NASPInet Conceptual Framework

Dr. Yi Hu et al.
Quanta Technology LLC

December 15, 2008

Table of Contents

- Executive Summary
- Major Architecture Design Assumptions
- NASPInet Architecture Framework
- Technical Feasibility of Key Requirements
- Background and Support Materials

Executive Summary

- Completed NASPInet conceptual framework development and high-level architecture design based on:
  - Statement of Work
  - Major assumptions for NASPInet
  - A set of defined major use cases of NASPInet
  - Comments/feedbacks on the draft design by the NETL/DOE review team
- Proposed architecture is highly feasible
  - Most services can be implemented using commercial technologies
  - Certain services requiring customization can use vendor's standard Application Programming Interfaces (API) and Software Development Kits (SDK)
- Out-of-scope item for NETL/DOE consideration
  - Include historian data mart in NASPInet for unified historical data retrieval
    - May need to coordinate with TVA Gen II development
- Specifications to be developed based on this framework approved by NETL/DOE
  - Specifications will define NASPInet requirements at a high level, focusing on "what" is needed, not "how" to build it

QUANTA TECHNOLOGY

Page 3

Revisions made according to the December 1st and 4th DOE-NETL review meeting discussions included the following:

1. Swapped Class C and Class D data definition. This change allows the NASPInet to group Class A, B and C data as real-time streaming data, and Class D and E data as non-real-time historical data.

2. Noted that the NASPInet architectural framework would potentially support historical data from internal data stores in the future and as a minimum provide the services for transporting historical data from external sources through the NASPInet Historian.

3. Clarified major architecture assumptions.

4. Provided more illustrations on security and QoS management.

Project Schedule and Next Steps

- Next steps
  - January 12, 2009
    - Prepare for NASPI meeting in Phoenix AZ
  - **January 26, 2009**
    - Provide presentation materials for project status report
    - Provide materials for draft specifications review
  - *February 3, 2009*
    - Review draft specifications
  - *February 4-5, 2009*
    - Present project status report at NASPI meeting in Phoenix
- **April 27, 2009** – Final project deliverable

Legend: Dates in **red** are contractual due dates
Dates in *blue* are scheduled near-term meeting dates

QUANTA TECHNOLOGY

Page 4

A framework for the NASPInet specification development is included in the Background and Support Materials section of this presentation.

Table of Contents

- Executive Summary
- Major Architecture Design Assumptions
- NASPInet Architecture Framework
- Technical Feasibility of Key Requirements
- Background and Support Materials

Major NASPInet Architecture Assumptions (1)

- Regarding granularity level in Requirement P10, NASPInet will support the following:
  - Final implementation to enable access control granularity to the signal (variable) level – a subscriber will only get signals that it subscribes
    - For aggregated data transporting through the NASPInet, disaggregation could occur at any point prior to receipt by the subscriber
  - Initial implementation to allow access control granularity at IEEE C37.118 frame level or any other standard-based data packet level
- The main function of NASPInet is to facilitate the inter-entity data exchange through data publishing and subscription
  - Implementation of certain functions, such as time alignment of data, data processing, data storage, data validation, etc., will only be supported in NASPInet on an as-needed basis
    - The architecture framework can support these functions but including these functions may incur performance penalties
- One entity can have more than one physical Phasor Gateway (PG)
  - For example, one utility that owns facilities in several geographical areas may need to have several physical PGs, one PG for each area
  - Scalability may be realized by using multiple physical PGs

Major NASPInet Architecture Assumptions (2)

- Each connecting entity will be responsible for enforcing access control and system security on its side of the PG
  - User account setup and access level control
  - Application's class of data access control
  - Cyber security measures within entity's own network connecting to PG
- NASPInet will only manage QoS from publishing PG to subscribing PG. Each connecting entity will be responsible for QoS assurance within its own networks
  - From data source to publishing PG, and/or
  - From subscribing PG to users/applications

Major NASPInet Architecture Assumptions (3)

- Within NASPInet, subscribing PGs must get access authorization for any class of data (class A – E) from the data's originating publishing PG, and each PG can only publish data that originates from it
  - Prevent resending received data to other PGs within NASPInet without proper authorization from publishing PG
    - Registered signal ID, PG ID, data signature, etc. may be used to prevent redistribution of streaming data
    - Historical Class D and E data retrieval will require authorization from data originator (publishing PG) irrespective of where the data is stored
  - Data archive/distribution outside NASPInet will be governed and enforced by external policies, agreements and procedures
    - For example, access of data streamed/stored by non-publishing entities that does not go through NASPInet

Table of Contents

- Executive Summary
- Major Architecture Design Assumptions
- NASPInet Architecture Framework
- Technical Feasibility of Key Requirements
- Background and Support Materials

NASPInet Data Class Grouping

NASPInet Traffic Attribute	Real-time streaming data			Historical data	
	CLASS A Feedback Control	CLASS B Feed-forward Control	CLASS C Visualization	CLASS D Post Event	CLASS E Research
Low Latency	4	3	2	1	1
Availability	4	2	1	3	1
Accuracy	4	2	1	4	1
Time Alignment	4	4	2	1	1
High message rate	4	2	2	4	1
Path Redundancy	4	4	2	1	1

Table key:
4 – Critically important, 3 – Important, 2 – Somewhat important, 1 – Not very important

Page 10

The above table is based on SOW with the following change according to NETL/DOE review team suggestion:

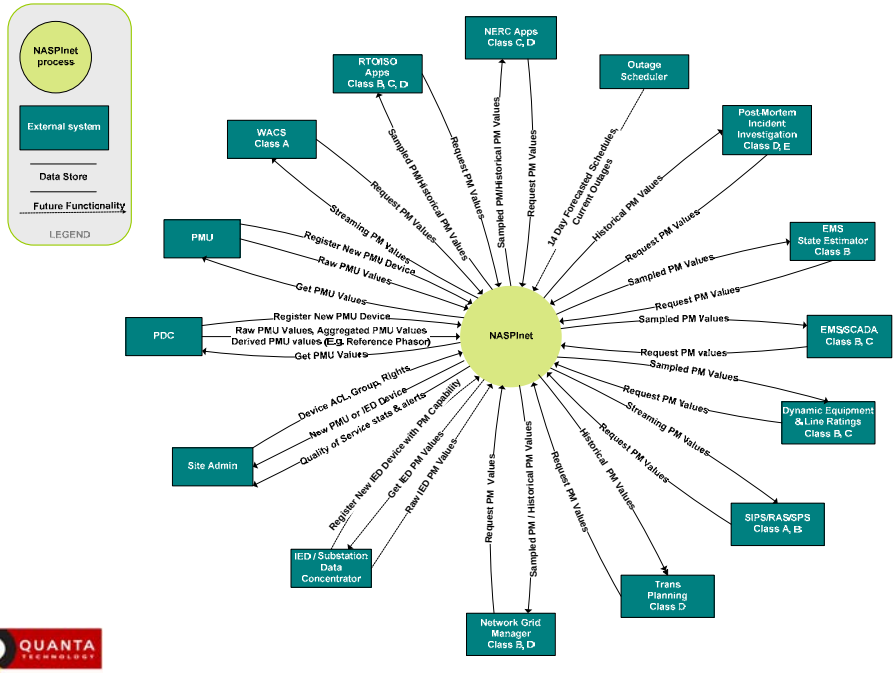
Swapped Class C and Class D definition to enable easy data type grouping – real-time streaming data (Class A, B, and C) and historical data (Class D and E). The two types of data requires different mechanism to facilitate data publishing and subscription. Real-time streaming data is typically in a one-to-many publishing/subscribing situation, while historical data access is always one-to-one.

The data type grouping is used in the use case development and system architecture design.

List of Major NASPInet Use Cases

Category	Case	Description
System Admin	SYS1	Connect/disconnect a Phasor Gateway (PG) to/from the Data Bus (DB)
	SYS2	Update DB – add/delete/update DB's hardware and software
	SYS3	Update PG – add/delete/update PG's hardware and software
System Usage	NET1	Add/delete/modify a real-time data stream data device (such as a PMU, a PDC, or a data publishing application)
	NET2	Add/delete/modify a historical data source (such as a historian, or an archive system)
	NET3	Setup/start/stop/delete/edit a real-time data connection (class A, B, and C data)
	NET4	Setup/start/pause/resume/stop a historical data delivery (class D and E data)
	NET5	Monitor/Alarm/report Quality of Service (QoS) of data transfer

System Context Diagram



This view of NASPInet is intended to validate the scope of the system by identifying data flows between NASPInet and external systems and end-user applications at a high, abstract level. The system context diagram is exploded into data flow diagrams to illustrate the major flows that will be support by NASPInet and to identify major system components that form the building blocks of NASPInet. The data flow diagrams are included in the Background and Supporting Materials section of this slide deck.

