Pacific Northwest
NATIONAL LABORATORY

Proudly Operated by **Battelle** *Since 1965*

PNNL VOLTTRON^{TM} Application Development

Srinivas Katipamula

DOE Building Technologies Office: Technical Meeting on Software Framework for Transactive Energy
July 23-24, 2014

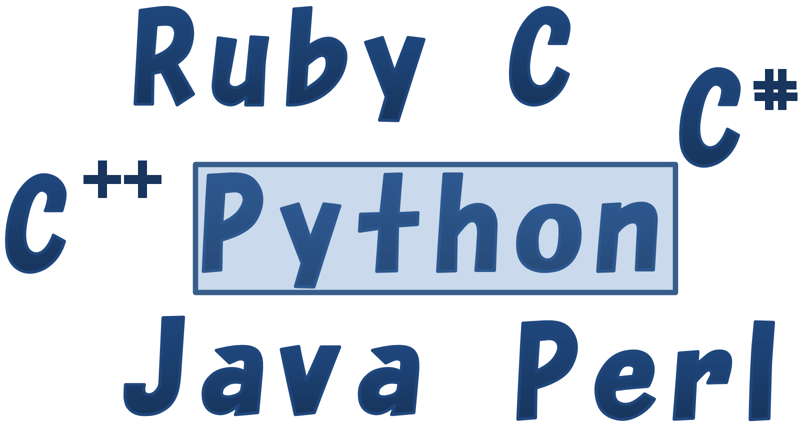
U.S. DEPARTMENT OF
ENERGY

Presentation Outline

- Application Development Environment
- PNNL Developed VOLTTRON Apps
- VOLTTRON Services Used by PNNL Apps
- Walkthrough of an Application Development
- Lessons Learned
- Concluding Remarks

Application Development Environment and Language Support

- VOLTTRON is a native Linux application
  - Can be run on PC and MAC using virtual machine (VM)
  - VirtualBox by Oracle is flexible free VM software
- VOLTTRON allows application and development flexibility
  - Applications can be developed in nearly any software language giving developers increased flexibility
  - Any application dependencies can be packaged with the application (i.e., external software libraries) to simplify deployment for end users
  - All PNNL applications were developed in Python 2.7



Ruby C C#

C++ Python

Java Perl

Recommended Development Environment

- Eclipse IDE (integrated development environment) is not required for agent development, but it can be a powerful developmental tool
  - Can be downloaded - http://www.eclipse.org/
- Useful Eclipse plug-ins
  - **EGit** integrates Git source control with Eclipse - http://download.eclipse.org/egit/updates
  - **Pydev** support Python programming, code refactoring, debugging, code analysis and many other helpful feature - http://pydev.org/updates
- For more details on how to use and configure Eclipse with VOLTTRON to create applications refer to - http://buildingsystems.pnnl.gov/documents/buildinggrid/PNNL-23182.pdf

Presentation Outline

Application Development Environment

PNNL Developed VOLTTRON Apps

VOLTTRON Services Used by PNNL Apps

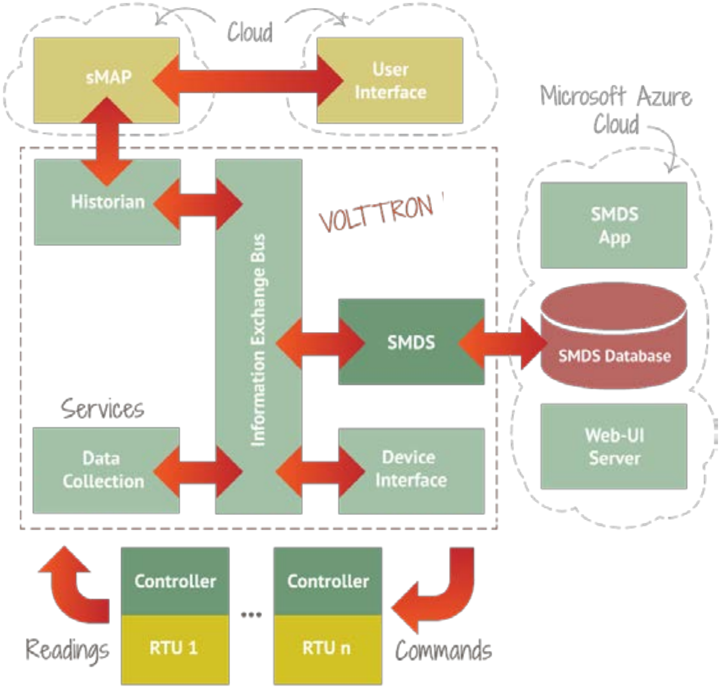
Walkthrough of an Application Development

