Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 22, 2016

F. Li
J. Pan
PetroChina Huabei Oilfield Company
L. Jiang
Y. Song
BII Group Holdings Ltd.
November 19, 2015

The Architecture for Ubiquitous Green Community Control Network
draft-li-ugccnet-architecture-00

Abstract

In this document, it identifies gateways for field-bus networks, data storages for archiving and developing data sharing platform, and application units to be important system components for developing digital communities: i.e., building-scale and city-wide ubiquitous facility networking infrastructure. The standard defines a data exchange protocol that generalizes and interconnects these components (gateways, storages, application units) over the IPv4/v6-based networks. This enables integration of multiple facilities, data storages, application services such as central management, energy saving, environmental monitoring and alarm notification systems.

Table of Contents

1. Introduction

This document identifies gateways for field-bus networks, data storages for archiving and developing data sharing platform, and application units such as for providing user interfaces of analysis

and knowing the environmental information to be important system components for developing digital communities: i.e., building-scale and city-wide ubiquitous facility networking infrastructure. The standard defines a data exchange protocol that generalizes and interconnects these components (gateways, storages, application units) over the IPv4/v6-based networks. This opens the application interface to handle the statuses of multi-vendor facilities on a generalized digital infrastructure. The standard assumes distributed operation of the infrastructure by multiple service providers and integrators, and defines a component management protocol that autonomously interoperates such distributed infrastructure. Security requirements are taken into consideration in this standard to ensure the integrity and confidentiality of data.

2. Terminology and Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

- access control: The means to allow authorized entry and usage of resources.

- actuator: A transducer that accepts a data sample or samples and converts them into physical action.

- eXtensible Markup Language (XML) namespace: A method for distinguishing XML elements and attributes that may have the same name but different meanings. A URL is used as a prefix to a "local name." This combination ensures the uniqueness of the element or attribute name. The URL is used only as a way to create a unique prefix and does not have to resolve to a real page on the Internet.

- A transducer that converts a physical, biological, or chemical parameter into a digital signal.

- universally unique identifier (UUID): An identifier that has a unique value within some defined universe. In this standard, the query-expression and lookup-expression of transport data structure has a UUID unless otherwise stated.

3. System Architecture

This protocol specification applies to a TCP/IP-based facility networking architecture. One of the main goals of this specification is to enable interoperability among facility networking components. Thus, GW, Storage and APP, what we call "Component" in this document,

have the same generalized communication interface. Registry has different communication interface with these components. A Component works as a part of data- plane, and a Registry works as a part of control- plane.

In the networking environment, Registry works as a broker of Components. It manages meta information e.g., the role of each component and the semantics of Point ID, in order to bind components appropriately and autonomously. We here describe them in more detail and show how they collaborate with each other.

3.1. Gateway

Gateway component has physical sensors and actuators. It generalizes the data model and the access method for those devices, encapsulating each physical (field-bus) data model and access method. It acts on its actuator according to the written value from a component (e.g., APP), and it provides physical sensor readings for other components (e.g., Storage and APP).

3.2. Storage

Storage component archives the history of data sequences. The written values from other components should be permanently stored in the backend disks. It provides the archived values to the components that have requested them.

3.3. Application

APP component provides some particular works on sensor readings and actuator commands. It can have user interface to display the latest environmental state. It can also allow a user to input some schedules of actuator settings, and it can as well analyze some sensor data in realtime and provide the result as a virtual device.

3.4. Registry

The Registry works as a broker of GW, Storage, and APPs. The main role of registry is to bind those components appropriately and autonomously. It is separated from the data-plane. It does not work on sensor readings or actuator settings directly. It should allow system operation without Registry.

4. Concerns of the network design

All components can behave both as the TCP (IETF RFC 793) initiator and receiver at once. It implies the components should be put on a flat network. A flat network means there are no middle boxes which

could disturb bi-directional communications such as NAT routers and firewalls.