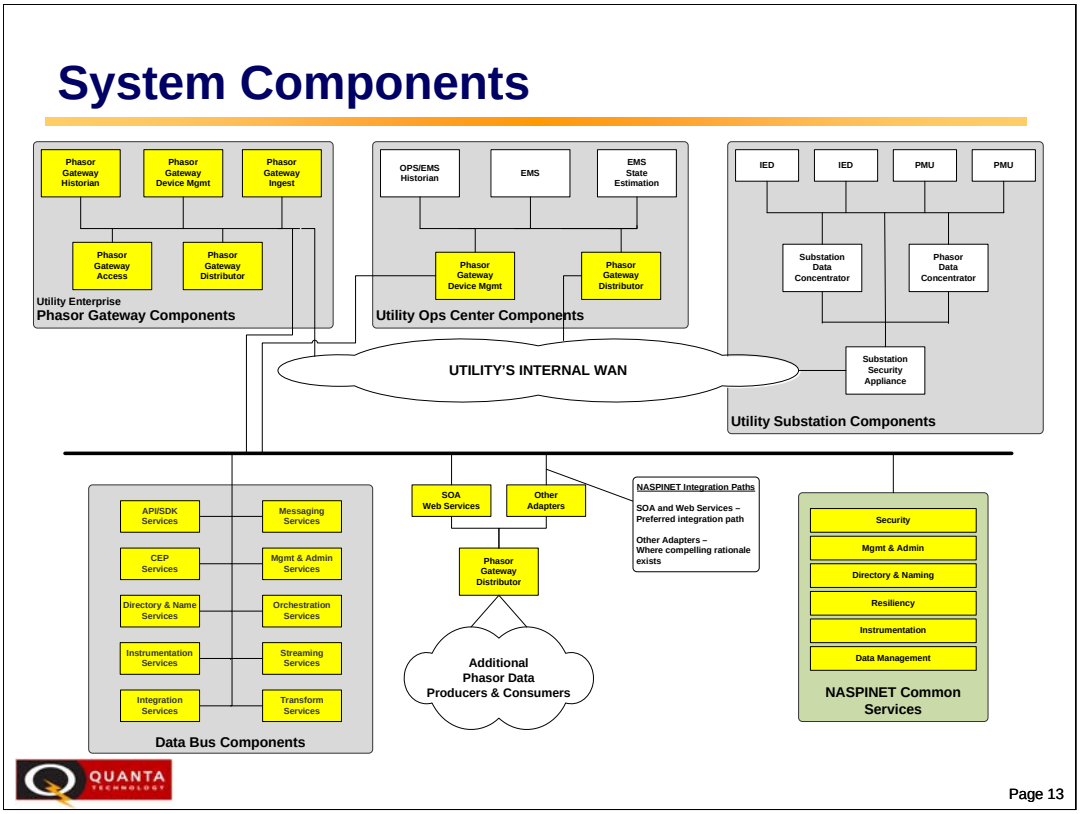
<u>Nomenclature</u>

•Circles represent components of NASPInet,

•Boxes represent external systems and user applications.

•Data stores indicate persistent storage required.

•Data flows arrows indicate direction of flow, there is no intent to try to represent sequencing of required flows in this view. These will be represented in UML sequence diagrams later.



This diagram is intended to identify the major system architecture components that form the NASPInet Phasor Gateway and Data Bus. It also maps the components to where they may likely be deployed. Descriptions of these components are given in the Background and Supporting Materials section at the end of this presentation, including a discussion on Security Services. Major services contained in each component are shown in the next slide.

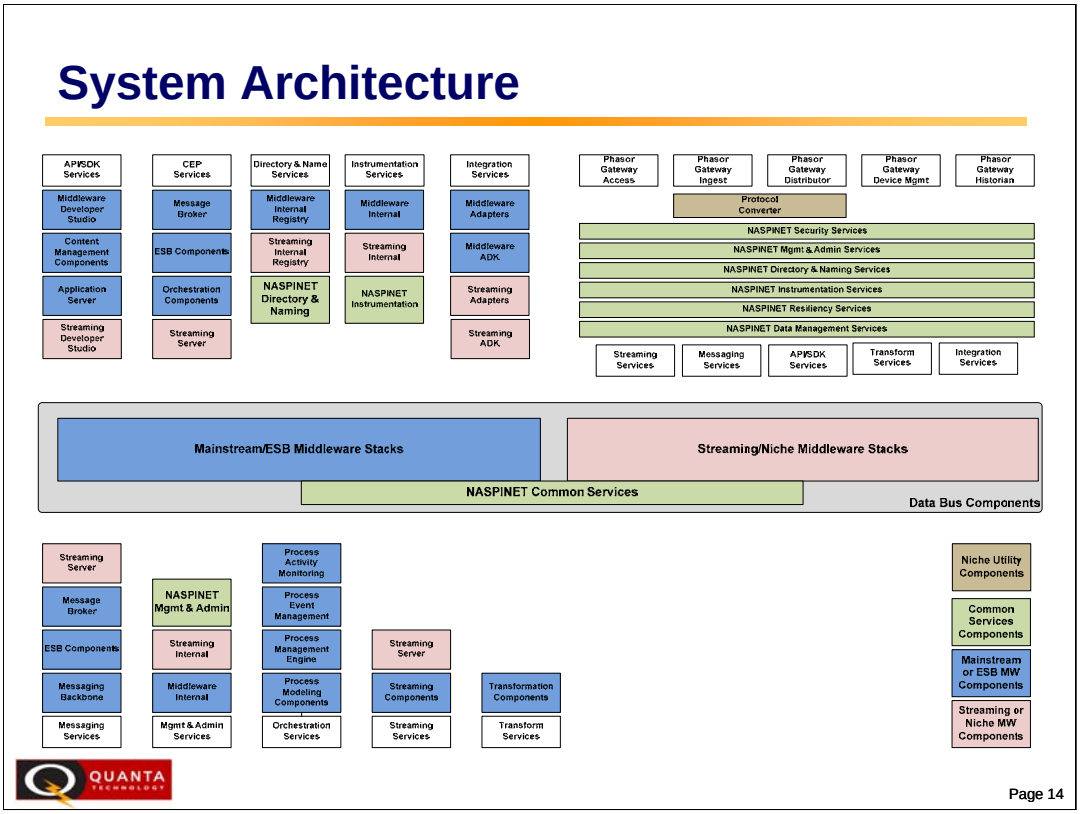
While other integration approaches are potentially possible, the Service Oriented Architecture (SOA) was referenced here as a plausible approach due to its facilitation of development and ongoing maintenance and support, and due to its advances in performance in recent years. This architecture supports both approaches of sending all Phasor Measurement signals to NASPINET consumers for their PG's to extract the desired information based on subscription, security, etc. and of sending only the signals requested. The SOA approach favors the latter; nevertheless, the former may be implemented to optimize use of the NASPInet bandwidth. Potential bidders for implementing NASPInet will have the option of providing solution for either or both of communication approaches.

For more information on SOA, please visit the following websites for example:

•http://www.ibm.com/soa

•http://www.tibco.com/customers/

While it has not been used in streaming PMU data, the SOA has been used in various large-scale applications in the Energy, Telecomm and Financial sectors, including Amazon and EBay.



This diagram breaks down each of the components identified above into a set of services contained in it. The services may come from three general service stacks, as indicated by different color boxes in the diagram:

•The Enterprise Service Bus (ESB) or Middleware stack commonly used in information systems and system integration

•The specialized Streaming Processing / Middleware stack for high-performance high-volume streaming real-time data in specific standards and protocols (e.g. C 37.118, 61850, etc.)

•NASPInet Common Services listed in System Components slide

The PG set of components will heavily reuse the common services and some of the data bus components. We propose to leverage commercially available tools that address the utility specific protocols associated with PMUs and PDCs where appropriate, and supplement these tools with custom solutions.