Lessons Learned

Concluding Remarks

PNNL VOLLTTRON Applications

- **Demand Response Agent**
  - Make rooftop units (RTUs) grid responsive
- **Automatically Detect and Diagnose Faults for RTUs**
  - Detect economizer and ventilation failures as they occur and notify building operator to correct them
  - Smart monitoring and diagnostics system for conditioned-based maintenance service
- **Intelligent Duty Cycling**
- **Embedded Advanced RTU Controls – Partner Solution**
  - Improve operational efficiency of RTUs through use of advanced RTU controls leading to energy and carbon emission reductions over 50%



Pacific Northwest NATIONAL LABORATORY

Proudly Operated by Battelle *Since 1965*

Presentation Outline

Application Development Environment

PNNL Developed VOLTTRON Apps

VOLTTRON Services Used by PNNL Apps

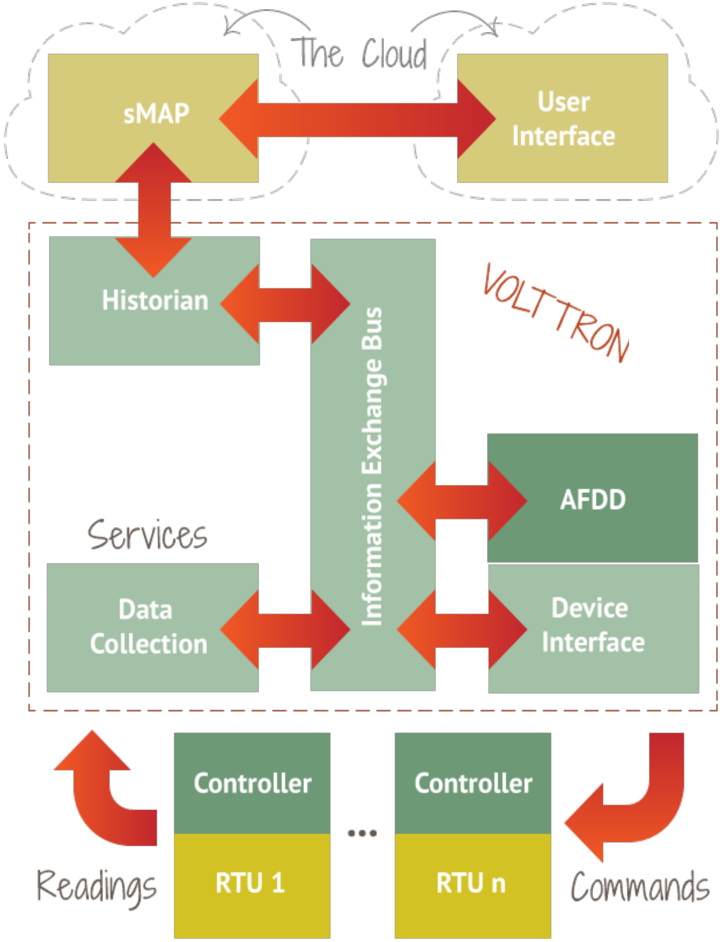
Walkthrough of an Application Development

