigations of large-scale disturbances and blackouts [Dag06]. It can also support the business case: [Roo06] notes that monitoring systems that can run the system at higher load levels (and of course hence with more profitably) are one of the “exciting new technologies that will be tools for the future”.

Finally, we note that some projects in the power industry have used the ubiquitous TCP/IP protocols to provide reliable data delivery; [MRW+07] is a large-scale example of this. [Kro06] notes this trend and its cyber-security implications. However, it is not just cyber-security that is a problem with TCP/IP. Computer science researchers have long understood that TCP/IP, while “reliable” and quite useful for general purpose web applications, is inadequate for many mission-critical wide-area applications, because it has narrow coverage of failures and a high and unpredictable latency (for a recent study, see [BCH+05]). The IntelliGrid project from EPRI realizes this, and has an entire section overviewing these limitations. Fortunately, distributed computing technologies in recent decades have shown a number of ways to provide broader coverage of failures with latencies that are lower and more predictable, using advanced protocols and communication services built, for example, over UDP/IP, as we show later in this report.

3. Requirements for Next-Gen. Grid Communications

In this section we describe requirements that should be met by any communication system that delivers power grid *status data*, including traditional binary status values, analog readings, vectors or other complex data types, as well as any other real-time data that needs to be shared, such as notification of decisions made by a balancing authority or ISO. Today's communication system for the grid meets few of these requirements.

These requirements fall in the areas of quality of service, flexibility, and other misc. requirements (mostly related to cyber-security). This list is not intended to be exhaustive, but to help motivate the wide range of properties a future communication system for the power grid must at a minimum provide. Finally, we note that most of the requirements in this section cannot be implemented at the network layer alone: most at least need help from higher layers of software (using distributed systems technologies), and many necessarily must be implemented completely above the transport layer of the traditional computer network protocol stack (*i.e.*, above TCP/IP or UDP/IP).

A. Quality of Service Requirements

The first kind of requirements which must be met are for *quality of service* (QoS). QoS is a broad category which refers to runtime behavior, mainly performance. Some researchers include cyber-security related issues with QoS, but most do not. We do not here; they are below in Section 3.C.

The major performance-related requirements are *latency*, the delay that the communication system imposes, and *rate*, the number of updates which must be delivered per unit of time. Today's requirements for latency and rate are given in [IEEE-1646]. Fast applications have latency requirements of about 4 ms within a substation and 8–12 ms external to substations. Medium-speed applications have latency requirements of 16 ms internal and 1 s external to a substation. Low-speed applications have latencies of 1–10 s. For PMU data, a latency of no more than 20–30 ms is required. In the future, intra-substation performance requirements are expected to decrease to 2 ms for synchronized sampled waveform data for protective relaying [IEEE-1646]. It is crucial that the communication system supports these required latencies for all applications, and in particular it must ensure that delivery of updates for slower applications does not interfere with those for the faster applications. The latency requirements from [IntelliGrid04] are similar to the above, with the 2 lowest latencies required being QOS-1 (4 ms) and QOS-2 (10 ms).

The requirements for rates span "periodic update intervals ranging from a minimum of 1 cycle up to 30 days," [IEEE-1646]. Thus the rate requirement today ranges up to at least 60 updates per second and the 1646-specification further notes that for system protection using PMUs a rate of 250 updates/sec may be required.

Availability of data is defined in [IEEE-1646] as being the "state in which data are where the user needs them, when the user needs them, and how the user needs them." In other words, data delivery must be predictably timely and reliable, and in the expected format. Availability of data (and, more generally, any kind of fault tolerance) inevitably requires redundancy, often in multiple forms across a large system: *spatial redundancy* (multiple copies of a server or data link), *temporal redundancy* (resending data at a later time), and *value redundancy* (e.g., checksums and error-correcting codes). Value redundancy is typically handled at the network level, as is temporal redundancy, most commonly in the form of message acknowledgements and retransmits in network protocols. Spatial redundancy must be implemented above the network level, however, and falls in the realm of distributed computing. Some form of spatial redundancy is required to provide highly available data, for example, to avoid single points of failure. A final aspect of providing availability of data is that the communication infrastructure should be *self-healing*, able to reconfigure itself at many levels in the face of failures and cyber-attacks.

Availability of data is clearly important, but the topic has seldom been systematically considered by power researchers. One example of where it has been studied is [BTB04] which looked at the impact of communication delays on the performance of third-party load frequency control and found that missing or late control signals could render the system unstable. But in general, contingency planning does not (yet) address communication failures, despite the heavy reliance of control schemes on that communication.

An example of the need for availability of data is dynamic security assessment (DSA), a kind of SPS. DSAs perform an assessment of the near-term power security of the grid based on a snapshot of the system condition (sensor data). These have traditionally been offline, because the simulations have not been fast enough. Lately, they have begun to be deployed in a handful of countries [WM06], and [SMD+06] notes:

> *With the recent advances in computer technology and the intra- and inter enterprise communication networking, it now becomes cost-effective and possible to apply distributed computing to online DSA in order to meet real-time performance requirements needed in the electric utility industry.*

These online DSA systems have three different categories of reliability requirements: occasional use, continuous-use, and mission-critical, [WM06]. It is thus important for the communication system that provides these snapshots to support a range of data availability. If not, then either the mission-critical DSA applications will not have the data ability they require, or the occasional-use ones will inevitably have more data availability than required, and perhaps much more, due to wasteful over-provisioning of communications.

A further example of the need for supporting a range of data availability is in the IntelliGrid effort, which provides “a vision for the electric delivery system of the future” and has very detailed case studies and technical analyses [IntelliGrid04]. It lists, in Table 43, different levels of availability of information for different categories of applications. The levels are given in Table 1.

Communication safety is defined as “measures and controls to avoid any deterioration or losses of information (reliability)” [IEEE-1646]. For any complex, wide-area data dissemination system, such mechanisms and controls will necessarily be implemented in multiple layers at multiple locations of the system, most of them above the network layer. Communication safety is a catch-all phrase, implemented in part by mechanisms that provide availability of data, as outlined above, as well as traditional cyber-security properties of confidentiality and integrity, which are discussed below.

Level	Availability (%)	Downtime/Year
Ultra	99.9999	~ ½ second
Extremely	99.999	~5 minutes
Very	99.99	~1 hour
High	99.9	~9 hours
Medium	99.0	~3.5 days

Table 1: Data Availability Requirements for Application Categories [IntelliGrid04]

B. *Flexibility Requirements*

The grid's future communication system must support many kinds of flexibility. It is important to note that these span a long projected lifecycle consisting of many phases:

- Design, development, and deployment of the initial communication system and the applications which exploit it;
- Adaptation during operations to power anomalies, IT failures, and cyber-attacks;
- Decades of maintenance, including adding new power applications and underlying IT technologies which have not yet been conceived.

Insufficient flexibility will mean that the communication system becomes ineffective or unaffordable, or both, at some point, with serious consequences for the stability of the grid.

1) Motivation from Extending Today's Control and Protection Schemes

To help motivate the need for flexibility, note that all data collected at high frequencies within a control area cannot possibly be brought into that area's central EMS/SCADA (let alone all the data for an entire interconnection). Rather, the right data need to go to the right computer at the right rate and latency, depending on the purpose of the computer's applications. This functionality and data usage will change over time, so the communication system that moves the data must be very flexible to enable this. Ultimately, the monitoring, operation, control, and protection of the power grid should be changeable by software alone, rather than by having to dispatch technicians into the field.

As another example, consider SPS/RAS schemes. With more flexible communications, an existing SPS could be updated, or a new SPS installed, completely in software. This would include changing the input data, changing the logic which used those data, and changing the output (control) signals. Instead of using offline studies to set the controls every few months, online computation could be used to adapt the controls continually, using real-time data.

Also, consider PMU and *wide area measurement systems (*WAMS) deployment. With more flexible communications, PMU data could be handled just like any other data (the distinction is already blurring), rather than with today's special-purpose networks and phasor data concentrators whose financial cost limits the attractiveness of PMU deployment and whose latency cost limits the kinds of applications that can be created to use the data. With this flexibility, low latencies, and good availability of data, the PMU monitoring of today can be extended to control tomorrow.

Finally, we note that many other underlying services of the power grid can utilize better communication, including fault detection and location and even overload detection.

2) *Flexibility Required*

The following kinds of flexibility (most of which must necessarily be implemented above the network layer) are needed in order to achieve the envisioned results:

Multicast: multiple recipients must be able to receive updates of the same status datum.

Heterogeneity of communication topologies: The structure or shape of the patterns of communication must be changeable easily and in software. Further, multiple different topologies of communications between different entities must be allowed at the same time.

Heterogeneity of delivery latency and delivery rate: Different recipients of the same data must be able to receive updates at different latencies and rates, [IEEE-1646 Table 3 & 5.6.1]. In [BPA06] this is called "bandwidth-flexible communications." For example, consider a simple sensor value. A protection application in the same substation may require latencies of a few milliseconds and a rate of 60 or more per second, while an application in an ISO that is loosely monitoring the sensor's value might accept a latency of a second or more and a rate of a few updates per second. Note that if all subscribers to a data variable had to be given one rate, then *either* the substation monitoring application would get much less than its required rate *or* the wide-area network would carry a lot of unnecessary traffic to the ISO.

Temporal synchronism: the delivery system must be able to deliver a consistent collection of data from multiple sources, despite any rate filtering. This is essential for delivering phasor measurement unit (PMU) data, and is described further in context in Section 6.B.

Heterogeneity of computing resources: The communication system must function (and provide QoS guarantees) despite having to span multiple networking technologies, CPU or device architectures, programming languages, and operating systems or runtime systems.

Extensibility: the communication system must be able to incorporate and manage new kinds of computing resources with new and/or different capabilities over its long intended lifetime [Hos07].

Open architecture: The communications system must be designed, developed, and deployed in a way which allows for easy interoperability across multiple vendors' products, [PJM07,BPA07]). Failing to insist on open architectures results in "silo" or "stovepipe" systems. Whatever the term, closed systems lack of flexibility and extensibility makes them far too expensive and brittle to form the basis for a long-term communication infrastructure for the power grid.

Multicast can be implemented completely in the network, or by cooperation between the network layer and higher layers [Bir05]. Typically, for reasons of flexibility and robustness, computer scientists implement multicast at least partially above the network layer while exploiting underlying network features such as Ethernet broadcast or IPv6 multicast (neither of which provides reliable delivery or any kind of ordered message delivery).

Both communication topology flexibility and heterogeneity of latency and rate require a sophisticated management infrastructure controlling the communication network; and providing heterogeneity of resources and extensibility requires well-designed middleware, a layer of software above the operating system but below the application that embodies time-tested techniques for resource heterogeneity and extensibility. In the last decade, distributed computing researchers have devised middleware that provides QoS while cleanly separating the delivered QoS properties' architecture from the underlying computing resources, [ZBS97]. This is essential in order to provide QoS in the face of heterogeneous and changing computing resources.

3) *Flexibility for Tomorrow's Applications*

Today's power grid communication system has almost none of the kinds of flexibility described above: it evolved to meet needs as they arose so each design and implementation decision closely reflects the problem at hand. However, power researchers and engineers looking to the future have articulated that there will need to be many new patterns of data exchange between entities in the power grid [WMB05, IntelliGrid04], so providing these kinds of flexibility is crucial for the future of the power grid.

Many of these proposed applications fall under the rubric of what is often called the "smart grid," which includes a very wide range of applications. [IntelliGrid04] notes in Section 3.4.1.2 (entitled "Technologies for a Resilient Communications Infrastructure," implicitly indicating the need for reliable communications) "The communication infrastructure of IECSA should support restoration technique(s) that offer a wide range of services that meet varying resilience requirements of applications." Given that the different applications have varying resilience requirements, it follows that their communication infrastructure also does. *Advanced Metering Infrastructure (AMI)* is a two-way communications system whereby customer meter information is automatically collected, and price signals can be given. It is a key enabling technology for *Demand Response (DR)*, whereby customers can adapt their demand for electricity based on price signals, for example by delaying the use of an air conditioner or dishwasher when a blackout is being approached (and hence current prices are very high). This is done usually without human intervention, through pre-programmed software and even appliances. DR has great potential to help with the stability of the grid, because it can decrease demand when and where needed. DR technology can be viewed as creating "virtual generators" that can be turned on when needed. These virtual generators have two very useful properties that physical generators do not: they can be used very close to where power is needed (as contrasted with physical generators, which often must be very far from high demand areas), and they are much better for the environment. Another smart grid application, *smart wires*, is a Distributed FACTS technology whereby multiple distributed static series compensator (DSSC) modules act in a coordinated manner, [DBS+07].