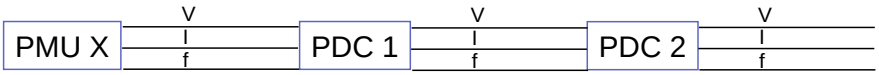
NASPInet Name Service Requirements

- Must-have requirements
  - Be able to uniquely identify any physical device and each of its logical signals across the NASPInet
  - Enable users/applications to find and select any device/signal from NASPInet that they are allowed to access through the name service
- Good-to-have requirements
  - Support existing naming conventions through mapping and conversion

Proposed NASPInet Name Service Method

- Through a registration process for registering all devices and the signals that each device provides in NASPInet name service database
  - Manual: Owner will register a device and its signals by filling a standard meta data form for all required information on owner's PG
  - Automatic: device sends in all information for registration
- A unique signal ID number will be assigned by the name service for each registered signal
- Required information will include location, ownership, device type, device number, device configuration, signal source, etc., as well as:
  - Information for tracing intermediate devices' output signals (e.g. PDC) back to the original PMU
    - Enable users/applications to select from different signals having the same original PMU, and subscribe to a signal either from its original PMU or the output of a specific PDC
- Support mapping NASPInet signal ID number to an entity's own naming convention or notation used within its systems and applications
- Support signal/device discovery through standard database queries to name service with:
  - Requesting PG's credential check
  - Publishing PG's access control

Name Service Illustration Example



Original PMU signal ID	Device Name	Signal
X	PMU X	V
X	PMU X	I
X	PMU X	f
X	PDC 1	V
X	PDC 1	I
X	PDC 1	f
X	PDC 2	V
X	PDC 2	I
X	PDC 2	f

(1) NASPInet will automatically assign a unique identifier for each data signal from each device.

(2) User applications will be able to see what data is available and select the signal from a specific device.

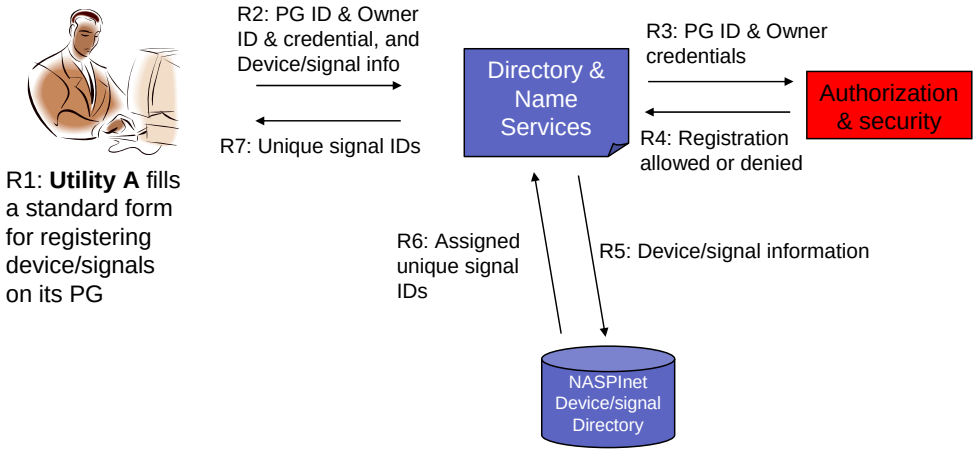
(3) NASPInet will be able to trace every signal back to its origin and the PDCs that it has gone through.

- *Signals with the same origin may not have the same attributes after going through intermediate devices. For example, reporting rate may be changed.*
- *Data access must be authorized by the PG of origin (an attribute of each PMU device).*

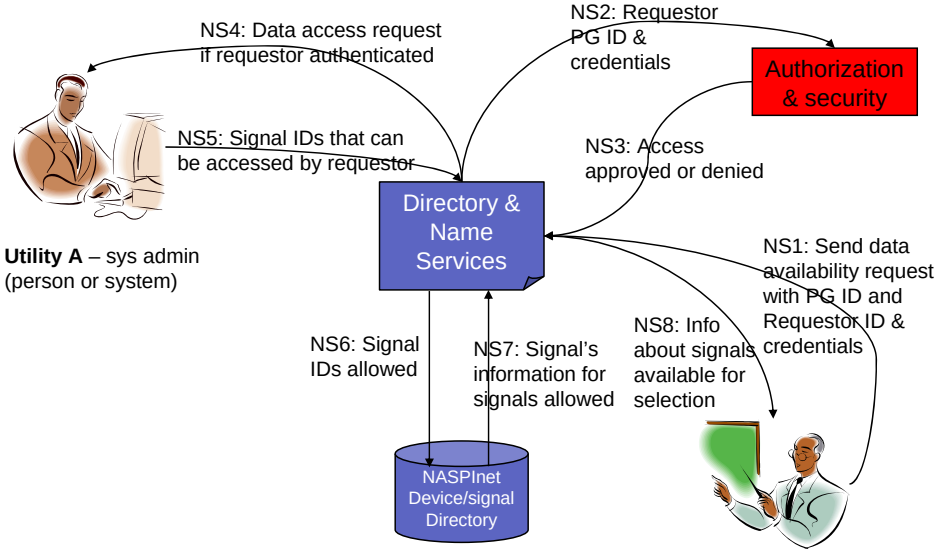
Main Advantages of the Proposed Method

- Compared with name service based on a structured naming convention, the proposed method:
  - Improves security in exchanging signal ID numbers through the NASPInet
    - The signal ID number does not reveal any specific information about the signal
  - Guarantees uniqueness with minimal length of an ID number
    - Name (or ID number) will be too long if using structured naming convention to guarantee uniqueness
  - Allows more complete, non-abbreviated, information to be used by name services
    - Using short names according to certain structured naming conventions may not be able to include all the information and guarantee uniqueness
  - Improves scalability
    - Avoid limitation of the naming structure (e.g. if device name is limited to 2 bytes, it may limit how many new future devices can be named)
  - Improves ease and flexibility of data access
    - Enables multiple ways (browse, search, etc.) to find a desired signal
    - Allows signal level granularity of name service
  - Enables future automated device and signal registration

Device/Signal Registration Process Illustration



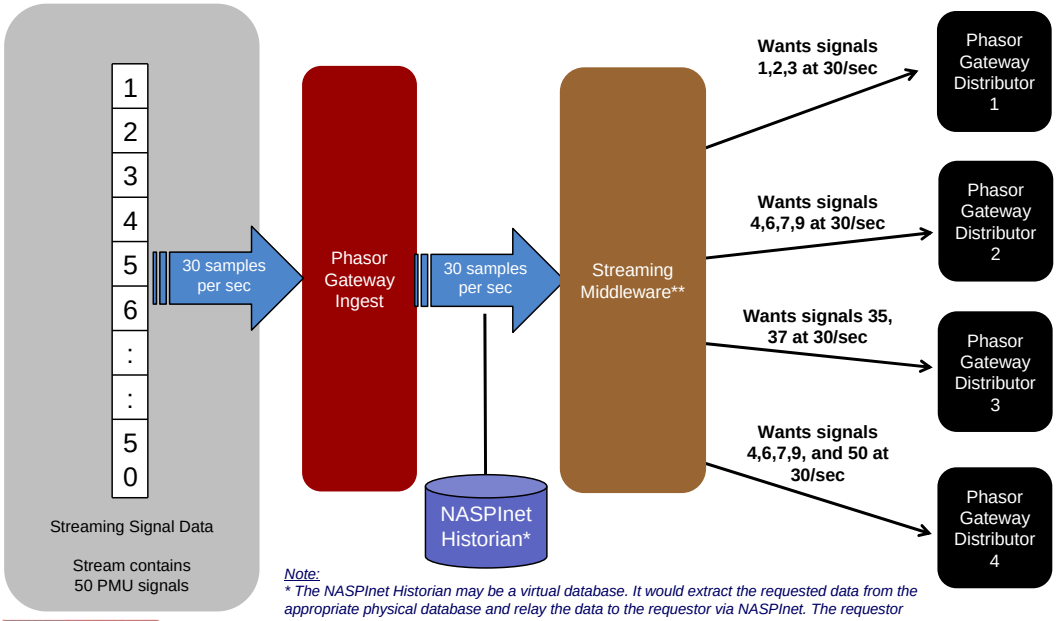
Device/Signal Access Process Illustration