Lessons Learned

Concluding Remarks

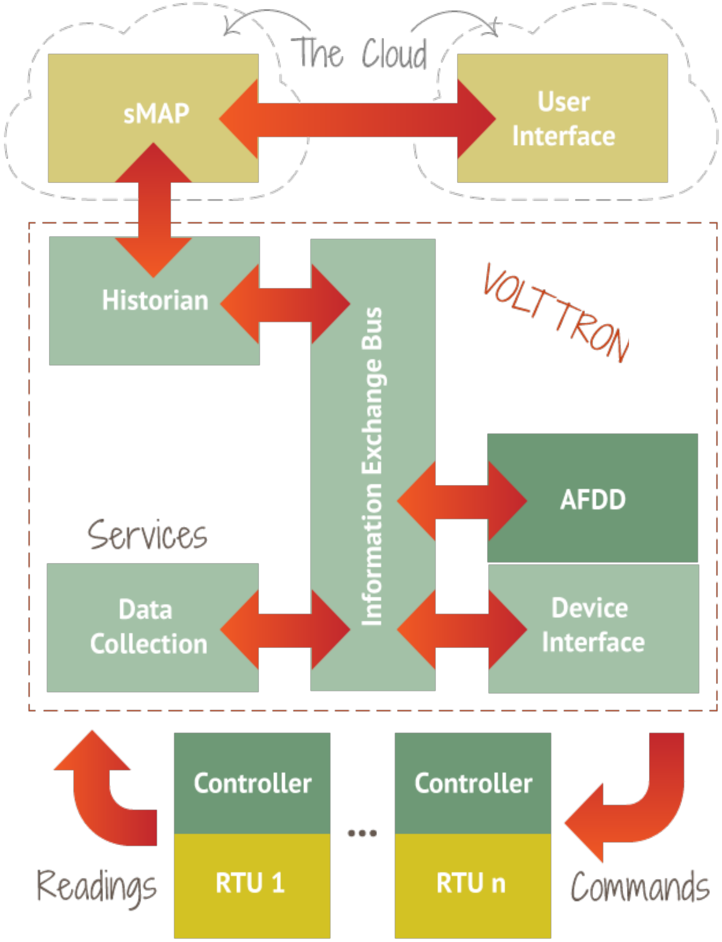
VOLTTRON Platform Services Used

- Device drivers – BACnet and MODBUS drivers
- Message bus – Information (data and messages) exchanged between applications and services thru the VOLTTRON message bus using Publish/Subscribe mechanism
- sMAP – Data is stored in a simple measurement and actuation protocol (sMAP) data historian
  - sMAP is a fast open source time series database, **but can't store non-numeric data**



VOLTTRON Platform Services Used – Device Control

- BACnet/MODBUS driver publishes data from devices to the platform and also stores the data in the sMAP historian
- Actuator agent
  - **Device control**: Actuator agent will accept commands from applications and issue the commands to the specified device
  - **Device access scheduling**: Allows scheduling of agents' access to devices to prevent multiple agents from controlling the same device at the same time
    - Supports time-of-day, day-of-week and priority