These smart grid applications (and all others we have seen) rely heavily on communication, by definition, and they have a wide range of requirements in flexibility and other areas. Indeed, smart grids inherently need smart communication: flexible, timely, robust, secure, and self-healing. *Clean energy solutions* also need close integration (and hence communication) of such diverse distributed generation and storage technologies as chilled water plants, thermal storage, concentrated solar deployments, fuel cells, batteries, heat pumps, bio-mass/bio-fuels, and wind.

The flexibilities and QoS outlined above are not a unique requirement for the US. The need for better communication for the power grid, and the rapid movement towards a smart grid and clean energy, are also prevalent in Europe [BBH06], and elsewhere.

C. Other Requirements

Other important requirements do not fall under the rubric of either QoS or flexibility. While the main focus of this paper is explaining how the QoS and flexibility requirements can be met, we overview other requirements for completeness. More information on cyber-security and the power grid can be found at [ENE06, TCIP07, Cle07, IntelliGrid04 Appendix A].

Cyber-security properties include *confidentiality, integrity, and availability*. Confidentiality requires that only authorized recipients can read particular data; integrity requires that recipients be able to determine that a data item was created by a particular entity and that it was not modified in transit. Confidentiality and integrity can be implemented using traditional cyber-security techniques such as encryption, though how to employ those techniques in something as large and diverse as an electric power grid is open research. Availability of data was discussed above. The role of these cyber-security properties in power grid control and monitoring applications is discussed in [HBB+07]. Some experts also include *non-repudiation* in security-related requirements for the power grid, the property that an action (such as one application issuing control signals to change generation) cannot be denied: it provably can be attributed to the application and/or user that issued it, [Cle07].

It is important to note that cyber-security research for decades has strongly focused on encryption and flow of data (the confidentiality requirement above), [Bir06], but starting in the mid-1990s many researchers (and, more importantly, research funding agencies) realized that this was not the entire scope of necessary research related to cyber security. Thus, confidentiality and integrity have been added to most researchers' view of security. Many practitioners outside of the cyber security research community still view cyber security as mainly involving encryption. However, as cyber security pioneer Roger Needham has noted: "If you think encryption is the solution to your problem, then you do not understand encryption, and you do not understand your problem" (quoted in [VR01]). The power industry, is increasingly realizing the broader scope required for cyber-security in the power grid, [Cle07], though many certainly still believe the obsolete notion that encryption and strong passwords, with perhaps a firewall, is most or all of the solution.

Other requirements are related to cyber-security, but separate from it. If more data is being shared between more entities in the power grid, as is the trend, then *trust management* becomes very important [OR02, Dio06]. In particular, it is crucial for participants in the power grid to have a systematic and comprehensive way to reason about (1) how much trust to place in data they receive, especially when it comes through chains of processing, and (2) how much access to their data to provide other, possibly untrustworthy, participants. The inputs to these decisions are dynamic, and it is important that the policies that make these decisions be changeable in software (flexible).

4. OVERVIEW OF SUPPORTING DISTRIBUTED SYSTEMS TECHNOLOGIES

A distributed computing system is one where a number of computers connected only by a network are coordinating to provide some service or set of services. This is an area of computer science that has come almost out of nowhere in the last two decades to be something which programmers have to deal with frequently (for an overview, the reader can see [TvS07, CDK05], [Bir05] for more about reliability techniques for distributed systems, and [VR01] for more discussion of architectural issues and tradeoffs in distributed systems). For more on why wide-area distributed systems are hard to program, even for expert computer scientists, and what can be done about it, see [ZBS97].

A. Technologies Not Yet Considered for the Power Grid

Some of the areas in which computer science researchers have made major advances in distributed computing in the last few decades include:

Distributed architectures: different ways to deploy components (including their roles and interactions) in a distributed system; the resulting tradeoffs between performance, availability, cyber-security, flexibility, etc.

Naming and Discovery: ways to enable programs to use high-level, human-readable (and hence meaningful) names to refer to components in a distributed system, rather than the low-level, machine-readable identifiers (such as a 32 or 128 bit IP address) which computers must ultimately use. Different naming systems are appropriate for different kinds of distributed systems: the right one can provide a great deal of flexibility with good performance and scalability; the wrong one can cripple a distributed system or make it unwieldy to change as the system inevitably evolves.

Synchronization: ways that distributed components can coordinate their activities to achieve a common goal; how a set of components can reach a decision in the face of bad values, failed nodes, variable network delay, etc.; ways to reason about a system where the relative ordering of distributed actions cannot always be known; ways to let components in a distributed system avoid interfering with shared data structures by providing distributed mutual exclusion; how a group of components can elect a leader to simplify the process of agreement despite failures and other anomalies.

Consistency: different ways in which caches and replicas of data can be coordinated, and the resulting level of consistency (compared to a single copy of the data); tradeoffs between consistency level, performance, and reliability; algorithms to take a snapshot of the global state of a distributed application program and their inherent tradeoffs between performance and consistency.

Middleware: software layers above the network which provide high-level programming models (such as distributed objects) while shielding programmers from the diversity of programming languages, CPU architectures, operating systems, network technologies, and other differences across a distributed computing system. Middleware, especially CORBA-based distributed object middleware, has been in wide use since the mid-1990s in industries as varied as aerospace, railroads, and telecommunications. CORBA's predecessor, Cronus, has also been in use as of the mid-1980s in military applications, [STB86,GDS86], though in these early days it was sometimes called a "distributed operating system" (which is now considered to be something that is separate from middleware, so not all research in the 1980s on "distributed operating systems" is on middleware, in fact most is not). A familiar form of middleware, relational databases, was in widespread use even before the 1980s.

Fault-tolerant computing: how to do all the above techniques despite a range of failures, including message loss, variable network delay, corruption of messages, crashes of server computers, etc.; how to use replication and other techniques to provide services that are resilient despite the above-named anomalies; tradeoffs between fault tolerance and performance; how to structure a system in order to enable it to degrade gracefully in the face of anomalies and overload; how to perform fault detection, fault isolation, and error correction at many layers in the system, and the resulting tradeoffs.

Most of these technologies did not even exist in the early 1980s. Note that all are above the network level: their essence is *embodied knowledge about how to use* a network. These kinds of above-the-network technologies can greatly improve the power grid's communication system, but to date, with a few possible rare exceptions, have not been adopted or even proposed for use in the operational side of the power grid. Indeed, even [IntelliGrid04], in Appendix C ("Resilient Communication Services"), noted *(emphasis is ours):*

In this appendix, we review a number of state-of-the-art and emerging solutions for survivable services supported by various networking technologies.

Their technical review focused mainly on OSI Level 2 (e.g., extensions to Ethernet) and did not go above Level 4 (transport, e.g., using SCTP rather than TCP).

B. Technologies Being Considered for the Power Grid

Two main kinds of distributed systems technologies are being considered for widespread deployment in the grid. In fact there seems to be a rush to deploy them, perhaps due to vendor marketing, but also because they are considered best practices in business systems (and reasonably so). In our opinion (and in that of other computer scientists), deploying such technologies in a critical infrastructure for which they were not designed, and without carefully considering which QoS requirements (including reliability-oriented ones) can be met (or cannot possibly be satisfied), is highly inadvisable.

Web services are a very popular way to make distributed services available via web browsers. Parameters are encoded in a URL, session identifiers for a service invocation are stored in local cookie files, and the results are sent back in the form of html. Web services also offer remote procedure calls (RPC), a key building block in any distributed system, using XML and HTTP. Unfortunately, implementing an RPC in this way is very inefficient in terms of bandwidth and is much slower than implementing it in a more traditional, direct way, [Bir06], and is much worse at error handling [Bir04]. This inefficiency means that sensor data delivered via web services will inherently have a much higher minimal latency than is necessary, and they will take up more expensive long-distance bandwidth than is necessary. Yet many industries seem to be plunging headlong into adopting web services, thinking they will solve their problems when in fact they may be creating more. As Ken Birman, a leading expert in reliable distributed computing notes in [Bir06] *(emphasis is ours)*:

It doesn't take an oracle to see that the mania for web services, combined with such rampant online threats, contains the seeds of a future debacle. We're poised to put air-traffic control, banking, military command and control, electronic medical records, and other vital systems into the hands of a profoundly insecure, untrustworthy platform cobbled together from complex legacy software components.

To this list one can add the electric power industry. A number of utilities and industry organizations see web services as key technology (see for example [PJM07]). But the fundamental problem with web services is not just their inherent inefficiency and insecurity, as bad as these problems are. Rather, it is that web services are basically "cobbled together" — indeed, the *raison d'être* of web services is to enable any semi-skilled programmer to cobble together a number of distributed services in an arbitrary fashion —so the user generally has little or no control over their internal algorithms, configuration, or performance; or of the tradeoffs made on behalf of them. Thus, the extent to which the software (which may be an arbitrary composition of different web services mechanisms and products) can meet the end-to-end requirements for a critical infrastructure, especially requirements involving quality of service and reliability, is generally impossible to know. Indeed, it is difficult to imagine these frameworks meeting IntelliGrid's stringent availability requirement given in Table 1.

Service-oriented architecture (SOA) is another technical paradigm being touted for the power grid's communications, as a way to link different power grid applications. While no widely-agreed upon definition of an SOA exists [Wik07], the basic idea is to try to package distributed resources in a way that they are loosely coupled but reusable. SOA does not even prescribe a particular underlying communications mechanism, and can be implemented on top of a wide range of technologies (including web services, CORBA, and other middleware). The fundamental problem with using SOA (at least as represented in the state of the practice for business systems today) for a critical infrastructure is the same as that for web services: you simply don't know what you will get with respect to QoS, reliability, etc.

Ironically, GridStat, which we describe next, can be considered a SOA for grid communications in that it lets many different applications plug and play with a variety of communications patterns and QoS requirements, rather than many applications re-inventing the wheel with respect to communication services. However, as we discuss below, GridStat offers a high degree of control over its infrastructure and of the QoS it delivers, unlike SOAs. Further, GridStat could implement a web services API at the "edges" yet use its controllable mechanisms to offer more predictable latencies and robustness.

5. BASELINE GRIDSTAT ARCHITECTURE

To support the emerging needs of the electric power system for a flexible communication system, we have designed the GridStat middleware to manage network resources to achieve low-latency, reliable delivery of information produced anywhere on the network and sent to multiple other points, *i.e.* GridStat provides QoS-managed multicast. Using the middleware approach allows data providers and application developers to avoid the thornier issues involved in wide-area communication and to concentrate on application needs. Middleware handles the issues concerning different networking technologies, operating systems, programming languages, device types, *etc.* for them. GridStat was designed to meet the flexibility and QoS requirements outlined above, and ongoing work is addressing other miscellaneous (security and trust) requirements.

In the GridStat middleware architecture, network management and data delivery are handled by separate but interacting subsystems called *planes*. The *management plane* allocates resources and adapts the network in reaction to changing power system configurations or communication network failures. *Data plane* components are responsible for forwarding data from each source to potentially many destinations as directed by management-plane components.

GridStat's data plane supports a *publish-subscribe* communication model. Thus, items in each periodic data stream produced at a source (publisher) are distributed to destinations (subscribers) without the publisher having to explicitly track all the subscribing entities. This simplifies application programs and gives the system the flexibility to add subscribers and even change the characteristics of existing subscriptions at runtime. A publisher simply announces the availability of its data stream(s) to the management plane. Subscribers then request that the middleware set up delivery paths (with QoS, including rate, latency and redundancy) to accomplish the delivery.

GridStat middleware components at the publishers and subscribers provide programming interfaces (APIs) that applications use to make the announcements and requests. Similarly, the publisher and subscriber middleware APIs allow applications to send and receive data items. Publish-subscribe systems are sometimes called a *message bus*, and can be considered a *distributed data bus*.