NS1: **Utility B** looks for some voltage signals in utility A's area through its PG

Page 20

Stream Processing Conceptual Example



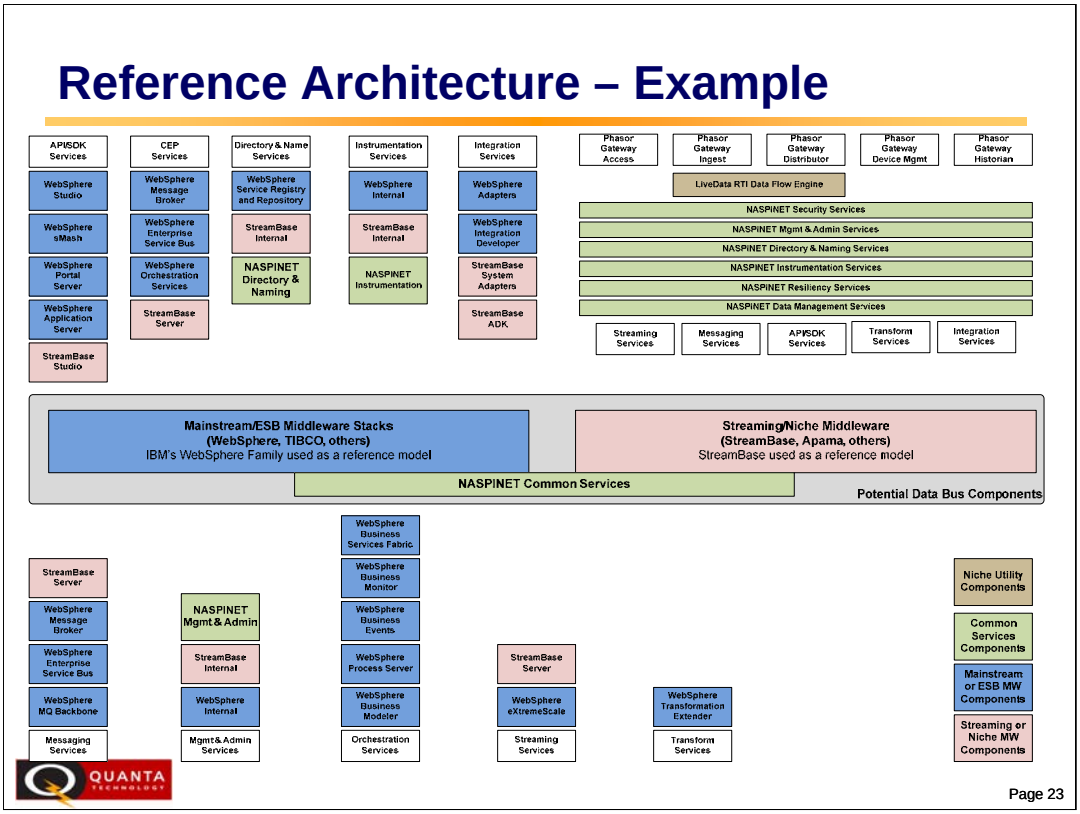
Note:
** The NASPInet Historian may be a virtual database. It would extract the requested data from the appropriate physical database and relay the data to the requestor via NASPInet. The requestor application is assumed to transform the data to the format that meets its needs.*
*** The Streaming Middleware can potentially allow throttling or "titration" of PMU data signals.*

QUANTA TECHNOLOGY

Page 21

Table of Contents

- Executive Summary
- Major Architecture Design Assumptions
- NASPInet Architecture Framework
- Technical Feasibility of Key Requirements
- Background and Support Materials



This diagram attempts to map the architecture components' services onto commercially available products to (1) determine whether the proposed architecture is commercially viable, (2) see how much custom development is required, and (3) to develop order of magnitude cost estimates for the architecture.

Besides bringing the NASPInet functionality required, the system architectures of the reference products (e.g. StreamBase) have proven to scale well by well-written multi-threaded components that support multiple occurrences of their services, by using enterprise class hardware servers, and by using enterprise integration infrastructure for communications among internal services and components. These industrial strength products also support high available configurations and Business Continuity or Disaster Recovery configurations very well.

While the niche stream processing engine has not been used or proven in the proposed problem domain, it has been deployed in analogous problem domains and has some very impressive performance citations: to the tune of 100K+ transactions/sec per CPU core.

The abilities of this reference architecture design to meet core NASPInet requirements are elaborated in the next few slides. For more information on a sample of potential commercial products for general - mainstream enterprise middleware and streaming middleware, please visit the following vendor websites:

•http://www.ibm.com/software/websphere

•http://www.tibco.com/

•http://www.streambase.com

•http://www.progress.com/apama/index.ssp

•http://www.coral8.com/

Technical Feasibility of Key Requirements

- The following slides use StreamBase and IBM Websphere (or other enterprise middleware such as TIBCO and Oracle) in the reference design as an example to illustrate the technical feasibility of key NASPInet requirements:
  - Quality of Service management
    - Real-time streaming data performance
    - Bandwidth usage optimization and management
    - Support for different priorities in simultaneous streaming of different class data
  - System availability and scalability
  - Security
  - System flexibility and configurability
  - Support for incremental deployment

Quality of Service Management (1)

- Real-Time Streaming Data Performance
  - Managed through Data Bus services
    - Maps QoS to Class of Service
    - Manages NASPInet resources; reserving resources to high-priority data
    - Negotiates and adapts QoS to resource constraints
  - Delivered via Stream Processing Middleware in the Data Bus – ingests, processes and routes results from continuous source data streams
    - Stream Processing Middleware operates above the network layer so the underlying network architecture could layer additional QoS and other mechanisms
  - Stream processing engines like StreamBase are built for high volume, low latency, and high availability and field proven in managing financial transactions
    - Benchmarks of up to 500K messages trade messages processed per second, per CPU
    - Assuming conservative performance weighting of 50%, expectations would still be ~200K messages/second on a typical rack/blade server

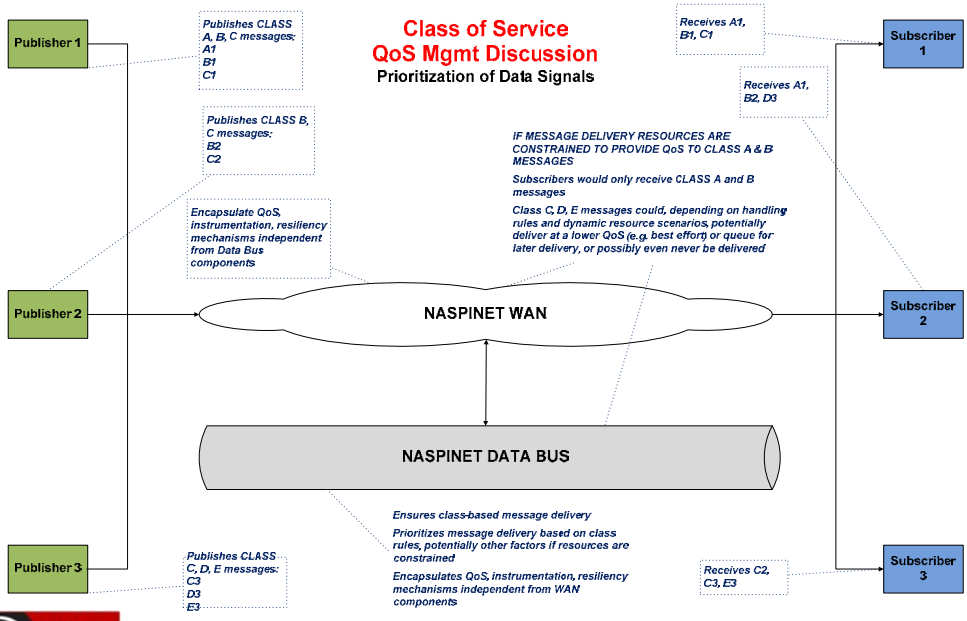
QUANTA TECHNOLOGY

Quality of Service Management (2)

- Bandwidth Optimization & Management
  - Stream Processing Engines operate on real-time data streams and optimally route results to consumers; for example, a 50 signal data stream could be distributed to 100's of consumers at once, optimizing bandwidth usage and latency
  - Stream Processing Engines like StreamBase provides ability to configure stream processing characteristics to provide additional QoS management
  - This architectural framework could also support multiple deployments, with affinity for geographic, bandwidth or other characteristics, to further partition the Data Bus for bandwidth optimization and management
- Support for different priorities in simultaneous streaming of different class data
  - A typical Enterprise Service Bus (e.g. IBM Websphere, TIBCO, Oracle BEA-WebLogic…) has built-in functions to prioritize messages by type or by content sensitivity for routing.