Presentation Outline

Application Development Environment

PNNL Developed VOLTTRON Apps

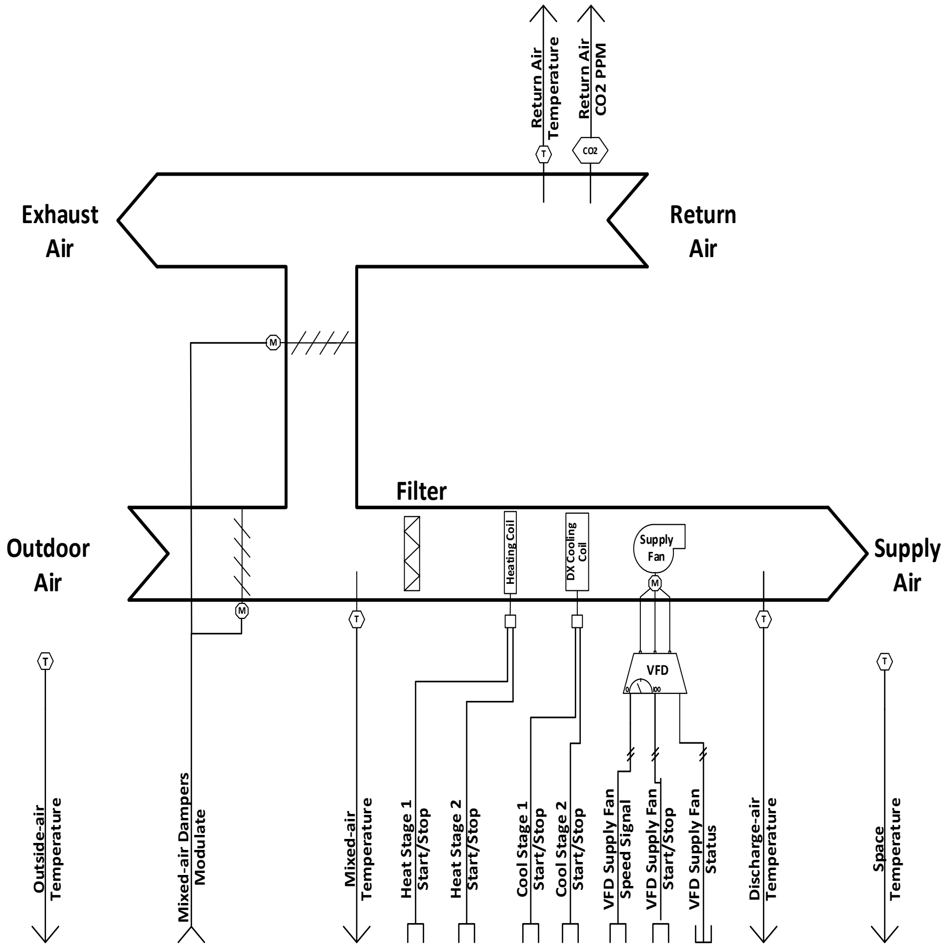
VOLTTRON Services Used by PNNL Apps

Walkthrough of an Application Development

Lessons Learned

Concluding Remarks

Application Development - RTU Proactive AFDD Application



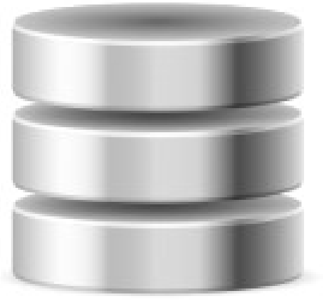
Automated Fault Detection and Diagnostic (AFDD) Capabilities:

- Comparing discharge-air temperature with mixed-air temperature (AFDD0)
- Checking damper modulation (AFDD1)
- Sensor faults (outdoor-, mixed- and return-air temperature) (AFDD2)
- Not economizing when RTU should (AFDD3)
- Economizing when RTU should not (AFDD4)
- Excess outdoor air (AFDD5)
- Inadequate outdoor ventilation air (AFDD6)

Unique: Diagnostics algorithms will initiate proactive tests **on schedule** (e.g., commanding damper, etc.)

Identifying Needs of the AFDD Application

- **First, identify the list of sensor values and command outputs necessary for application**
  - Identify the protocol to use (MODBUS or BACnet)
  - Identify other data sources
    - Weather information from the VOLTTRON Weather Agent or other application within VOLTTRON
    - Other data sources
- Configure the data points