The basic architecture of GridStat is illustrated in Figure 1. The active components of the management plane are called GridStat *QoS brokers* and those of the data plane are called *status routers.* The QoS brokers are hierarchically arranged with policies set at higher levels in the hierarchy controlling more global aspects and allowing local concerns to be implemented using policies at lower levels. The hierarchical organization was designed to support decomposition of communication management along geographical or organizational boundaries. For example, the hierarchy enables utilities to have policies about bandwidth allocation for different applications within the company and to limit internal and external subscriptions to status variables. These policies can also be set at levels below the utility (regions of a utility) and above it (*e.g.*, bandwidth allocated to different entities' subscriptions on an RTO's exchange network).

In contrast to the hierarchy of the management plane, the status routers have a flat organization. A collection of status routers managed by the same communication and cyber-security policy and in the same administrative domain (or sub-domain) is called a *cloud* in the data plane. A cloud is directly managed by a QoS broker at the lowest level in the hierarchy, which we call a *leaf QoS broker*. The status routers forward each incoming data item on outgoing communication links that have a downstream subscriber for that item. Communication links between clouds are managed by the least-common-ancestor QoS broker of the brokers that manage the individual clouds, based on the policies in place at that QoS broker.

In announcing publications and requesting subscriptions, publishers and subscribers interact with a

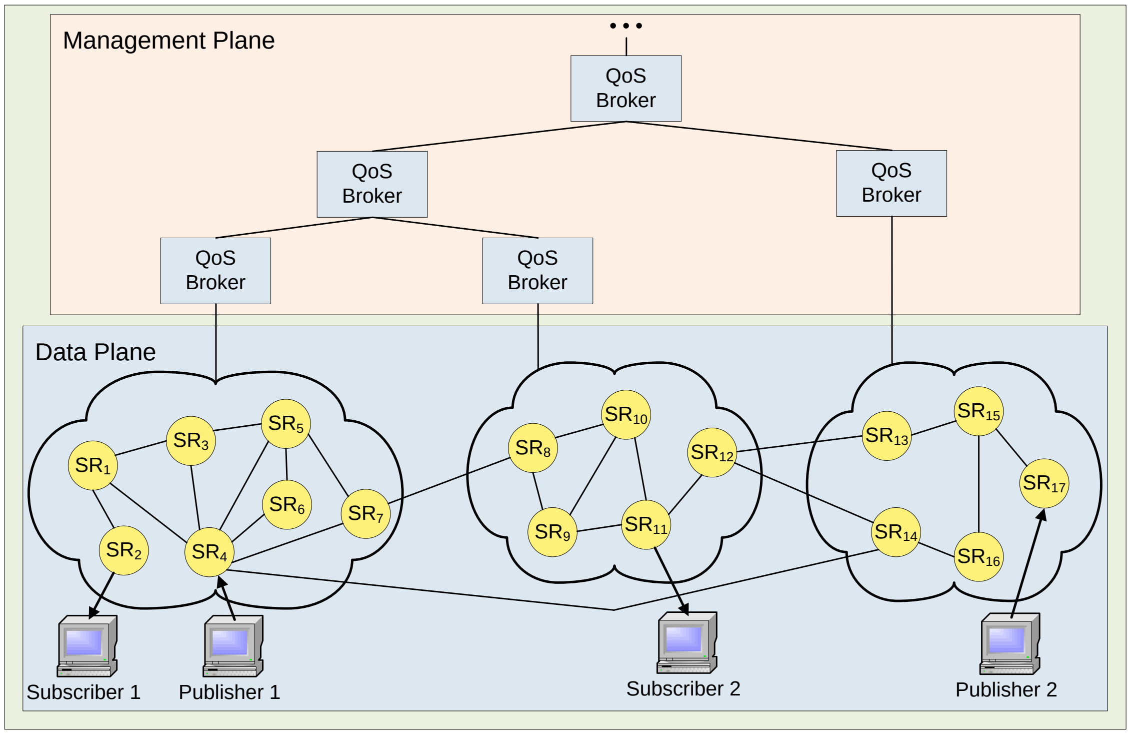

Fig. 1: GridStat Architecture

QoS broker using the middleware APIs. For a publication, the QoS broker simply notes that a publication is available and its publication rate. Subscriptions are more complicated: after verifying that a subscription request can be satisfied within the available resources, the QoS brokers communicate with the management interfaces of the status routers along the data delivery path to set up forwarding rules. Since subscribers can request *lower* update rates than publishers publish, and different subscribers of a given status variable can request delivery at different rates, some items may not need to be forwarded. Therefore, status routers implement *rate filtering* as part of their forwarding mechanisms; this supports the requirement for heterogeneity of delivery rate. Conversely, since a subscription may require use of redundant paths, even a single subscription may cause an item to be forwarded on multiple links. Thus, spatial redundancy is efficiently supported: each update message only traverses any network link at most once. Utilizing spatial redundancy, specifically where all redundant links are used actively (instead of as backups), provides essentially zero fail-over time when a link fails, compared with best practices failover time in a LAN “10 to 50 ms or greater” as reported in [SHS07].

Note that in an actual wide-area deployment of GridStat, the status routers and network links would be engineered such that different kinds of communication media were used, with very different properties (error rate, bandwidth, latency, cyber security vulnerabilities, susceptibility to jamming, etc). Further, they could be obtained from different providers: backbone Internet providers, cable modem ISPs, cellular phone providers (for some low-bandwidth, high importance traffic), broadband over power lines, etc. Thus, updates from a single publisher could be made to traverse different network technologies and different providers (and possibly different operating systems). This diversity has the potential to greatly enhance the reliability of data delivery, in part because the different paths from publisher to subscriber can potentially be much less likely to suffer from common mode failures, for example due to a single vulnerability in a single instance of technology (such as WiFi or Linux).

6. Design of the Principal GridStat Data Plane Entities

As previously mentioned, the principal GridStat entities are *publishers, subscribers, status routers,* and *QoS brokers.* QoS brokers are the main component of the management plane and do not participate in data forwarding at all. Publishers, subscribers and status routers are all data plane components. Understanding the performance, cyber-security and reliability of the GridStat framework requires an understanding of each of these entities and the interactions between them.

There are two types of interactions: *command interactions* and *forwarding interactions.* Data plane entities support both types of interaction and management plane entities support only command interactions.

Command interactions are implemented by *command modules* using CORBA client-server technology. Java RMI and Microsoft’s .NET would be appropriate alternatives to CORBA – the important thing is to leverage a distributed system technology base to achieve interoperability between implementations using different languages and running on different operating systems.

Forwarding interactions are conducted in GridStat using an abstraction called an *event channel,* depicted by the lines connected to the status routers in Figure 1. Each GridStat event channel represents a point-to-point communication capability between the entities that it connects. The event channel abstraction provides a great deal of flexibility concerning the underlying network that carries data between two entities: it could be an actual point-to-point link, or an IP network, or an ATM network, or a SONET ring, for example. GridStat was designed to be implemented on top of all of these kinds of networks. This is very important, because practical wide-area deployment GridStat would be likely to involve data paths crossing multiple network technologies and involving multiple utilities. Further, in practice it will be necessary to provide QoS guarantees such as latency that span these different network technologies.

The GridStat event channel is a building block for QoS-aware point-to-multipoint (multicast) communication which, in GridStat, has a distributed implementation provided collectively by the status routers and management plane. Of course, providing QoS in GridStat involves receiving certain QoS guarantees from the underlying network. For example, the public Internet currently does not support QoS guarantees yet a private internet, using the same protocols but carrying only GridStat traffic, could provide adequate QoS guarantees.

The data plane is structured as follows: each publisher is connected by an event channel to a status router. The event channel carries data items for all the publications made by the publisher. Similarly, a subscriber is connected to a status router by an event channel that carries data items for all the subscriber's subscriptions. Between the publisher and subscriber there is at least one delivery path through status routers. Each status router has potentially many incoming and many outgoing event channels that connect it to publishers, subscribers, and other status routers. A status router's job is to forward each incoming item on the outgoing event channels where it is needed. The forwarding rules are determined by the management plane when subscriptions are added, and then installed in the status router using a command interaction.

As in any communication network, the delivery latency of a packet across a link is made up of several components:

Transmission time, T=(L/b), the length of the packet (L) divided by the link bandwidth (b);

Propagation delay, P=(D/c), the length of the link (D) divided by the speed of light in the particular communication medium (c). These are fixed by the packet length, network technology and distance between nodes;

Processing time, C, and

Queuing delay, Q; both C and Q are determined by the performance of the routing hardware and software and the offered load.

Thus, per-link latency is T+P+C+Q. In GridStat, it is the end-to-end latency that is of interest so these quantities must be summed over all the status routers and links that a packet traverses.

The GridStat prototype implementation provides a baseline for evaluating the forwarding performance of status routers, primarily C and Q in the latency expression. C is directly under the control of the status router implementer – it depends on the speed of the routing hardware and the complexity of tasks that are performed on each incoming packet as well as the quality of the implementation of those tasks. The queuing delay incurred by a particular packet is also potentially under the control of the routing software but the guarantees that are possible are ultimately affected by the total offered load for each link. In order to limit the effect of queuing delays to those that are necessary, status routers implement rate filtering as follows: incoming packets that are not needed to meet the rate requirements of any downstream subscriber are dropped. There is a tradeoff involved in this, however: performing the computation to implement this feature increases C for packets that are forwarded so the relative contributions to latency of additional network traffic and additional computation must be evaluated.

A. Design of the Publisher and the Subscriber

The GridStat publisher and subscriber are implemented in software shared libraries that can be linked with applications. The libraries implement the communication steps that are needed to establish and maintain a connection with a status router, including reconnecting when communication is lost. Library implementations are available in Java and C# (the favored language of Microsoft's .NET middleware framework). They have been designed so that publisher and subscriber applications can exist on a wide range of devices, ranging from embedded devices of limited computational power to high-powered computers running modeling and simulation applications running in control centers. It is important to note here that "subscriber" and "publisher" are merely roles that an application program or even hardware device may play and that any application or device may play either or both roles with respect to many publications. For an application, publishing is simply a matter of asserting the intent to publish, stating the intended publishing rate, then periodically calling the Publish method.

Subscribing involves identifying a particular status variable along with the desired update rate, latency, and redundancy. When it receives this request the status router forwards it to a QoS broker which arranges, using management plane facilities, the needed forwarding rules in the network of status routers. Neither the application, the Publisher, nor the Subscriber need be aware of the communication network topology (the number and location of status routers and the communication links between them). Routing is handled entirely by the management plane. Additionally, GridStat employs *static routing*, meaning that the route(s) from a publisher to subscriber go through the same set of status routers (unless there is a failure). This is more efficient than *dynamic routing*, for example used in the Internet Protocol (IP). It is necessary for IP to employ dynamic routing due to the changing set of computers on the internet (and the fact that no one computer can know all IP addresses). In a critical infrastructure such as the electric power grid, the communication infrastructure is much more fixed and thus static routing is a reasonable approach and is generally more efficient than dynamic routing. GridStat currently will reconnect status routers if a link fails, but we have not yet integrated a recalculation of routes. A naïve version of this would simply use the existing routing facilities of the management plane on a per-subscription basis, but it is future research to determine better ways of doing this more quickly and efficiently at a high level, taking into account many factors including the type and number of subscriptions and the criticality of the power applications they are supporting.

When the setup is complete the application can receive the most recently received status item for the subscription using either (or both) of two different interfaces. The *cache API* lets the application program treat the subscribed status variable as if it were a local variable. This frees the application from concerning itself with directly handling variable updates. The *callback API* lets the subscriber register a callback so that it is notified of updates as they arrive. This is useful, for example, for integration into a local database. Additionally, the application may request notification of QoS violations – for example, if an expected update is late or doesn't arrive.

B. Design of the Status Router

The purpose of the collection of status routers is to provide a message bus between publishers and subscribers that is specialized for forwarding streams of periodically-generated update messages.

Compared with an IP router, a status router is specialized to provide multicast, rate filtering, and subscription modes (defined below) while supporting a range of latency, rate, and redundancy requirements, even for the same data item. These specializations are all based on the observation that resource allocation and routing decisions for a large status network are made infrequently, on the basis of subscriptions, rather than packet-by-packet as in an IP router.