QUANTA TECHNOLOGY

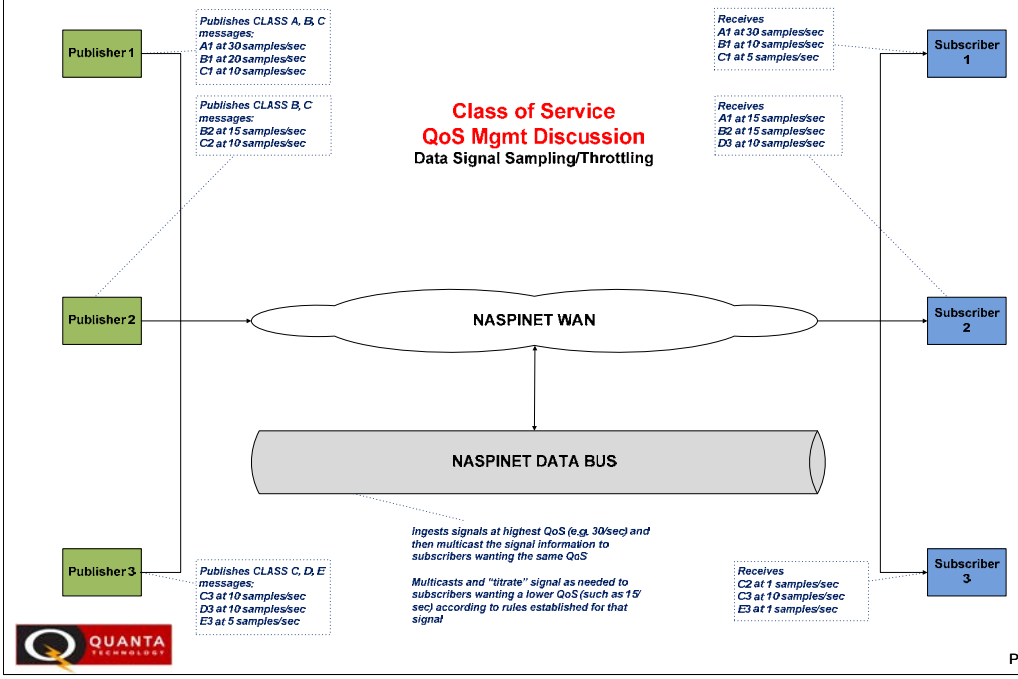
QoS Management Illustration (1)



QUANTA TECHNOLOGY

Page 27

QoS Management Illustration (2)



Page 28

System Availability and Scalability

- Architecture components follow core principle of logical and physical partitioning to enable availability and scalability
- Platforms – Servers, storage, etc.
  - Scalability by adding or enabling CPUs, implementing partitioning and/or virtualization, adding platforms, grid configurations
  - Availability via features inherent in hardware, operating systems, system software, storage networks, enterprise management suites
- Data Bus components
  - Enterprise Middleware stacks are mature, proven
  - Streaming Middleware scalability and availability proven in Finance, Trading, Telecomm industries for 5+ years
- Network
  - Scalability via incremental bandwidth coupled with other network aspects such as QoS and Data Bus aspects such as Stream Management
  - Availability via redundant network design

QUANTA TECHNOLOGY

System Security

- Commercial Public Key Infrastructure (PKI), Identity Management (IM), Trust Management (TM), Key Management (KM) components will act as broad enablers in the following areas:
  - Authentication – Multiple factors and contexts, PKI
  - Authorization & Access Control – Role-based, grant by exception
  - Confidentiality – Channel/payload encryption, KM
  - Audit & Compliance – Multi-level logging (transactions, events, user activities, devices, etc.) and incremental penetration testing
  - Non-Repudiation – Multi-level logging, PKI, IM
- Infrastructure – Intrusion Detection Systems (IDS), Network Access Control (NAC), Information Leakage Protection (ILP), as well as traditional perimeter components (such as firewalls) will act as broad enablers to prevent unauthorized access, changes and usage
- Physical – Industry recognized data center attributes such as Uptime Institute's Level IV Tier classification

QUANTA TECHNOLOGY

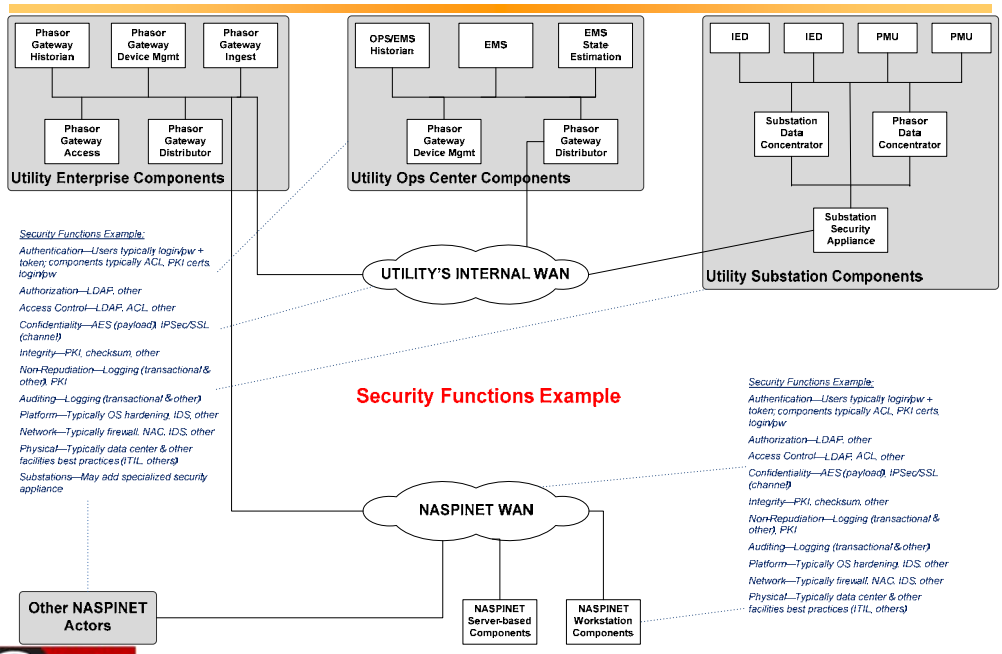
Page 30

Multi-factor Authentication of something you know/have/are; contexts=time interval, location etc. such as RSA fob

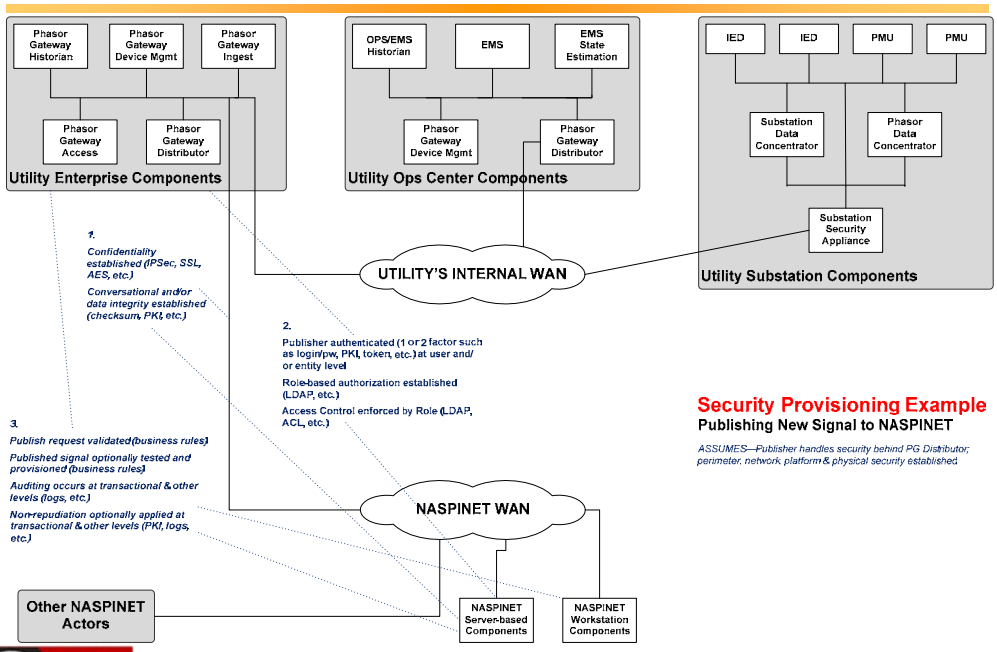
Confidentiality protection via a strong channel, for example, VPN or dedicated network, etc. and via payload information encryption using AES256 for example.