Database

Internet

Pacific Northwest NATIONAL LABORATORY

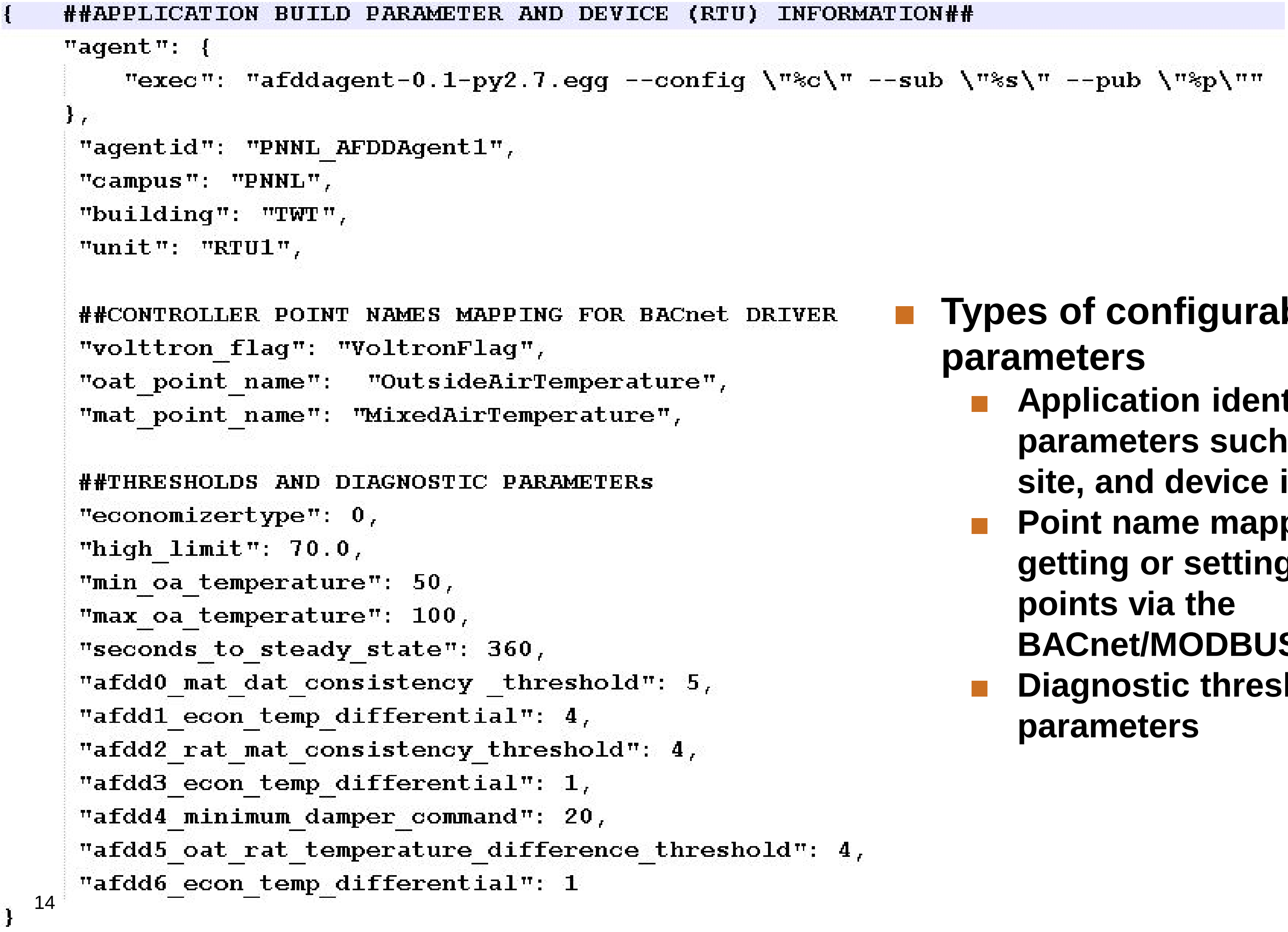
Proudly Operated by Battelle *Since 1965*

Application Development: Communication and Control

- Communication and control of BACnet devices
  - VOLTTRON provides a network discovery tool as well as a device configuration tool
- BACnet discovery provides a list of BACnet devices on a network (Built on BACnet Who-Is and I-AM)
- BACnet configuration tool will generate a usable configuration file that will allow applications to communicate with the device
- Ensure that the control points on the device are writeable
  - Auto discovery may say some points are writeable, even though they are not
  - Performing simple test could be useful
  - May need BACnet vendor software to enable write access to the desired points