Subscription modes are a feature of GridStat's status routers which allows the status router network to adapt quickly to changing operational situations. A mode contains the forwarding rules corresponding to a set of subscriptions. A mode change globally changes the set of subscriptions quickly without the need to recalculate routes, allocate resources, *etc.* Because such management calculations are complex, they take a significant amount of time for each created subscription. Adding subscriptions *during* a crisis or contingency, while possible, could result in unsatisfactory delays in providing relevant data to subscribers. Subscription modes get around this problem by allowing pre-contingency planning for communication needs in addition to the required practice of pre-contingency planning for responses in the power network.

Multicast has been mentioned previously as a key requirement for GridStat. It provides the property that *any entity with a legitimate need for data should be able to receive it in a timely fashion* while limiting the resources needed to support the property. Spatial redundancy—provisioning multiple disjoint paths from publishers to subscribers—is indirectly implemented by the status routers but no explicit mechanism is needed for this; it is simply a consequence of appropriate forwarding rules being established in the status routers based on routing computations performed by the QoS Brokers, [Ira06]. GridStat implements the requirement for supporting heterogeneous rate delivery with *rate filtering* mechanisms in the status router.

Multicast and rate filtering interact and are implemented by a single forwarding mechanism that works as follows. The status variable ID and timestamp are extracted from each incoming packet. The ID is used as a key to look up the outgoing links with active subscriptions for that ID. The lookup yields for each link a list of subscription intervals. A calculation based on the interval and timestamp yields a *forward* or *do not forward* decision for that link. If any of the subscriptions on a link produce the decision *forward* then the packet is sent on the link; otherwise, it is dropped—there is currently no need for it downstream.

Since one potential use of the status dissemination network is to support applications using synchronous phasor measurements there is a subtlety in rate filtering for synchronous phasor measurements: a large part of their benefit comes from the fact that, being taken simultaneously throughout the power grid, they can be compared to compute, for example, voltage angles. Rate filtering is necessary for PMU data because these devices can produce 1–4 readings per power cycle, yet data can only be reasonably used (at least today) at one or two orders of magnitude lower rate. Without rate filtering, many status updates from PMUs (and other sources) would be wasting significant bandwidth. It would be most unfortunate, however, if the rate filtering were to filter update streams from different sources differently—delivering, say, 4 updates per second from one stream timestamped at 0, 250, 500, and 750ms past the second while delivering updates from another stream timestamped at 125, 375, 725, and 875ms past the second. GridStat's rate filtering is designed so that subscriptions with identical rate requirements for status variables with compatible publication intervals result in identical timestamps on the delivered updates for all subscriptions, [Joh05, JHGB06, Gje06].

We call this property *temporal synchronism.* It means that a state measurement program can collect PMU data from widespread locations and be assured that, despite rate filtering, the measurements that arrive from each PMU are taken at the same GPS time, for example all at 0, 250, 500, and 750 ms past the second.

C. Implementation

The GridStat prototype consists of Java implementations of all of the entities mentioned above. In addition there are C# publisher and subscriber implementations that interoperate with the Java status router implementations.

The next section reports performance measurements made on the prototype implementation. It should be noted that Java imposes a number of performance consequences on the system that affect the reported results:

- Interpretation overhead for virtual machine byte codes compared with native code;
- Just-in-time (JIT) compilation overhead for converting byte codes to native code
- Garbage collection overhead

As a result, we can be confident that the performance reported here (which is based on early 2004 hardware) is a lower bound on what the architecture is capable of achieving. Even with the Java implementation it is clear that performance requirements of many current and envisioned power system monitoring and control applications given in section 3 are within the capabilities of the prototype, as we describe below in section 7.C.

7. EXPERIMENTAL RESULTS

We now overview the results of extensive performance experiments on the GridStat prototype. For more details, see [Gje06], which includes 74 pages of detailed experimental results.

There are two primary metrics of interest in this overview. The first, *forwarding latency*, is the amount of time it takes an SR to forward an update message. The second, *load scalability*, is the number of update forwards per second which can be sustained without degrading forwarding latency. Other metrics such as the *drop rate*, the fraction of forwarding updates dropped by an SR, were measured extensively but are not presented here in detail for reasons of brevity.

The experiments were conducted in a rack-mounted system whose main computers are 19 Dell Power Edge 1750s with a Dual Xeon 2.8 GHz CPU, 533 MHz system bus, 1 GB of DRAM, and a triple 1Gb/s Ethernet interface connected by switched Ethernet. The system software included Fedora 2 Linux (kernel 2.6.10-1.9_FC smp) and Java 2 Platform Standard Edition 5.0 (version 1.5.02) from Sun Microsystems, using UDP sockets The packets that are read/written from/to the UDP sockets are stored in a buffer pool of ByteArrays which are allocated using the `allocateDirect` factory method. This factory allocates the memory for the buffers directly from the OS heap as to avoid the copying of the contents of the buffers between the JVM heap and the OS heap, when doing I/O calls to interact with the UDP sockets..

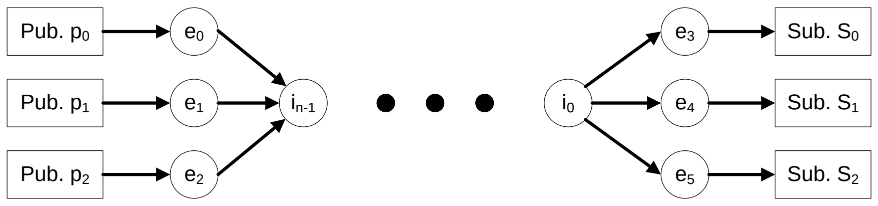

Fig. 2: Baseline Experimental Topology

A. *Baseline Forwarding Experiments*

The experimental topology for the baseline forwarding experiments is given in Figure 2. In our experiments, the performance metrics are measured on a *reference system*, a single publisher p_0 and a single subscriber s_0. These subscription endpoints are on the same computer so the end-to-end latencies can be measured very accurately using the local clock. Additionally, there are two *load systems*, which consist of two pairs of publishers and subscribers, (p_1,s_1) and (p_2,s_2). Each publisher and subscriber is connected to an *edge status router (ESR)*; these are denoted e_i. In these experiments, the reference system and the load systems share a chain of *n* SRs, *backbone status routers (BSRs)*, denoted i_0 through i_{n-1}.

Figure 3 depicts the forwarding latency and load scalability observed in our experiments. Here, the number of BSRs, *n*, is varied from 1 to 7, and both the forwarding latency and its standard deviation increase close to linearly with *n*. With a load of approx. 18K updates/sec, the forwarding latency for each BSR (derived from the slope from 1 to 7 BSRs) is 0.360 msec. When the load is approx. 50K updates/sec, the forwarding latency for each BSR is 0.532 msec. We do not present results beyond approx. 50K updates/sec, because at this point the BSR's CPU gets overloaded, causing the forwarding latency increases more than linearly and the drop rate (well below 1% at lower rates) increases drastically.

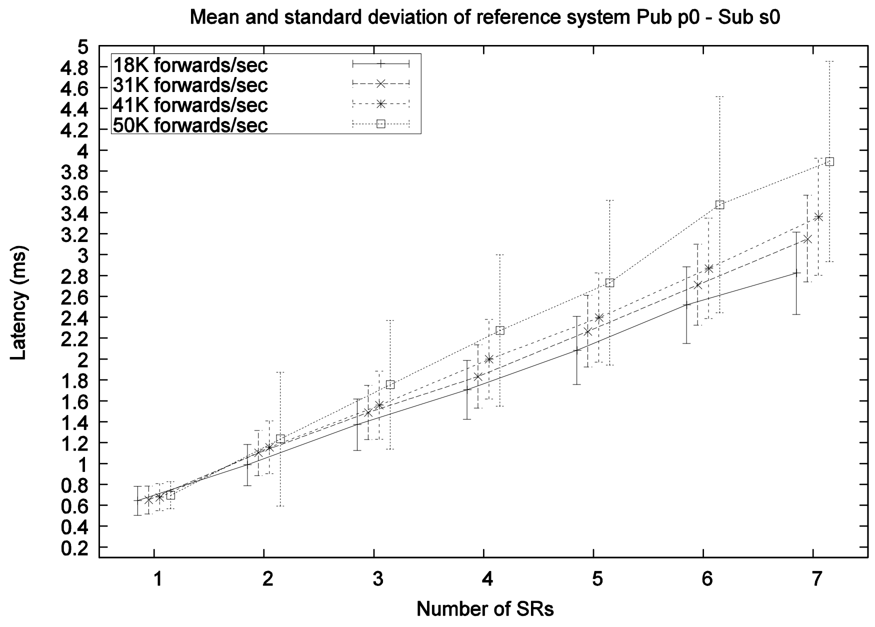

Fig. 3: Baseline Forwarding Latency and Load Scalability

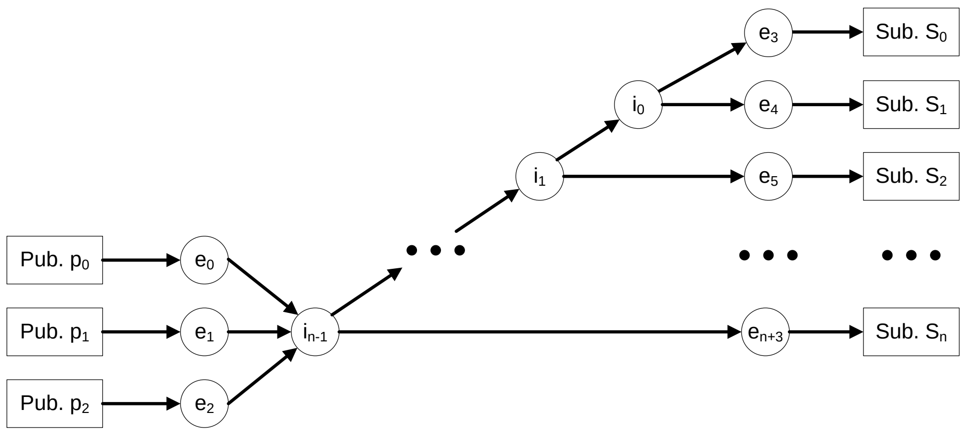

Figure 4: Multicast Experimental Topology

B. Multicast Forwarding Experiments

The previous experiments measured costs associated with forwarding update events through a linear chain of BSRs, i.e., a point-to-point communication. In this experiment, we generalize this to measure multicast performance, where BSRs forward each update to multiple outgoing destinations. The experimental topology for these experiments is given in Figure 4. The BSR i_0 through i_{n-1} each has two outgoing communication links. Subscribers S_1 through S_n subscribe to all variables published by both p_1 and p_2. As in the baseline experiment publisher p_0 and subscriber s_0 constitute the reference system.

Figure 5 depicts the forwarding latency and load scalability observed in our experiments. Here, the number of BSRs, *n*, varies from 1 to 4, and both the forwarding latency and its standard deviation

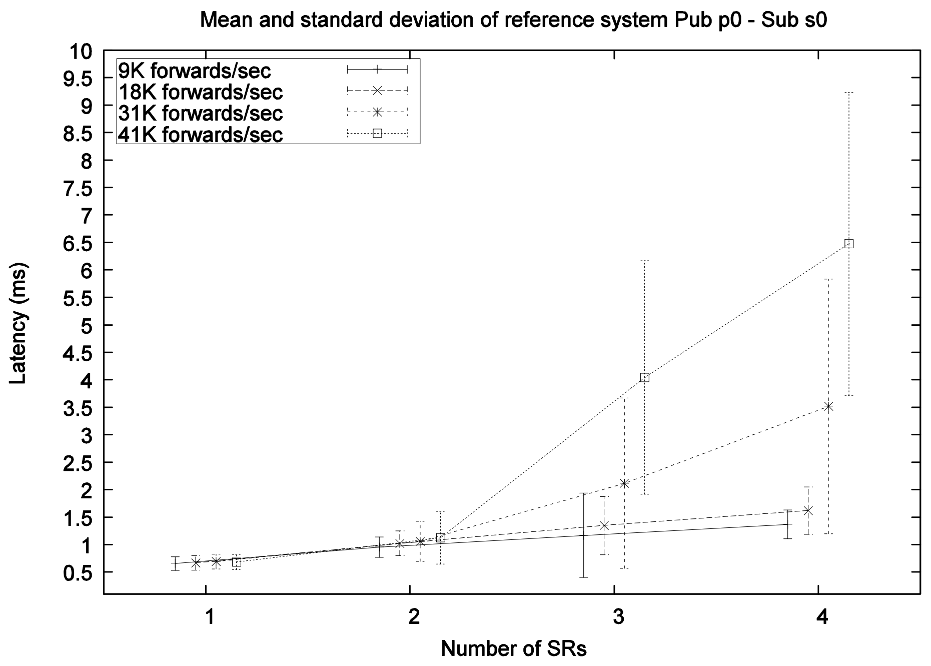

Fig. 5: Multicast Forwarding Latency and Load Scalability

increase close to linearly with n for a load of 9K and 18K updates/sec. With a load of approx. 18K updates/sec, the forwarding latency for each BSR (derived from the slope from 1 to 4 BSRs) is 0.317 msec. When the load is approx. 41K updates/sec, the forwarding latency for each BSR is 1.932 msec. For this experiment the CPUs got overloaded with a lighter forwarding load (due to the multicast) so the 50K updates/sec is not presented. Note that for n=1 no multicast occurs; it only occurs when n > 1. The sharp increase in the end-to-end latency for the 31K and 41K updates/sec can be explained by the CPU load on i_1 through i_{n-1} for these experiment runs. Due to a limitation of a Java library class the garbage collector is extensively executed for these runs (for more detail see [Gje06]) . We expect this sharp increase to be greatly reduced in the C language version of the status routers that we are presently building.