To avoid the issue, we strongly recommend building IPv6 (IETF RFC 2460) network. There could be other solutions than IPv6 such as http proxy based or NAT traversal solutions. However, they would depend on the requirements of the network configuration. This specification does not reject such solutions, but they are required to make interoperable with other systems and not to disturb the specification.

5. System Model

Component is the basic unit for all the GWs, Storages, and APPs. The interface of Component provides data and query method. GW, Storage, and APP are the inherited classes of Component. Thus, they have the same interface (i.e., data and query method), and they communicate with each other using the same protocol. Here, Query is a method for retrieving data (including event-based data transfer) from Component; Data is a method for pushing data into Component.

Registry works as a broker of Components with another type of interface (i.e., registration and lookup method). The interface of Registry provides registration and lookup method. Here,

- Registration is a method for registering the role of components and semantics of Points.

- Lookup is a method for finding appropriate components and Points.

Typical implementations for GWs, Storages, APPs and Registry would be:

GW implementations encapsulate field-buses and provide INPUT/OUTPUT access for physical devices (by query and data method).

Storage implementations archive the history of data posted by data method, and provide the historical data by query method.

APP implementations provide other functionalities. For example, they can have user interface. Data processing component must be also categorized to an APP implementation.

Registry implementations manage the relationships between Point ID and components, provide registration of the role of components and semantic of Points by registration method, and provide inquiry of appropriate components and Points by lookup method.

This generalization enables open development of facility networking components (i.e., GWs, Storages and APPs) by any vendors. And we would deploy facility networking systems for customer buildings without customized programming, by binding these developed components.

The role of Registry is to increase the autonomousness of component-to-component collaboration. It allows autonomous collaboration of components, by sharing the information of component roles in an operational domain (in fact, not only in an operational domain but also with other external domains).

6. Point

6.1. Introduction

This section introduces the concept of Point. A Point shall have an URI-based globally unique identifier. It identifies a dataflow that exchanges data (i.e., sensor readings, actuator commands and meta-control signals) among components.

6.2. Definition

A Point is an elemental message channel for a specific data sequence among Components. A sequence of sensor readings, actuator commands and others (e.g., virtualized sensor readings, meta-control signals) shall be bound to a Point. We denote a message in a Point (whether it is coming from a sensor or it is outgoing to an actuator) by value. Any object type is allowed for values in a Point.

Delivery of values among components shall be made by invoking other components' interface. The provided methods are:

Query: to read objects from specified Points

Data: to write objects into specified Points

By using these methods, a component can get data of the specified Points from another component, and it can also transfer data to another component with specification of the Points.

6.3. URI-based identification

A Point is associated to a globally unique data sequence. The data shall have been generated from a specific sensor or to a specific actuator in the world. Thus, in order to identify the data sequence globally, each Point should have a globally unique identifier. Note

that for private operation, it does not necessarily need to be globally unique. However, it is not recommended.

In UGCCNet, every Point shall have a URI for its identifier. Practically, we will first assign IDs for physical sensors and actuators, and then we will use the IDs for the Point IDs. This operation goes well with the traditional facility networking operation.

Taking URI for identifiers enables global access (if the Point is public) to the Point. Let X(=http://gw.foo.org/sensor1) be a Point ID.

If components do not know the registry server that manages the Point ID (X), they should try to access X directly. Then, the URI can redirect to the registry server. If components already know the registry server for X, Point ID may not need to be reachable. However, in order to obtain operational consistency, the host of the URI should be the host name of the GW (because physical sensors and actuators are attached to the GW). Thus, typical URI format should be:

point ID = "http://(GW host name)/(any format to identify the Point in the GW)"

6.4. PointSet

This specification also defines PointSet to enable hierarchical management of Points. A PointSet aggregates multiple Points and multiple PointSets. This definition allows the conventional operation of grouping of Points hierarchically. However, PointSet feature is optional. All the components should allow operation without pointSet.

7. Common communication protocol

7.1. General

This specification defines two types of communication protocols for components and registry, including the component-to-component communication protocol and component-to-registry communication protocol. The protocol message for component-to-component and component-to-registry communication is intended to use Simple Object Access Protocol (SOAP Version 1.2 Part 1: Messaging Framework").