Configuring the AFDD Application

- Applications utilize a JSON style configuration file



```
{    ##APPLICATION BUILD PARAMETER AND DEVICE (RTU) INFORMATION##
    "agent": {
        "exec": "afddagent-0.1-py2.7.egg --config \"%c\" --sub \"%s\" --pub \"%p\""
    },
     "agentid": "PNNL_AFDDAgent1",
     "campus": "PNNL",
     "building": "TWT",
     "unit": "RTU1",

     ##CONTROLLER POINT NAMES MAPPING FOR BACnet DRIVER
     "volttron_flag": "VoltronFlag",
     "oat_point_name":  "OutsideAirTemperature",
     "mat_point_name": "MixedAirTemperature",

     ##THRESHOLDS AND DIAGNOSTIC PARAMETERs
     "economizertype": 0,
     "high_limit": 70.0,
     "min_oa_temperature": 50,
     "max_oa_temperature": 100,
     "seconds_to_steady_state": 360,
     "afdd0_mat_dat_consistency _threshold": 5,
     "afdd1_econ_temp_differential": 4,
     "afdd2_rat_mat_consistency_threshold": 4,
     "afdd3_econ_temp_differential": 1,
     "afdd4_minimum_damper_command": 20,
     "afdd5_oat_rat_temperature_difference_threshold": 4,
     "afdd6_econ_temp_differential": 1
}
```

- Types of configurable parameters
  - Application identifying parameters such as agent id, site, and device information
  - Point name mapping for getting or setting device points via the BACnet/MODBUS driver
  - Diagnostic thresholds and parameters

VOLTTRON Platform Service - Scheduling

- To schedule active control of a device an application must publish a schedule request on the message bus with the topic
  - Example topic format - “**RTU/actuators/schedule/request**”

- Components of schedule request
  - Request type (NEW_SCHEDULE, CANCEL_SCHEDULE)
  - Requestor ID (typically agent ID)
  - Task ID (Unique task identifier)
  - Task priority
  - Device (Typically in the form “**campus/building/device**”)
  - Start time and end time of requested schedule block

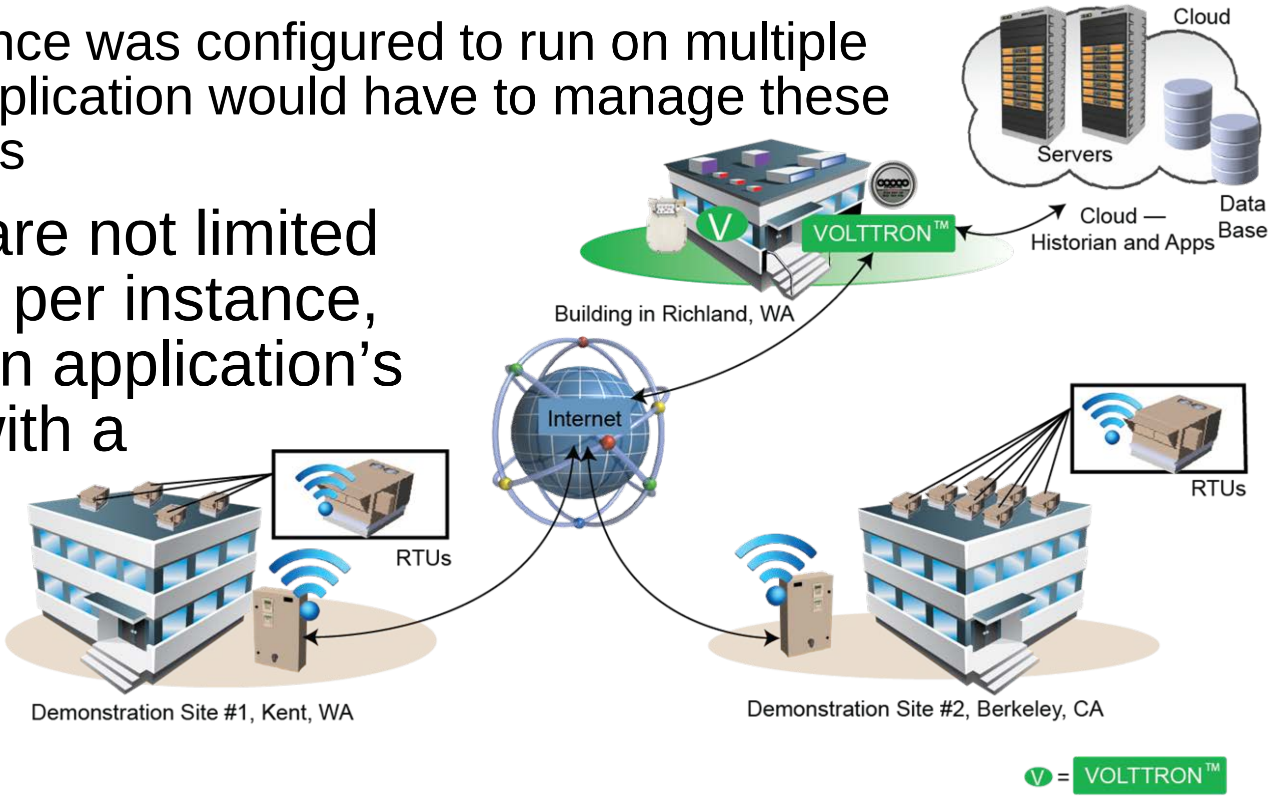
VOLTTRON Platform Services – Scheduling (Cont.)