C. Discussion

The GridStat prototype demonstrates that the flexibility and QoS management needed for a next-generation SCADA replacement is feasible on off-the-shelf commodity computers and portable software (Java code). However, much better performance is achievable (these numbers are conservative), for many reasons. First, the computers used in the experiments were state-of-the-art as of early 2004, but today even a mid-level PC is considerably faster, and as of 2007 have quad cores (4 processors). The forwarding latency reported above also includes Ethernet latency through the switch to the next SR. Second, using Java incurs overheads associated with byte-code interpretation and just-in-time compilation as well as delays due to garbage collection. We are starting a second version of the SR prototype in C, that will eliminate these overheads and hence be faster. It is expected to be finished in mid-summer 2007

For today’s applications, handling a sustained load of 50K updates/sec in a small or medium-sized substation’s communication processor is likely to be sufficient. Additionally, this forwarding latency of approx 0.5 msec is also adequate for many situations, including those given in the IEEE 1646 specification and the QoS requirements outlined in Section 3. For example, a path spanning a control area 800 km across might go through 5-10 SRs, for a cumulative latency of approx. 2.5–5.0 msec in addition to the propagation time(3.9 msec for this distance). Thus, not counting latency in the destination application program, overall latency of approx. 6.4–8.9 msec is achievable. Thus, GridStat allows wide area communication for new kinds of remote protection or control at a latency that is a small fraction of the 3 to 4 AC cycles typically required for locally triggered breaker opening today.

This performance also opens up new possibilities for remote protection and a new class of controls to contain cascading, even with the above numbers on Java and 2004 vintage hardware. [TSP98] predicts that disturbances travel at 500 km/sec in an example system with 64 generators, and different kinds of fast controls such as transient stability controls typically need to react in as little as 100 msec and as much as 1–2 seconds. GridStat can in realistic situations deliver data much faster than electrical disturbances, but much research would be required to develop best practices on how and when such remote schemes would be beneficial.

The above experimental results were conducted using general purpose computers. Network processors are another hardware platform that could be used for status routers, [Com03]. A typical network processor features a single, general purpose CPU and a number of specialized packet processors called MicroEngines—specialized CPUs that are optimized for network protocols. They can perform many kinds of activities related to packet forwarding, such as computing UDP checksums, in a single instruction cycle, and support a high degree of internal parallelism. A network processor interfaced to several Ethernet and fiber inputs can do IP forwarding between them at line speeds (1-10 Gbit/sec or more).

We are presently wrapping up an initial investigation into how fast GridStat routing can be performed on a network processor [Swe07]. The main goal of this project is to see how fast GridStat-style routing can be on network processors rather than to implement the full functionality of today's Java-based status router. The initial experimental results were conducted on Intel's simulator configured to mimic the Radisys ENP2611 PCI card. The ENP2611 integrates an Intel IXP2400 network processor chip having 8 packet processors, 4 network connections (1 at 100 Mbps, 3 at 1 Gbps), and buffer and program memory on a PCI card that plugs into a host computer. The simulator is a full implementation of the IXP2400 chip that also faithfully simulates the externals such as the network links and the various on-board memories. The simulator runs the same binary code as the hardware itself and makes it possible to obtain exact cycle counts and instruction traces for status router activities.

The ENP2611 is capable of performing a simplified GridStat routing algorithm, supporting only subscription periods that are a power of 2 milliseconds, at 1Gb/s line speeds[2]. This corresponds to approximately 2 million GridStat update forwards per second per output link, using the same packet size (62 bytes) as the Java experiments reported above. Measured forwarding delays ranging from 1–3 microseconds have been obtained on different experiments.

These numbers suggest that the overhead of GridStat could be negligible: in any wide area deployment the forwarding latency added by GridStat would be much smaller than the propagation time (i.e., the speed of light over the distances involved) and the application program latencies at the endpoints. The throughput is also very high. It is important to remember, however, that this prototype does not yet implement the full functionality of GridStat's Java status routers. More research is needed to evaluate ways of providing complete GridStat functionality on newer network processors, including finding ways to provide the temporal synchronism needed by PMU data streams.

Finally, in this context we feel it is important to comment on a common practice in the power grid today. Often sensor data are stored in a database, and then different applications retrieve them from there either in a *two-tiered* client-server architecture, where an application directly calls a database, or a *three-tiered architecture* where the client calls a server which then accesses a database server. An example from a water and power utility project supporting both two tiered and three-tiered access is [MRW+07].

We believe that this cannot meet the requirements of evolving distributed power applications. Putting a database on the critical path for data delivery adds unnecessary latency, so data are less fresh when they finally reach the applications that can act on them. The latency arises both from the time required to store and retrieve data from the database and from the need for a full communication roundtrip for the application to request the data and for them to be delivered. With a publish-subscribe system such as GridStat, data are pushed out to the applications as soon as they are available so only a one-way network latency is incurred. Further, it would be fairly straightforward to add to GridStat some simple mechanisms so that subscribers could indicate that they only want updates when a sensor value has changed by a certain amount or percentage, instead

[2] The restriction to powers of 2 eliminates the need to divide by arbitrary numbers, an operation that is *very* expensive on this hardware.

of updates being sent solely based on the passage of time. These *value-triggered update mechanisms* have the potential to save an enormous amount of bandwidth. Research would be required to understand how they can be made efficient and scalable, how they would affect subscriber applications, and what their behavior and bandwidth usage would be during a power disturbance (when sensor values are changing rapidly). With the results of this research, it would be possible to ascertain the circumstances under which value-triggered updates would be worthwhile.

We note that value-triggered updates are an example of the kind of feature that can only be built the semantics of a flow of status updates and requirements of the subscribers are understood. They are not feasible with a generic publish-subscribe framework. In the next section, we describe higher-level capabilities that can be created with such an infrastructure.

8. ADVANCED GRIDSTAT MECHANISMS AND FEATURES

In the previous sections, we have described the design, implementation, and experimental evaluation of GridStat's baseline mechanisms. We now overview some of GridStat's advanced mechanisms and features that support the flexibility requirements described in Section 3.B. For more details, see [Gje06].

In order to mask temporary omission failures in the network, such as a short burst of packet losses, GridStat provides an *extrapolation mechanism.* The extrapolation is performed in the subscriber library. Several extrapolation algorithms are included in the library. In addition, a subscriber application can provide a custom extrapolation algorithm built from a base class that GridStat provides. Each subscriber can use a different extrapolation function and parameterization, or none at all. Once initialized, extrapolation is transparent to the subscriber application; *i.e.* the application uses the cache API or the callback API just as if there were no extrapolation present. Extrapolation must only be used after careful analysis of the behavior of the variable in question, only for very short durations, and only in steady state, when there is not a power anomaly underway. It can, however, be useful when there is an IT anomaly underway (either router failures or a cyber-attack) so long as it does not affect power dynamics. The variables for which extrapolation can be used, the maximum duration, and under what conditions is an open research question.

Many power application programs need to perform a simple calculation, or a few of them, over a large array of data items, often from a single source. Sending those data items to all entities which need the results of the calculation can consume a large amount of bandwidth. GridStat therefore provides a *condensation mechanism*, a kind of application-specific plug-in, that can subscribe to a large set of status variables and/or alerts, perform an application-defined calculation over them, and publish the result as a new status variable and/or alert. This can push data analysis functionality closer to the source of the data to be analyzed (for example, in the same substation), not only saving bandwidth and redundant processing costs downstream, but also potentially allowing for a quicker response. GridStat includes a graphical tool for designing condensation functions. It lets the designer set threshold filters; trigger the calculation based on time, a status update or an alert; define the algorithm for the calculation; and define an output threshold filter. All but the calculation algorithm are optional. Condensation mechanisms can be used not only for operational data but also could be used for alarm processing, to help identify the root-cause event, for example with techniques such as the SuperCalibrator , which provides a model-based error correction of substation data, [MCG+07].

GridStat also supports *limited flooding*. This allows an alert to be sent to all status routers in all clouds at or below a given level in the management hierarchy. The flooding mechanisms allow an alert to bypass the normal routing mechanisms and get delivered to all entities in a given part of the power grid with very low latency and very high resilience. It would be possible to limit the amount of bandwidth and status router computing cycles that alerts can consume by doing bandwidth policing at the sources of alerts, and also by aggregating alerts based on their category or sub-category; we plan to implement such mechanisms in the future.

A subscriber can subscribe not only to the direct value of status items, but also to *derived values*. These include the moving average, maximum or minimum value over the last *x* seconds, rate of change, *etc.* Derived values can be useful in a number of circumstances, including cases where sharing the direct value would be market-sensitive.

Triggers can be constructed from the condensation mechanism. These can be used, for example, to generate an alert under certain conditions, such as when a value (or derived value) crosses a threshold. Many EMS systems offer triggers, but if they are pushed into the communications system then they can be used by many applications without having to re-invent the wheel.

9. Higher-Level Capabilities that GridStat Enables

A very flexible and robust communication framework such as GridStat enables a number of higher-level capabilities. In this section we outline three examples of them to help motivate why more flexible communications are required. Many other applications are possible, including a host of new ancillary services exploiting interactions between entities—something that is not feasible with today’s grid communication system. Note also that the examples below are possible even when cooperating utilities are using different EMS systems, they just would have to be able to plug into GridStat (which is vendor neutral and can accommodate any conceivable underlying network technology).

A. Data Load Shedding to Prevent Data Blackouts

Electric utilities can do *load shedding*, which for the sake of clarity we will call *power load shedding*. When they are nearing a blackout, they can shut off power to certain customers; they try to do so with customers with whom they have a load shedding agreement. Load shedding can help avert a blackout, but it is an extreme measure, so utilities are hesitant to employ it unless absolutely necessary.

GridStat enables load shedding in the IT domain. When a subscriber asks for a subscription, it must specify its required rate, latency, and number of redundant paths. But the GridStat subscription API requires the subscriber to provide these parameters for both best-case values that the application hopes for, and also worst case values that it can live with. During normal operations, subscribers are given the best case delivery service. This provides better fidelity in the sensor values for the application and also provides additional fault tolerance: if a few updates are dropped the worst case requirement is likely still met. However, in the face of an IT anomaly such as an IT failure or a cyber attack, *data load shedding* can be done: the operational mode can be switched to throttle back less important status update flows to their worst case level. This can quickly free up IT capacity to enable effective responses to the IT anomaly. Additionally, GridStat could relatively easily be extended with a priority for each subscription. Then, less-important flows could be throttled back more aggressively so that more important flows would still get much more than their worst case requirement.

Note that, unlike power load shedding, data load shedding is not disruptive to customers or to anyone else: the subscribers still receive a usable quality of status delivery (as defined by the application in question: its worst-case it can live with), albeit without the additional fidelity and fault tolerance that the best case delivery provides. Interestingly, data load shedding is strongly analogous to demand response, in that both provide a form of demand elasticity, only on the IT network, not the electricity network. Research is needed to study how to use these capabilities in a range of disturbances, including their affects on different power applications.