Please see the Background and Support Materials section for more discussions on Channel/Payload encryption.

Security Functions – Example



Security Services Provisioning Example



Security Provisioning Example

Publishing New Signal to NASPINET

ASSUMES—Publisher handles security behind PG Distributor; perimeter, network, platform & physical security established

QUANTA TECHNOLOGY

Page 32

System Flexibility/Configurability

- Configurability is enabled via:
  - Bias toward mature commercial tools with extensive inherent configuration options
  - These tools provide rich, standards-compliant foundations that promote building components, services and applications in a configurable manner
    - For example, developing stream processing components using configurable visual models, or configuring the behavior of integration adapters
- Flexibility & extensibility is enabled via:
  - Commercial middleware stacks providing standards-based extensibility via Java, Web Services, XML awareness and other characteristics
  - Service-Oriented Architecture (SOA) approach to customization and development enables reuse & combination of business and process logic
- Support of device-level or signal level data transport and access
  - Payload and framing are done outside of the Data Bus. So it has the flexibility of supporting many derivations and tools to parse and sample content as needed
  - The proposed architecture can re-use utility niche protocol stacks from vendors (e.g., SISCO, Livedata, or possibly a PDC vendor) to support converting to/from C37.118 for communication and retrieval of data

QUANTA TECHNOLOGY

Support for Incremental Deployment

- Incremental Scalability
  - All architectural components scale as needed both logically and physically
    - Providing flexibility in resource acquisition, deployment and operation
  - Enables NASPInet to appropriately manage architectural resources relative to operational costs, energy consumption and other relevant IT asset portfolio metrics
- Proactive Provisioning
  - Decouples provisioning of system services from their instantiation and operation
  - Provides flexibility of development, modeling, simulation and testing before PMUs are brought online
  - When the PMUs come online, the Data Bus can instantiate, operate and monitor these new streams with relatively low effort
- Service Oriented Architecture
  - Fosters reuse, aggregation and extension of system services

QUANTA TECHNOLOGY

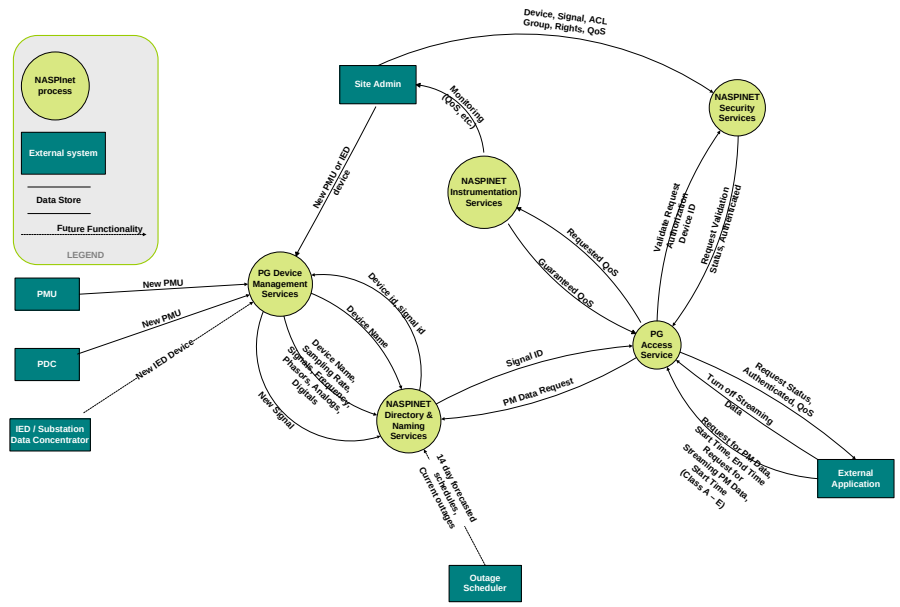
Questions?

yhu@quanta-technology.com

Table of Contents

- Executive Summary
- Major Architecture Design Assumptions
- NASPInet Architecture Framework
- Technical Feasibility of Key Requirements
- Background and Support Materials

NASPInet System Admin Data Flow



Page 37

This data flow diagram shows the system administrative flows of NASPInet site administrators and external users and applications.

<u>Considerations and assumptions:</u>

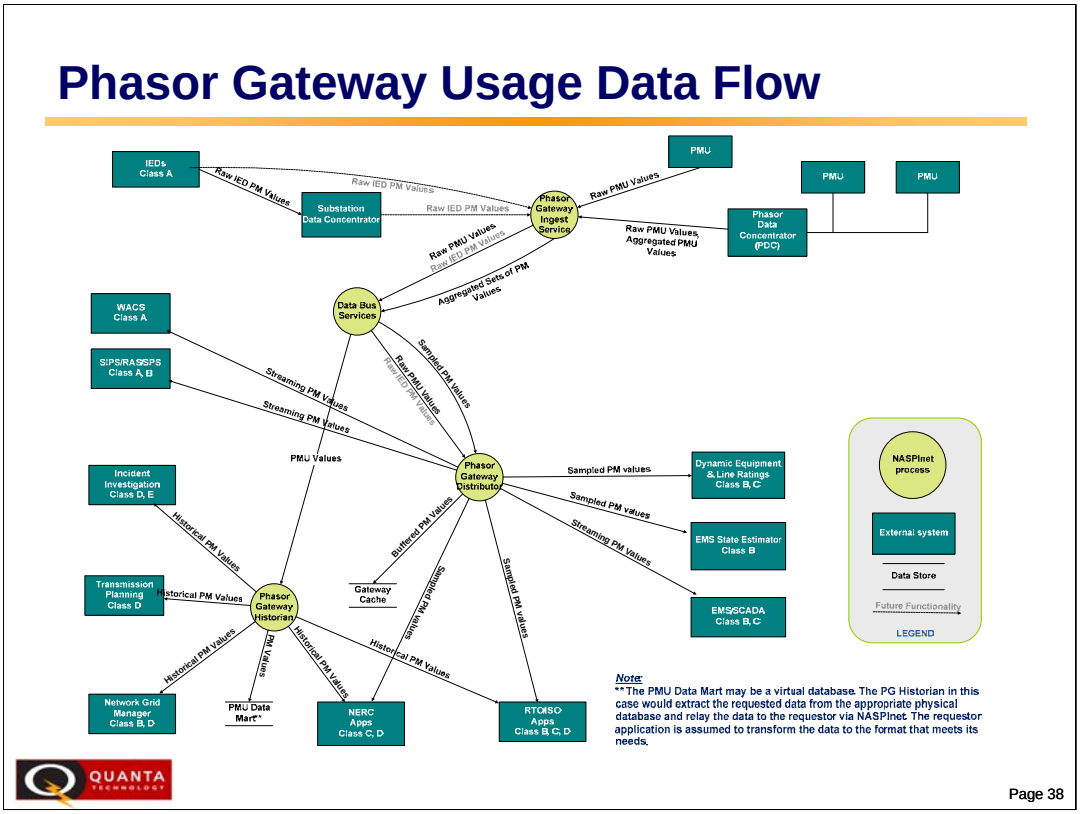
•New PMU device signals of existing PMUs can be added to NASPInet.

•New IED devices that produce PM data can potentially be added to NASPInet either directly or via substation data concentrators in the future.

•Applications will be allowed to subscribe for a specific level of service (QoS) and subset of signals for a PMU device.

- •Based on subscriber admission
- •Subject to QoS-Class of Service mapping
- •Maybe resource-constrained and requiring re-negotiation and QoS adaptation

•NASPInet can allow external scheduling systems to indicate near term unavailability of devices and/or device signals for proactive QoS notifications.