- Scheduling a block of time for applications is first come first serve, but employs the following priority schema
  - **HIGH** – High priority applications cannot be pre-empted under any circumstance, but they can preempt other applications that use of LOW_PREEMPT priority
  - **LOW** – Low priority applications cannot be preempted **once they start device control**
    - Considered started once the earliest time slot on any device has been reached
    - **Cannot** preempt other tasks
  - **LOW_PREEMPT** – Low preempt priority applications can be preempted at any time
    - Applications are given a grace period to "clean up" interactions with the devices before being revoked (e.g. 120 seconds, configurable)
    - **Cannot** preempt other tasks
- For more details on formatting schedule request visit

https://github.com/VOLTTRON/volttron/wiki/ActuatorScheduleRequest

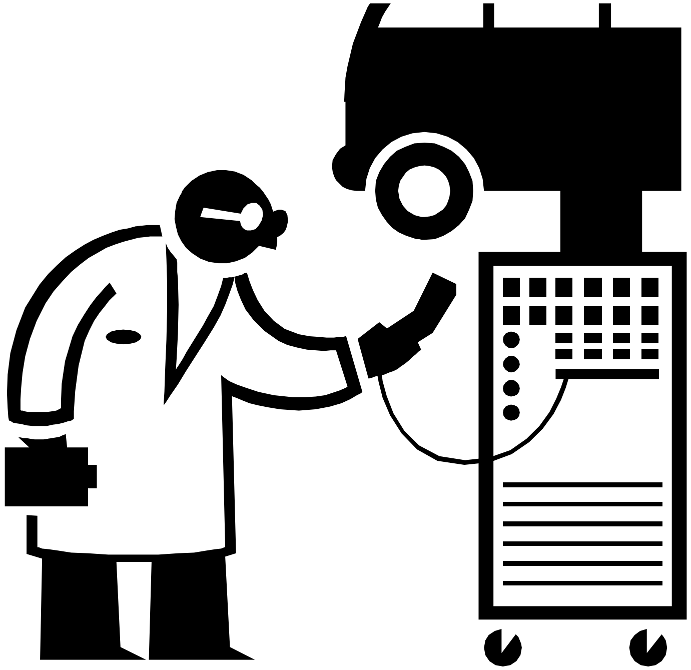
How do we to Handle Multiple RTUs?