7.2. Component-to-component communication protocol

7.2.1. Types of component-to-component communication protocol

This section specifies and describes the following three types of sub-protocols for component-to-component communication. Note that instances of components are GWs, Storages, and APPs. As for the accessing methods to a registry.

FETCH protocol -- for data retrieval from a remote component.

WRITE protocol -- for data transfer to a remote component.

TRAP protocol -- for event query registration and event data transfer.

7.2.2. FETCH protocol

FETCH is a protocol for data retrieval from a remote component. We here denote the component that inquires data from the remote component by 'Requester', and the component that replies with the data by 'Provider'.

7.2.3. WRITE protocol

WRITE is a protocol for data transfer to a remote component. We denote the component that submits data to the remote component by Requester, and the component that receives the data by Target.

7.2.4. TRAP protocol

TRAP is a protocol for event query registration and event data transfer. We here give names for components in the following manner.

Requester -- the component that sets event-based query to Provider.

Provider -- the component that transmits data when it has received query-matching updates.

Callback (Data) -- the components that receives data from the Provider.

Callback (Control) -- the components that receives control signals from the Provider.

This subsection provides the definition of collaboration among these components. Though the roles are explicitly categorized in general,

in most of the practical systems, Callback (Data), Callback (Control) and Requester will be the same component.

7.3. Component-to-registry communication protocol

7.3.1. Type of component-to-registry communication protocol

This section specifies the following two types of sub-protocols for component-to-registry communication.

REGISTRATION Protocol -- for registration of the role of components and semantics of Points.

LOOKUP Protocol -- for searching appropriate components and Points.

7.3.2. REGISTRATION protocol

REGISTRATION is a protocol which enables a component to register the role of components and semantics of Points. We denote the component that submits registration request to its Registry by "Registrant".

7.3.3. LOOKUP protocol

LOOKUP is a protocol for a component to search appropriate access components (for component-to-component communication), and to search Points by semantic-query. We here denote the component that searches appropriate components and Points from its Registry by 'Requester'.

8. Security Considerations

UGCCNet protocol is basically open. It assumes multi-domain operation and public access from other domain's system components. In this context, security requirements to the system would be listed as follows:

- To avoid unintended data disclosure to the public.
- To avoid unauthorized access to writable resources.
- Availability and confidentiality of remote communication host.
- Integrity and confidentiality of data.
- To avoid unintended access or operational conflicts.

To get confidentiality of remote communication host, we would be able to take VPN, SSL, SSH and other related technologies. HTTPS, or SIP

and its security extension would help in getting integrity and confidentiality of data.

Access control and access confliction management shall be other important but different types of security issues that should be discussed independently. Generally, access control is used to allow only specific users to access both readable and writable resources, which would certainly help to avoid unauthorized access from or unintended data disclosure to the public (sometimes anonymous) users. In order to manage this, the system would need to introduce the concept of users to identify who is accessing the resources. We assume URI-based identification for user authentication just as Point ID takes URI for its identifier. Authentication of these users and components (probably by taking advantage of the existing authentication platforms) would certainly need to be considered.

9. Acknowledgements

Funding for the RFC Editor function is currently provided by PetroChina Huabei Oilfield Company and BII Group.

10. Normative References

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://www.rfc-editor.org/info/rfc793>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <http://www.rfc-editor.org/info/rfc3986>.

Authors' Addresses

Fengmin Li
PetroChina Huabei Oilfield Company
Renqiu
P. R. China

Email: wty_lfm@petrochina.com.cn

Juchen Pan
PetroChina Huabei Oilfield Company
Renqiu
P. R. China

Email: tx_pjc@petrochina.com.cn

Lianshan Jiang
BII Group Holdings Ltd.
Beijing
P. R. China

Email: lsjiang@biigroup.cn

Yang Song
BII Group Holdings Ltd.
Beijing
P. R. China

Email: ysong@biigroup.cn