This diagram shows data moving from field devices to “publisher” phasor gateways and through the data bus to the “distributor” phasor gateways to external systems and end user applications. The data flows show the major types of data that need to be supported by the architecture:

•Streaming PM values

•Sets of PM values

•Historical PM values

<u>Considerations and assumptions:</u>

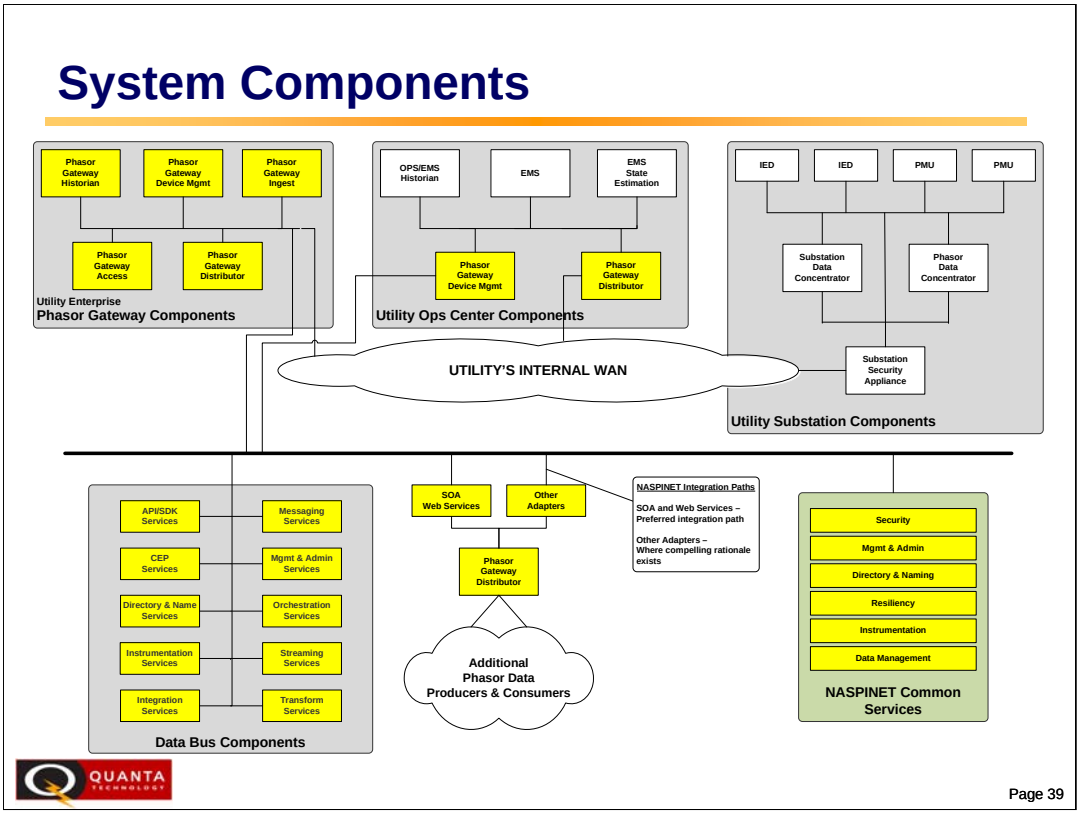
•The logical location of adding a data archival service is currently depicted as part of the NASPInet data ingests data flow. Assumption is that the NASPInet project will have control over the schema definitions to be used to archive NASPInet information. If data archival is out of the NASPInet scope, the historian/data mart would represent external systems that need to be integrated with NASPInet. Though the architectural framework can support internal and external historical databases, not including the data archive in NASPInet will add complexity to the implementation for managing multiple archival schemas when handling historical requests.

•PG Ingest Service would handle data from PDC, PMU, and potentially substation data concentrators that collect PMU related data from IEDs and other metering equipment within the substation in the future.

•PG Collector Service supplies (publishes) data to applications and systems that have registered for the data. To support varying performance characteristics of the subscribing systems, the gateway will need to cache PM values temporarily.

•QoS management will be:

- •Based on subscriber admission
- •Subject to QoS-Class of Service mapping
- •Maybe resource-constrained and requiring re-negotiation and QoS adaptation



This diagram is intended to identify the major system architecture components that form the NASPInet Phasor Gateway and Data Bus high level components. It also maps the components to where they may likely be deployed. Descriptions of these components and services are given in the following slides.

Phasor Gateway Components

- PG Ingest communicates with PMU, PDC and potentially other IED devices in the substation and put the data from them on the Data Bus for consumption by external systems and user applications.
- PG Distributor provides for all data requests from external applications and systems.
- PG Device Management supports adding, removing, activating and de-activating devices
- PG Access provides the security administration services to grant user roles/application access to specific device and signal data.
- PG Historian archives select Phasor Measurements in common structure and format to facilitate 'major' event investigations for example.

Data Bus Components (1)

- API/SDK (Application Programming Interface / Software Development Kit) enables applications, integration mechanisms, user interfaces, content delivery and other NASPInet components to be customized, developed and deployed.
- CEP (Complex Event Processing) enables event-driven processing, such as actions to be performed based on the values of multiple phasor measurements or QoS monitors.
- Directory & Naming Services enable the registry of services, components, processes, streams, signals and other entities internal to the Data Bus for subsequent invocation.
- Instrumentation Services provide visibility into key aspects of Data Bus components and services, such as performance, utilization and general health indicators.
- Integration Services enable external integration to NASPInet by exposing Data Bus services, processes and components via adapters.

Data Bus Components (2)

- Messaging Services enable synchronous and asynchronous message-based communication between NASPInet components and services with support for features such as guaranteed delivery, publish/subscribe and content-based routing
- Management & Administration Services enable the initial configuration and ongoing operation and maintenance of Data Bus components and services
- Orchestration Services enable business process modeling, development, instantiation, execution and monitoring, supporting the provisioning of new Phasor Gateway devices for example.
- Streaming Services enables massive volume, low latency delivery and processing of NASPInet data from PMUs/PDCs and also other IEDs in the future.
- Transform Services enable data transformation on the data flowing through the Data Bus as needed.

NASPInet Common Services (1)

These services support or enable functionality throughout NASPInet.

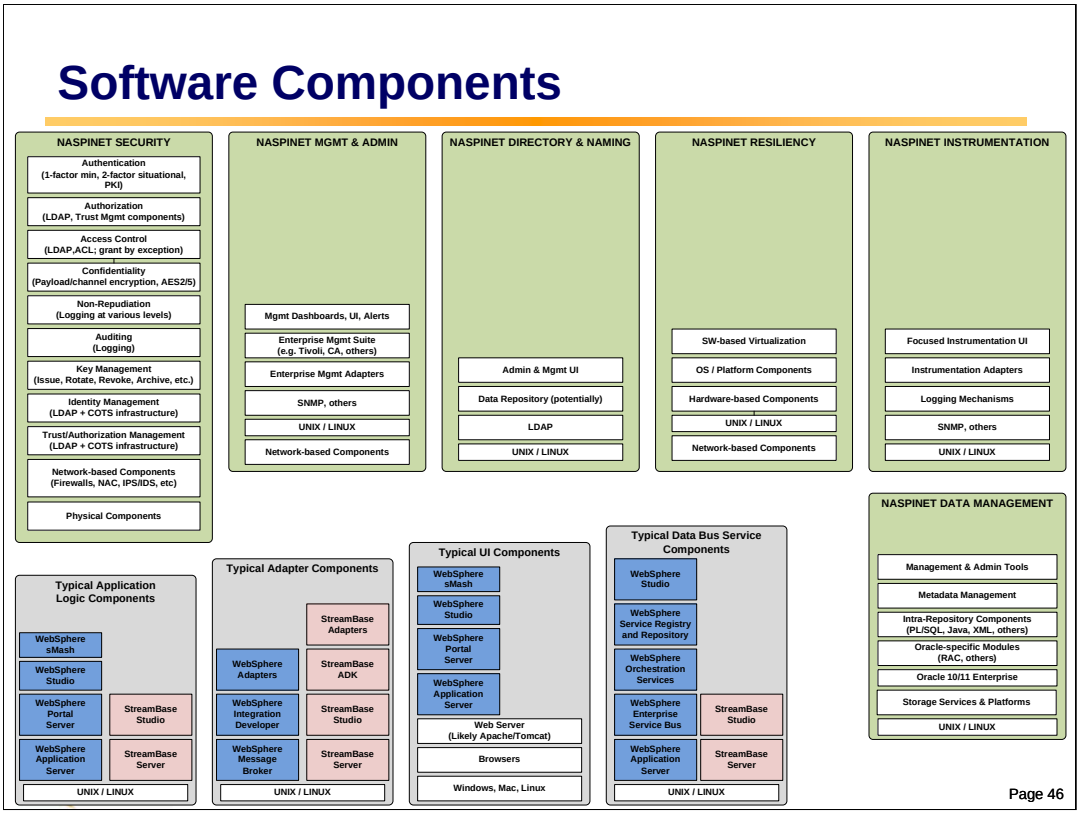
- Security ensures secured access to and usage of NASPInet resources and information. Key components typically include Authentication, Authorization, Access Control, Confidentiality, Auditing, Non-Repudiation and many others.
- Management & Administration enables the initial configuration and ongoing operation of NASPInet components and services, and will likely integrate or aggregate the analogous but more focused services within the Data Bus
- Directory & Naming provides for system-wide registry of PMU/PDC devices and signals as well as services, components, processes and other entities required by the PG and DB components.
- Resiliency ensures critical system attributes such as Fault Tolerance, Availability, Disaster Recovery, Business Continuity, etc.