B. Integrated IT-Power Contingency Planning and Adaptation

As noted earlier, GridStat supports subscription modes. These allow GridStat's status routers to switch routing tables; in effect, to switch between one bundle of subscriptions (including QoS requirements and number of redundant paths) to another bundle almost instantly. This is very helpful to avoid a barrage of requests for more subscriptions, because the calculation of the paths for each subscription is very computationally intensive.

Modes allow utilities to plan in advance for the communication reconfiguration to undertake for contingencies just as they now plan the electrical reconfiguration. Applied R&D could extend the state of the art and then best practices to include coordinated response including:

1. Power dynamics contingency planning, including operator countermeasures
2. Switching to a mode for that contingency to provide additional data that enables operators to better identify the details and severity of the contingency
3. Automatically open up new windows in the operators' EMS system with these new contingency-specific diagnosis data to help operators quickly see what is happening

This comprehensive (and almost instantaneous) reaction can be done not just for power anomalies, but for a more comprehensive set of anomalies including:

A. Power anomalies (as today)

B. IT failures (failure of a network link or status router)

C. Cyber attacks

The state of the art today includes only items 1 and A above.

Taken as a whole, these more robust actions are not only online, but use more data and are responding to a much broader set of contingencies than is best practices today. This has the potential to make the grid even more robust. Of course, research and experimentation is required to develop these best practices, but the items to be researched are fairly straightforward.

C. Early Warning System without Sensitive Data Disclosure

An obvious benefit of utilities receiving a much broader set of operational data is the ability to detect problems that in today's grid go undetected. For example, in many recent blackouts, some of the early events (such as a transmission line tripping) were not noticed, or were noticed by different entities, none of whom had the entire picture of what was going on. With a more flexible communication system there is the potential to share much more than just large events. Variables that could be early indicators of impending problems could be shared.

Many practical problems would result when trying to set up an early-warning system, however, due to the tension between operational reliability and business interests. Consider the case where engineers from a dozen utilities are constructing an early-warning system. They presumably could come up with a list of many status variables that other utilities could use as early-warning indicators. It is also plausible that these engineers could pick a subset of these status variables which they believed were not market-sensitive. However, it is also plausible that company management would favor treating all such data as confidential. So, in practice, very few of the potentially useful early warning indicators might be used.

A more flexible data delivery system such as GridStat can help in this situation in at least two ways:

1. The actual values of variables might not be published, but rather, derived values such as the instantaneous rate of change (presumably much less sensitive) or a filtered subscription that only delivered data when the value or derivative was outside an expected range.
2. *Alert-triggered temporary subscriptions* could be established. These subscriptions would not normally be active. However, once sufficient early warning signs were detected, subscriptions would automatically be started that involved operational data which were market sensitive. These subscriptions would automatically shut down when the emerging contingency was eliminated. The decision to enable sensitive subscriptions could be made unilaterally by a higher entity (DoE/DHS, an RTO/ISO, etc) or by a real-time vote of the operations centers involved. Or the decision could be made automatically based on pre-arranged policies implemented by condensation functions that subscribed to a set of status variables.

The first technique is possible with GridStat today, and the second technique could be implemented with a modest amount of engineering effort.

10. ONGOING AND FUTURE RESEARCH AND DEPLOYMENT

The first GridStat demo was given in 2002, and it has been used in a technology evaluation demonstration with live data from Avista Utilities since 2003 (see http://gridstat.net/avista.php for more information, including live data) and with a grid frequency measurement device by PNNL since 2006 (see http://gridstat.net/pnnl.php for more information). A technology evaluation demonstration between two national energy labs is projected to start in Summer 2007

The prototype reported on here was completed in 2006. Several ongoing projects are further extending GridStat's capabilities.

GridStat to date only supports the publish-subscribe style of interaction. While this is fine for delivering status updates and alerts, other kinds of interactions in the power grid require other communication styles. One project is building round-trip client-server remote procedure call (RPC) mechanisms on top of GridStat’s QoS-managed one-way publish-subscribe infrastructure. The RPC mechanism can be used, for example, to invoke a command on an actuator and then get a confirming reply, using multiple redundant paths. It can also be used for control commands between control centers. This RPC mechanism allows the programmer to select among several kinds of fault tolerance. It also provides pre- and post-conditions over GridStat status variables. Pre-conditions allow an actuator call to be aborted if conditions make it inadvisable (for example, if a line is energized when it should not be). Post-conditions give the caller additional assurance that the command was carried out by monitoring variables that should change when the actuator command is executed on the device. This project will be finished in mid-2007, details can be found in [Vid07].

Remote Procedure Call is only one example of the kinds of protocols that can be built on top of GridStat. Indeed, GridStat can at some level, be considered for use in the places where many in the power industry are advocating using IP. Of course, GridStat provides much more control over the infrastructure and provides much better performance and fault tolerance than unadorned TCP/IP. The point is that power protocols that can be implemented on top of IP can also be implemented on top of GridStat, which in practice would likely use IP (though not the best-effort Internet) for many of its underlying network links.

The subscription modes described earlier are only implemented within a single cloud (leaf QoS broker). A second project is extending GridStat's mode change mechanisms in two ways. First, it implements global modes, meaning that a mode change can be affected across multiple clouds, possibly involving all status routers across a power grid. Second, modes can be hierarchical, meaning that a mode can pertain to a sub-tree of the QoS broker hierarchy. A mode could apply to, for example, only the scope of a single utility or ISO. With hierarchical modes, multiple modes can be in effect in a status router at once, yielding a coarse way to provision resources within the scope of a mode. We have developed two different mode change algorithms to support global and hierarchical modes. They provide different tradeoffs between consistency and performance. This project will also be finished in mid-2007; details can be found in [Abe07].

GridStat efforts to date have been focused on defining what status variable delivery could and should be, and has not yet focused on providing cyber security. A third ongoing project is extending the data plane to provide confidentiality, integrity, and protection from some kinds of malicious attacks. This is being done in a way that can be efficiently implemented using GridStat's multicast and so that security levels can be managed at the granularity of a single flow (a path from a publisher to one subscriber). This project is expected to be completed in late 2007 or early 2008. A complementary project to secure the management plane is expected to be completed in 2008. Additionally, many security technologies for the power grid are being developed as part of a five-year center-scale project, Trustworthy Cyber Infrastructure for the Power Grid (TCIP), funded by the National Science Foundation, the Department of Energy, and the Department of Homeland Security, [TCIP07]. The TCIP center involves not only well-known power engineering researchers (Anjan Bose, George Gross, Pete Sauer, Tom Overbye, Bob Thomas) but also a wide range of computer scientists who are all working on developing technologies to address broad, comprehensive security problems in the power grid (rather than the narrow, point solutions that industry has offered to date). TCIP h as over three years remaining as of May 2007. We anticipate that a number of technologies from this multi-university center (in which GridStat researchers participate) will be integrated into GridStat.

In the next few years we plan to enhance GridStat in many other ways, including a suite of configuration tools, policy languages, internal instrumentation, visualization tools, and other features which will make it much easier to deploy, use, and maintain. We also intend to harden it in wide-area deployments and further investigate the use of network processors with GridStat. We also hope to integrate GridStat with IEC 61850 and explore how the rich configuration information of its CIM (or perhaps modest extensions to it) can be used to help manage the grid's communication network more automatically. While GridStat supports some simple ways to deal with failures, much research is needed on ways of providing dependability at many levels in a realistic GridStat configuration.

To date we have designed, developed, and evaluated GridStat's data and management planes and some advanced mechanisms for them. A big part of what we plan to do in the next few years is to be involved in more pilot projects with utilities to evaluate different ways of using GridStat in a realistic setting. This will almost certainly increase our list of future work with suggestions for better management and development tools, specific additional features, etc. We also hope to extend GridStat to deliver IEC 61850 GOOSE messages; [MD07] notes "further down the path of substation automation is the use of IEC 61850 GOOSE messages over the corporate WAN to other substations". We believe GOOSE over GridStat is very doable and has obvious benefits.

11. Related Work

GridStat is unique in a number of respects: explicit support and management for streams of status updates, integrated rate filtering and multicast mechanisms, integrated QoS and disjoint path routing algorithms, and the fact that it was designed explicitly for the electric power grid.

A. Power Engineering Research, Standards, and Products

There are some related efforts in the power industry. The IEC 61850 specification includes an elaborate and elegant model-driven approach to substation automation, including self-describing substation devices, [Mack06]. It "provides a comprehensive model for how power system devices should organize data in a manner that is consistent across all types and brands of devices." Within its current scope, it seems to have great potential. We believe that GridStat and IEC 61850 are highly complementary: GridStat provides wide-area delivery (and could easily be extended to deliver messages in the standard's GOOSE message format), while IEC 61850 does not, itself, provide any wide-area delivery mechanisms; its focus is on the substation LAN. IEC 61850-capable devices live at the edges of the communication network. There is a consensus in industry that, at the present, more device configuration applications are needed to help exploit the full potential of IEC 61850. These applications can serve as effective tools in deploying devices using this standard. But this exact same kind of device type information could be very useful with tools to help configure and manage a GridStat-like, sophisticated communication infrastructure for the power grid.

An information architecture for the power grid is proposed in [XMV+02]. It contains proposals for different ways to structure interactions between control centers and substations, and reliability analyses of different schemes. However, it does not propose any communications mechanisms, and relies on off-the-shelf network technology whose reliability and latencies are not controllable and which does not meet most of the requirements outlined in Section 3 above.

The EPRI IntelliGrid project provides detailed case studies of many future kinds of interactions in the power grid, including QoS and security requirements at a qualitative level [IntelliGrid04]. However, it does not provide communication mechanisms of any kind. The wide area measurement system (WAMS) uses a pioneering wide-area data network developed by the Bonneville Power Administration [MKO96]. More recently the North American Synchro Phasor Initiative (NASPI) is deploying WAMS-like technology on all the US interconnections. Currently, these projects have to rely on special-purpose networks to carry PMU data for lack of the kind of flexible network described here. NERCNet is another purpose-built network that connects major entities in the power grid and is used, for example, to carry Inter-Control Center Protocol (ICCP) traffic.

There are recent industry solutions hoping to replace or at least supplant SCADA, including some that offer publish-subscribe such as IBM's WebSphere (a generic web services framework) and products by Telvent. While it is extremely difficult to extract any detailed technical information on such products on the web (or by asking sales people at power trade shows), we can make some inferences as to how they compare to GridStat. To the best of our knowledge, none of them offer the semantics of a status update (and hence the manageability and adaptability that comes from this), none of them offer rate filtering mechanisms, and they seem to offer relatively little control over the actual configuration. It is our belief that repackaged generic publish-subscribe systems intended for corporate enterprises are at best unproven technology for wide-area critical systems, while at worst Birman's warnings apply. In fairness, however, we note that web services products such as WebSphere support a much broader range of APIs than GridStat does, including transactional ones.

B. Computer Science Research

In computer science research there has been some work related to GridStat. The closest is PASS, [ZOB+01]. PASS provides a limited form of status dissemination, specifically, binary "up/down" status dissemination for remote devices. It does not provide QoS management, support heterogeneous delivery rates, or provide redundant delivery paths (though is very useful for its intended domain: wide-area military networks in the field with low bandwidth). Other publish-subscribe systems do not provide any kind of rate filtering, because they do not capture the semantics of a status variable flow. TIBCO sells publish-subscribe middleware that is used on stock trading floors to disseminate security trade information. However, it was not designed for a wide-area network spanning a large number of companies, and it does not provide the unique features summarized above. Quality Objects (QuO) is a middleware framework for providing quality of service across a wide area network [ZBS97,QuO06]. It supports multiple layers of adaptation and has been used to integrate technologies from over a dozen research projects and industry technologies. QuO is an example of the kind of QoS management that a well-designed middleware framework can provide. It has integrated multiple QoS subsystems including bandwidth reservation, replication, and multiple security mechanisms, and as such the best example we are aware of in how middleware can provide flexibility and QoS. However, QuO supports the distributed object paradigm rather than a publish-subscribe model (let alone its specialization, periodic streams of status updates).

QuO has a Resource Status Service (RSS) which provides both in-band (along the call path) and out-of-band (external resources) status information to be integrated and modeled and then used for application-level adaptation when wide-area resource conditions change [ZLS02]. RSS was used as a starting point for the MetricsService [ZSS04] of the open source Cougaar distributed agent-based middleware [Cou07], which was initially funded by DARPA but is now maintained by an open source community. RSS is closely integrated with QuO's distributed object model, and MetricsService is closely integrated with Cougaar's component model, which is closely related to Java Beans. However, neither of them was designed to support the delivery of status items and do not support rate heterogeneity or some of the other management features which GridStat has.

Cayuga [DGH+06] is a sophisticated publish-subscribe system based on a nondeterministic finite state automata (NFA) model with an event algebra. It allows the construction of sophisticated triggers that can involve different events (across different status variables, in GridStat terms) and can aggregate in different ways. As such, Cayuga might be an excellent basis for value-triggered updates for GridStat. It could provide the mechanisms for filtering updates based not just on rate (which GridStat does now) but on changes in the value (or even richer predicates across many variables). However, we do not believe Cayuga would be appropriate for the low-latency delivery rates required. While [DGH+06] does not report latency numbers, we believe that they would be significantly higher than those for GridStat.

The use of epidemic multicast algorithms [DGH+87] on the power grid instead of TCP/IP has been investigated in [BCH+05]. Also called “Bimodal Multicast”, [BHO+99], these algorithms offer high and stable throughput for multicast with good resilience. However, they are not intended for some of the ultra-low latency applications that GridStat supports, and do not support some of the flexibility requirements described in Section 3.B. These algorithms could be used in conjunction with GridStat, however, letting them “manage” an additional redundant path. Such a redundant path would not be managed as closely as GridStat’s paths, but in the face of some kinds of IT anomalies this could be an advantage. For example, two GridStat-managed paths and one epidemic-managed path (which plugged into GridStat endpoints) could offer significant benefits over either technique alone: lower achievable latencies than epidemic multicast alone and better failure coverage than GridStat alone.

A discussion of additional research in computer science disciplines that is related to GridStat can be found in [Gje06].

12. CONCLUSIONS

The communication infrastructure in today's power grid is based on the technology of the 1960s. It greatly limits the kinds of protection and control that can be employed and hampers the situation awareness of the grid's operators. The communication infrastructure is being augmented in a piecemeal fashion by newer network technologies such as optical fiber and Ethernet. However, to date there has been no utilization of advances in distributed computing technologies, that offer much more flexibility and adaptability than can be achieved using network-layer technology.

These distributed computing technologies will be crucial to meeting the requirements for an improved power grid communication system. QoS requirements for the grid include providing predictable latency and rate for status data delivery as well as high availability for the data. The communication system must be much more flexible in many ways. Things that are hard-coded, or even hard-wired, in today's grid must be changeable in software (with strong cyber-security, of course). The grid's communication system must also support multicast, different topologies of data dissemination, heterogeneity of delivery latency and delivery rate, and support for heterogeneous operation as well as extensibility across a wide range of computing resources (network technologies, CPU architectures, programming languages, etc.). Furthermore, cyber-security of the power grid's communications infrastructure is crucial.

GridStat is a flexible middleware framework for disseminating status data in the power grid that meets the above requirements for the next-generation communication infrastructure. It is in effect a QoS-managed, robust overlay network. GridStat's data plane delivers data with required latency, rate, and number of redundant paths. It supports temporal synchronism to provide a consistent global snapshot of GPS-timed PMU data. GridStat's hierarchical management plane allocates resources and configures the network to meet the requirements of the data plane's status variable subscriptions. GridStat's forwarding latency on 2004-vintage hardware running Java is on the order of 500 microseconds for a single status router—fast enough to meet latency requirements of many control and protection schemes over at least a hundred miles in typical configurations. Ongoing optimizations are lowering this latency significantly, and network processor technology improves both the latency and throughput by at least two orders of magnitude.

GridStat also features a number of advanced mechanisms to further enhance its robustness, performance, and flexibility. Cache extrapolation can tolerate omission failures of status updates. Condensation mechanisms save bandwidth by computing a calculation over many variables close to their source, and then disseminating the result to many subscribers. Triggers allow alerts to be issued when certain thresholds are crossed; and modes allow routing tables to be switched very fast in order to adapt to a contingency.

With a tightly-managed communications infrastructure such as GridStat that captures the semantics of status variable update flows, a number of higher-level capabilities and adaptations are possible. Contingency planning can be combined with mode changes in order to deliver the right data to fully analyze the contingency. EMS display windows can automatically be opened in a contingency to show that new data to operators as they “drill down” to the cause of the problem. Furthermore, these actions can be employed not only for power contingencies, but also in the case of IT failures or cyber attacks. Since GridStat captures both the desired and worst-case QoS (latency and rate) required by a subscriber, it can employ data load shedding in the face of IT contingencies yet still meet subscribers' requirements. Data sharing for early detection of emerging contingencies can be accommodated without continual divulgence of sensitive data by publishing derived values or by alert-triggered temporary subscriptions.

GridStat is complementary to IEC 61850, IntelliGrid, and other industry efforts, none of which provide mechanisms for flexible, wide-area data delivery with QoS and robustness.

ACKNOWLEDGEMENTS

We thank Stian Abelsen, Dave Anderson, Beth Bakken, Ginny Hauser, Loren Hoffman, Nathan Schubkegel, Kim Swenson, and Erlend Viddal for their comments on drafts of this report, and Nathan Schubkegel for his artwork for some of the figures. Many colleagues have provided insightful feedback on GridStat over the last seven years. In particular, we thank Roberto Baldoni, Sandro Bologna, Sudipto Bhowmik, Jeff Dagle, Deb Frincke, Bob Hill, Helen Gill, Don Kopczynski, Mats Larsson, Klara Nahrstedt, Tom Overbye, Phil Overholt, Bill Sanders, Pete Sauer, Ed Schweitzer, Andrea Servida, Neeraj Suri, Bob Thomas, Kevin Tomsovic, and Mani Venkatasubramanian for their useful comments on this research. This research has been supported in part by grants CNS 05-24695 (CT-CS: Trustworthy Cyber Infrastructure for the Power Grid(TCIP)) and CCR-0326006 from the US National Science Foundation; and by grant #60NANB1D0016 (Critical Infrastructure Protection Program) from the National Institute of Standards and Technology, in a subcontract to Schweitzer Engineering Labs Inc. We thank the US Dept. of Energy and Dept. of Homeland Security for their part in funding the NSF TCIP center. We thank Schweitzer Engineering Labs for their donation of equipment and PNNL for their loan of equipment.

REFERENCES

(Note that all the GridStat related references are available via www.gridstat.net.)

[Abe07] Abelsen, S. *Adaptive GridStat Information Flow Mechanisms and Management for Power Grid Contingencies*, MS Thesis, Washington State University, expected August, 2007.

[AF06] Adibi, M. and Fink, L. "Overcoming Restoration Challenges Associated with Major Power System Disturbances—Restoration from Cascading Failures", *IEEE Power & Energy Magazine*, 4(5), September/October 2006, 68–77.

[BBB00] Bakken, D. and Bose, A. and Bhowmik, S. "Survivability and Status Dissemination in Combined Electric Power and Computer Communications Networks", in *Proceedings of the Third Information Survivability Workshop* (ISW-2000), CERT, October, 2000, Boston, MA.

[BBD+02] Bakken, D, Bose, A., Dyreson, C., Bhowmik, S, Dionysiou, I., Gjermundrod, H. and Xu, L. "Impediments to Survivability of the Electric Power Grid and Some Collaborative EE-CS Research Issues to Solve Them", In *Proceedings of the Fourth Information Survivability Workshop*, IEEE, Vancouver, Canada, March 2002.

[BBH06] Bakken, D, Bose, A., and Hauser, C. *EC Efforts in SCADA-Related Research: Selected Projects*. Technical Report EECS-GS-008, Washington State University, 20 October, 2006. Available via http://www.gridstat.net/EC/EC-SCADA-CIP-Report.pdf

[BCH+05] Birman, K. Chen, J., Hopkinson, K., Thomas, R., Thorp, J. van Renesse, R., and Vogels, W. "Overcoming Communication Challenges in Software for Monitoring and Controlling Power Systems", *Proceedings of the IEEE*, 9:5, May 2005.

[BEB01] Bakken, D., Evje, T., and Bose, A. "Survivable Status Dissemination in the Electric Power Grid", in *Proceedings of the Information/System Survivability Workshop*, in *Supplement Proceedings of the International Conference on Dependable Systems and Networks (DSN-2001)*, IEEE/IFIP, Göteberg, Sweden, July 2001.

[BHO+99] Birman, K., Hayden, M., Ozkasap, O., Xiao, Z., Budiu, M. and Minsky, Y. "Bimodal Multicast". *ACM Transactions on Computer Systems*, Vol. 17, No. 2, pp 41-88, May, 1999

[Bir04] Birman, K. "Like it or not, Web Services *are* Distributed Objects!", *Communications of the ACM*, 47:12, 60–62.

[Bir05] Birman, K.. *Reliable Distributed Systems: Technologies, Web Services, and Applications*, Springer Verlag, 2005.

[Bir06] Birman, K. "The Untrustworthy Web Services Revolution," *IEEE Computer*, 39:2, Feb., 2006, 98-100.

[Bos03] Anjan Bose. "Power System Stability: New Opportunities for Control". Chapter in *Stability and Control of Dynamical Systems and Applications*, Derong Liu and Panos J. Antsaklis, editors, Birkhäuser (Boston), 2003.

[BPA06] BPA, Transmission Technology Roadmap, September 2006, http://www.bpa.gov/corporate/business/innovation/docs/2006/RM-06_Transmission.pdf.

[BTB04] Bhowmik, S., Tomsovic, K., and Bose, A. "Communication Models for Third Party Load Frequency Control," *IEEE Transactions on Power Systems*, 19:1, February 2004, pp. 543–548.

[CDK05] Coulouris, G., Dollimore, J., and Kindberg, T. *Distributed Systems: Concepts and Design, 4ed*. Addison-Wesley, 2005.

[Cle07] Cleveland, F. "IEC TC57 Security Standards for the Power Systems Information Infrastructure—Beyond Simple Encryption," IEC TC57 WG15 Security Standards White Paper, http://xanthus-consulting.com/Publications/White%20Paper%20on%20Security%20Standards%20in%20IEC%20TC5%20%20ver%2010.pdf.

[Com03] Comer, D. *Network Systems Design Using Network Processors*, Prentice Hall, 2003.

[Cou07] Cougarr home page, http://www.cougarr.org, 2007.

[Dag06] Dagle, J., "Postmortem Analysis of Power Grid Blackouts—The Role of Measurement Systems", *IEEE Power & Energy Magazine*, 4(5), September/October 2006,30–35.

[DBS+07] Divan, D., Brumsickle, W., Schneider, R. Krantz, B. Gasoigne, R., Bradshaw, D., Ingram, M., and Grant, I. Á Distributed Static Series Compensator System for Realizing Active Power Flow Control on Existing Power Lines", *IEEE Transactions on Power Delivery*, 22:1, Jan. 2007, 642–649.

[DGH+87] Alan Demers, Daniel Greene, Carl Hauser, John Larson, Scot Shenker, Howard Sturgis, Daniel Swinehart, and Douglas Terry. "Epidemic algorithms for replicated database maintenance." In *Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing*, Vancouver, British Columbia, August 10-12, 1987, pp. 1-12. http://doi.acm.org/10.1145/41840.41841

[DGH+06] Demers, Johannes Gehrke, Mingsheng Hong, Mirek Riedewald, and Walker White. "Towards Expressive Publish/Subscribe Systems", In *Proceedings of the 10th International Conference on Extending Database Technology (EDBT 2006),* Munich, Germany, March 2006.

[Dio06] Dionysiou, I. *Dynamic and Composable Trust for Indirect Interactions*, PhD Dissertation, Washington State University, August 2006.

[Eco04] "Building the Energy Internet," *The Economist*, May 11, 2004 (Technology Quarterly issue).

[Ene06] Energitcs Incorporated, *Roadmap to Secure Control Systems in the Energy Sector*, prepared for US Dept. of Energy and US Dept. of Homeland Security, January 2006, available via http://www.controlsystemsroadmap.net/.

[EPRI03] "Smart Power Delivery—A Vision for the Future," *EPRI Newsletter Online*. June 9, 2003. Available: http://www.epri.com/journal/details.asp?doctype=features&id=618

[Gal07] Galvin Institute, "The Path to Perfect Power: The Perfect Power System," 2007, http://www.galvinpower.org/resources/galvin.php?id=92.