NASPInet Common Services (2)

- Instrumentation, in concert with the Management and Administration Services, provides visibility into key aspects of the NASPInet infrastructure such as performance, utilization and general health indicators.
- Data Management provides for process, storage, access and management of persistent and transient data within NASPInet. Key components typically include Relational Database Management engines, Storage Area Network infrastructure, Metadata Management, Hierarchical Storage Management, Information Lifecycle Management, etc.

Channel/Payload Encryption Example

- Potential security process steps:
  1. Add a good size "random pad" to each signal data (essentially increase the size) and encrypt the "padded signal" using the "signal key".
  2. The frame is assembled – which is concatenation encrypted signals.
  3. The frame is encrypted, OR
  4. The frame is coded using message authentication codes (e.g., HMAC).
- Note:
  - Keys need to be exchanged among devices accessing the network.
  - Symmetric key encryption incurs less overhead than asymmetric key encryption.
- Option A – Do steps 1 and 2
  - This provides data confidentiality.
  - The receiver needs one key to decipher a signal.
- Option B – Do steps 1, 2 and 3
  - The receiver needs two keys to decipher a signal – the frame key and the specific signal key.
  - This provides data confidentiality and two layers of encryption for greater security.
- Option C: Do steps 1, 2, and 4
  - Process
    - Sender sends frame (X) and the HMAC encoded frame (Y).
    - Receiver decodes Y and compares with X to verify message integrity and authenticate the source.
    - Receiver uses the signal key to decrypt the signal, using multiple keys If it needs more than one signal.
  - This provides data confidentiality, data integrity, data origin identification, and replay protection.
- Option D: Do all four steps
  - Two layers of encryption (signal level, frame level) followed by message authentication coding
  - This requires bigger message and process overhead but provides stronger confidentiality and security.

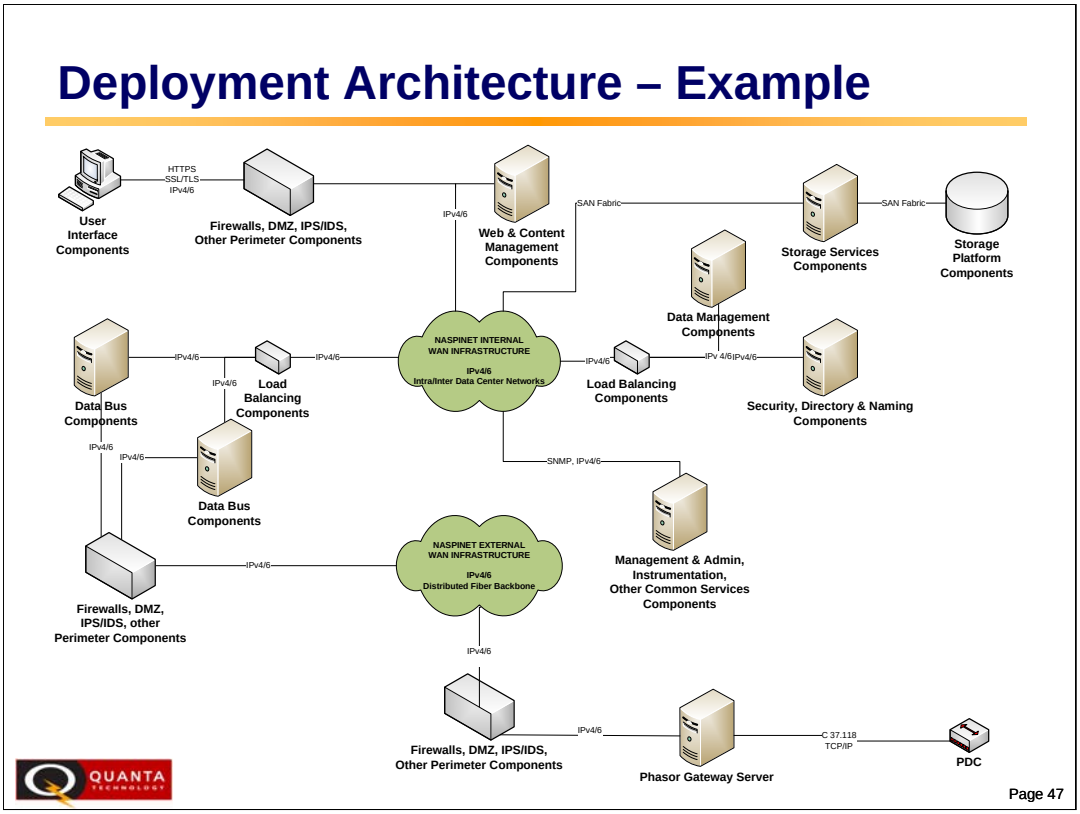


This diagram is intended to show the representative software stack of each of the architecture components starting at the operating system level, indicating all typical software components that will be utilized to form these higher level components. We take a J2EE approach and base the components on Unix/Linux.

The common services illustrate the use of other open source or commercial products as well such as Oracle RAC, Tivoli and LDAP Directory Services, and common technologies (not products) to provide specific functions such as security:

•Authentication: 1 factor, 2 factor, PKI; and

•Confidentiality: payload/channel encryption, AES2/5, etc.



This diagram shows an <u>example</u> physical deployment of the architecture with logical components mapped onto hardware components. This view will vary depending on the implementer's affinity for use of hardware virtualization, clustering and SAN technologies. We expect the implementation bidders to provide their recommended physical system architectures for production, QA/Test, staging, and development, etc.

NASPInet Specifications

- Defines NASPInet requirements at a high level, focusing on “what” is needed, not “how” to build it
  - Used to solicit detailed design and firm price quotations from qualified system vendors
  - Not intended to be a detailed design document that someone can build from
- Separate specifications required for Phasor Gateway, Data Bus, and NASPInet Common Service components
- Will allow for “tailoring” to address unique local requirements
- Commercial T’s/C’s and Form of Proposal by individual procuring entities

Specification Contents

- **Project Overview** (NASPInet summary, purpose of RFP document)
- **Scope of Supply** (vendor responsibilities, procuring entity responsibilities)
- **Functional Requirements** (what the component is expected to do (now "how" it should be done) – this will generally follow the NASPInet use cases)
- **System Requirements** (performance and availability requirements, data rates, latency, capacity and expansion, adherence to standards)
- **Architectural Requirements** (description of overall conceptual architecture (framework), requirements for system logical components (PG - ingest, device management, distributor engine, etc; DB – streaming services, API/SDK services, directory and name services, etc.); security management
- **Communication and Networking Requirements** (physical locations of communicating entities, quality of service requirements, data rates, latency, availability, etc.)
- **Hardware** (required characteristics of physical hardware)
- **Software** (operating systems, on-line diagnostics, system admin tools, etc)
- **Quality assurance and testing** (factory and field testing, third party certification)
- **Documentation and Training** (functional and technical information required by end users and support personnel)
- **Implementation and Sustainment Services** (project management and key technical personnel, installation and commissioning, long range sustainment options)

QUANTA TECHNOLOGY