[GDB+02] Harald Gjermundrød, Ioanna Dionysiou, David Bakken, and Carl Hauser. "Fault Tolerance Mechanisms in Status Dissemination Middleware", *Supplement of the International Conference on Dependable Systems and Networks (DSN-2002),* IEEE/IFIP, San Francisco, CA, June 2003.

[GDS86] Gurwitz, R. Dean, M. and Schantz, R. "Programming Support in the Cronus Distributed Operating System," in *Proceedings of the 6th International Conference on Distributed Computing Systems*, Cambridge, Massachusetts, USA, May 19-13, 1986. IEEE Computer Society Press, 1986, 486–493.

[Gje06] Gjermundrød, K.-H.. *Flexible QoS-Managed Status Dissemination Framework for the Electric Power Grid*, PhD Dissertation, Washington State University, 2006.

[Gun07] Gunther, E. "Grid Modernization: The Hard Parts," GridWeek 2007, April 23, 2007. http://www.sessionview.com/data/postevent/GW-07/GunterB.pdf

[HBB05] Hauser, C., Bakken, D. and Bose, A. "A Failure to Communicate: Next-Generation Communication Requirements, Technologies, and Architecture for the Electric Power Grid," *IEEE Power and Energy*, 3(2), March/April, 2005, 47–55.

[HBB+07] Hauser, Carl H., Bakken, David E., Dionysiou, Gjermundrød, K. Harald, Ioanna, Irava, Venkata, Helkey, Joel and Bose, Anjan. "Security, trust and QoS in next-generation control and communication for large power systems," *International Journal of Critical Infrastructures*, Interscience, to appear, 2007.

[Hof07] Hoffman, P. "Office of Electricity Delivery and Energy Reliability: Research and Development", presentation at GridWeek 2007 (www.gridweek.com), April 23, 2007, http://www.sessionview.com/data/postevent/GW-07/GridweekRDApril232007(3).pdf

[Hos07] Hossenlopp, L., "Engineering Perspectives on IEC 61850", *IEEE Power & Energy Magazine*, 5(3), 45–50.

[HP06] Horowitz, S. and Phadke, A. "Blackouts and Relaying Considerations—Relaying Philosophies and the Future of Relay Systems", *IEEE Power & Energy Magazine,* 4(5), September/October 2006, 60–67.

[IE07] Ingram, M., and Ehlers, R., "Toward Effective Substation Automation", *IEEE Power & Energy Magazine*, 5(3), 67–73.

[IEC-61850] IEC 61850 Communication Networks and Systems in Substations, IEC, http://www.61850.com.

[IEEE-1646] *IEEE 1646 Standard Communication Delivery Time Performance Requirements for Electric Power Substation Automation*, IEEE, 2004.

[IntelliGrid04] IntelliGrid Project, “The Integrated Energy and Communication Systems Architecture, Vol. IV: Technical Analysis”.2004, Available via http://www.epri.com/IntelliGrid/.

[Ira06] Irava, V. S. *Low-cost delay-constrained multicast routing heuristics and their evaluation*, PhD Dissertation, Washington State University, August 2006.

[JHGB06] Ryan A. Johnston, Carl H. Hauser, K. Harald Gjermundrød, and David E. Bakken. “Distributing Time-synchronous Phasor Measurement Data Using the GridStat Communication Infrastructure,” In *Proceedings of 39th Annual Hawaii International Conference on System Sciences*, Computer Society Press, Kauai, HI, January 4–7, 2006.

[Joh05] Johnston, Ryan A. *Obtaining high performance phasor measurements in a geographically distributed status dissemination network.* MS Thesis, Washington State University, August 2005.

[Kro06] Kropp, T. “System Threats and Vulnerabilities”, *IEEE Power & Energy Magazine*, 4(2), March/April 2006, 46–50.

[Mack06] Mackiewicz, R. “Technical Overview and Benefits of the IEC 61850 Standard for Substation Automation,” *Proceedings of the 2006 Power Systems Conference and Exposition*, IEEE, Oct. 29-Nov 1, 2006, p. 623-630.

[MCG+07] Meliopoulos, A.P., Cokkinides, G., Galvan, F., Fardanesh, B., and Myrda, P. “Delivering Accurate and Timely Data to All”, *IEEE Power & Energy Magazine*, 5(3), 74–86.

[MD07] Myrda, P. and Konahoe, K., “The True Vision of Automation”, *IEEE Power & Energy Magazine*, 5(3), 32–44.

[MKO96] Mittelstadt, W., Krause, P., Overholt, P., Sobajic, D., Hauer, J., Wilson, R., and Rizy, D. “The DOE Wide Area Measurement System (WAMS) Project—Demonstration of Dynamic Information Technology for the Future Power System,” EPRI Conference on the Future of Power Delivery, Washington, D.C., April 1996.

[MRW+07] McDonald, J., Rajagopalan, S., Waizenegger, J., and Pardo, F., “Realizing the Power of Data Marts”, *IEEE Power & Energy Magazine*, 5(3), 57–66.

[OR02] Oman, P. and Roberts, J. “Barriers to a Wide-Area Trusted Network Early Warning System for Electric Power Disturbances,” in *Proceedings of 35th Annual Hawaii International Conference on System Sciences*, 2002.

[PJM07] PJM, *PJM 2007 Strategic Report*, April 2, 2007, http://www2.pjm.com/documents/downloads/strategic-responses/report/20070402-pjm-strategic-report.pdf

[PKT06] Pourbeik, P., Kundur, P, and Taylor, C. “The Anatomy of a Power Grid Blackout”, *IEEE Power & Energy Magazine*, 4(5), September/October 2006, 22–29.

[QuO06] Quality Objects (QuO) homepage, 2006, http://quo.bbn.com.

[Roo06] Root, C. “The Future Beckons”, *IEEE Power & Energy Magazine*, 4(1), January/February 2006, 24–31.

[SHS07] Schomacher, M., Hoga, and C., Schmid, J., “Get On the Digital Bus to Substation Automation”, *IEEE Power & Energy Magazine*, 5(3), 51–56.

[SMD+06] Schainker, R. Miller, P.; Dubbelday, W.; Hirsch, P.; Guorui Zhang, “Real-time Dynamic Security Assessment: Fast Simulation and Modeling Applied to Emergency Outage Security of the Electric Grid” *IEEE Power & Energy Magazine*, 4(2), March/April 2006, 51–58.

[STB86] Schantz. R., Thomas, R., and Bono, G. “The Architecture of the Cronus Distributed Operating System,” in *Proceedings of the 6th International Conference on Distributed Computing Systems*, Cambridge, Massachusetts, USA, May 19-13, 1986. IEEE Computer Society Press, 250–259.

[Swe07] Swenson, K. *High Performance GridStat Status Routing Using Network Processors*, MS Thesis, Washington State University, expected August 2007.

[Taf06] Taft, J. “The Intelligent Power Grid,” *Innovating for Transformation: The Energy and Utilities Project, Vol. 6*, 2006, 74–76. Available via www.UtilitiesProject.com.

[TBV+05] Tomsovic, K., Bakken, D., Venkatasubramanian, M., and Bose, A. “Designing the Next Generation of Real-Time Control, Communication and Computations for Large Power Systems,” *Proceedings of the IEEE (Special Issue on Energy Infrastructure Systems)*, 93(5), May, 2005.

[TCIP07] TCIP: Trustworthy Cyber Infrastructure for the Power Grid, http://www.iti.uiuc.edu/tcip/.

[TEM+05] Taylor, C., Erickson, D., Martin, K., Wilson, R. and Venkatasubramanian, V. “WACS—Wide Area Stability and Voltage Control System: R&D and Online Demonstration,” *Proceedings of the IEEE* (Special Issue on Energy Infrastructure Systems), 93(5), May, 2005.

[TIBCO] http://www.tibco.com.

[TSP98] Thorp, J., Seyler, C., and Phadke, A. Electromechanical Wave Propagation in Large Electric Power Systems”, *IEEE Transactions on Circuits and Systems—I: Fundamental Theory and Applications*, 45:6, June 1998, 614–622.

[TvS07] Tannenbaum, A. and van Steen, M. *Distributed Systems: Principles and Paradigms, 2ed.* Prentice Hall, 2007.

[Uti07a] UtilitiesProject.com, “Enabling Your Vision. Choosing the Right Network,” 2007.

[Uti07b] UtilitiesProject.com, “Getting the Network Right,” 2007.

[Vid07] Viddal, E. *Ratatoskr: Wide-Area Actuator RPC over GridStat with Timeliness, Redundancy, and Safety*, MS Thesis, Washington State University, expected August 2007.

[VR01] Verissímo, P. and Rodrigues, L.. *Distributed Systems for System Architects*, Kluwer Academic Publishers, 2001.

[Wik07] Wikipedia, Service-Oriented Architecture, http://en.wikipedia.org/wiki/Service_oriented_architecture .

[WM06] Wang, L, and Morrison, K. “Implementation of Online Security Assessment”, *IEEE Power & Energy Magazine*, 4(5), September/October 2006, 24–59.

[WMB05] Wu, F., Moslehi, K., and Bose, A. "Power Systems Control Centers: Past, Present, and Future," *Proceedings of the IEEE* (Special Issues on Power Technology and Public Policy: Forty Years After the 1965 Blackout), 93(11), November, 2005, 1890–1908.

[XMV+02] Xie, Z., Manimaran, G, Vittal, V., Phadke, A., and Centeno, V. “An Information Architecture for Future Power Systems and Its Reliability Analysis,” *IEEE Transactions on Power Systems*, 17:3, August 2002, 857–863.

[ZBS97] Zinky, J. and Bakken, D. and Schantz, R., “Architectural Support for Quality of Service for CORBA Objects,” *Theory and Practice of Object Systems*, 3:1, April 1997, 55–73.

[ZLS02] Zinky, J., Loyall, J., and Shapiro, R., “Runtime Performance Modeling and Measurement of Adaptive Distributed Object Applications”, in *Proceeding of International Symposium on Distributed Object and Applications (DOA 2002)*, October 28-30 2002, Irvine, CA, USA.

[ZSS04] Zinky, J, Shapiro, R, and Siracuse, S. ”Complementary Methods for QoS Adaptation in Component-based Multi-Agent Systems”, in *Proceedings of the First IEEE Symposium on Multi-Agent Security (MASS 2004)*, Philadelphia, PA, USA August 30-31, 2004

[ZOB+01] Zinky, J. and O’Brien, L. and Bakken, D. and Krishnaswamy, V. and Ahamad, M. “PASS: A Service for Efficient Large Scale Dissemination of Time Varying Data Using CORBA,” in *Proceedings of the Nineteenth International Conference on Distributed Computing Systems*, IEEE, Austin, TX, May 31 – June 5, 1999, 496–506.

Author Biographies

David E. Bakken is an associate professor of computer science in the School of Electrical Engineering and Computer Science (EECS) at Washington State University (WSU). His research interests include middleware, distributed computing systems, fault tolerance, and quality of service frameworks. Prior to joining WSU, he was a scientist at BBN Technologies where he was an original co-inventor of the Quality Objects (QuO) framework. He has consulted for Amazon.com, Network Associates Labs, and others, and he has also worked as a software developer for Boeing.

Carl H. Hauser is an associate professor of computer science in the School of EECS at WSU. His research interests include distributed computing systems, concurrent programming models and mechanisms, networking, and programming language implementation. Prior to joining WSU, he worked at Xerox Palo Alto Research Center and IBM Research for a total of more than 20 years and was a coauthor of a seminal paper on epidemic multicast algorithms.

Harald Gjermundrød is currently a postdoctorate associate at the High-Performance Computing Systems Laboratory in the Computer Science Department of the University of Cyprus. His work on this research was conducted while he was a doctoral candidate at WSU. His email address in Cyprus is harald@cs.ucy.ac.cy.

Anjan Bose is a Distinguished Professor in Power in the School of EECS at WSU. His research interests include energy control centers, power systems analysis, and power system operations. He is a member of the US National Academy of Engineering and a member of the US National Research Council committee on Improving Cybersecurity Research in the US. He has worked in industry for Consolidated Edison and for Control Data and consulted for companies and governments worldwide.