- Each RTU has a
  - Separate instance
  - Each instance is configured to monitor, control, and produce diagnostic results for one RTU
  - Some diagnostics are contingent on earlier diagnostics being fault free
  - If each instance was configured to run on multiple RTUs the application would have to manage these contingencies
- Applications are not limited to one device per instance, especially if an application's interactions with a device are passive


Cloud
Servers
Data Base
Cloud — Historian and Apps
VOLTTRON™
Building in Richland, WA
Internet
RTUs
RTUs
Demonstration Site #1, Kent, WA
Demonstration Site #2, Berkeley, CA
V = VOLTTRON™

Testing: Test Early and Test Often

- Test device and sensor communication early in the development
- Perform unit tests on manageable pieces of code. This will make debugging and correcting any problems easier
- Test after any software change, data I/O change, or device change



Presentation Outline

Application Development Environment

PNNL Developed VOLTTRON Apps

VOLTTRON Services Used by PNNL Apps

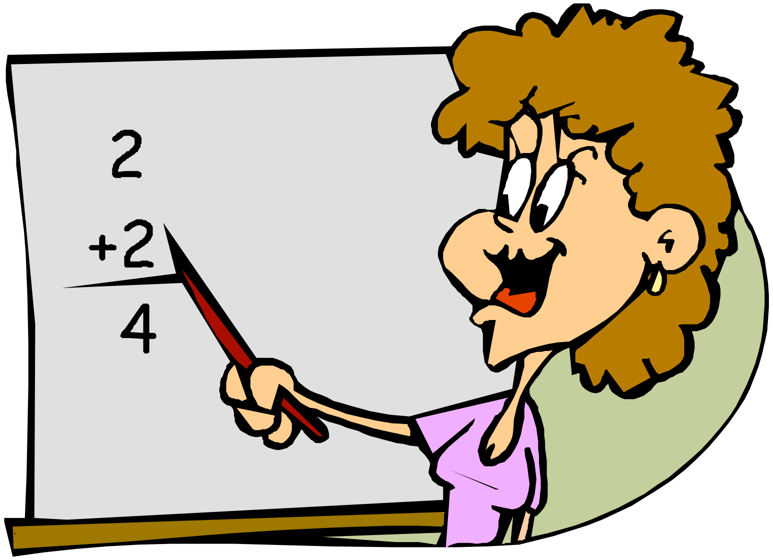
Walkthrough of an Application Development

Lessons Learned

Concluding Remarks

Lessons Learned

- During the proactive fault diagnostic an unexpected occurrence led to problems
  - PNNL applications and VOLTTRON service agent updates were performed
  - Application updates were performed first and the proactive diagnostic was restarted
  - Update coincided with the scheduled daily proactive diagnostics for the test site
- AFDD application started the scheduled proactive diagnostics immediately after the application was updated and restarted



Lessons Learned (cont.)

- AFDD application requested a device lock from the actuator agent
- AFDD application received the device lock and commanded the outdoor–air damper fully open (this occurred in mid-November)
- Next, the actuator agent was updated and restarted
- When the AFDD attempted to continue with the diagnostic, the actuator agent would not allow the AFDD application to modify the device controls
- When the actuator agent restarted, it had no recollection of any previous approved device interactions

Lessons Learned (cont.)

- Consequences
  - When occupants arrived in the morning, the room temperatures in the building were near 50ºF
  - Because the occupants had an override, RTUs were returned to normal operations
- Steps taken
  - Actuator agent was updated to store all previous device interactions and approved device lock requests
  - If the actuator agent was restarted for any reason, all scheduled device interactions would be saved
  - AFDD application was updated to handle the unexpected loss of a device lock

Presentation Outline

Application Development Environment

PNNL Developed VOLTTRON Apps

VOLTTRON Services Used by PNNL Apps

Walkthrough of an Application Development

Lessons Learned

Concluding Remarks

Summary

- 11 RTUs are being monitored and control for over1 year, with very little “supervision”
- Seamless interaction of data and information between applications and devices has been a “myth” for a long time
- What’s good?
  - VOLTTRON has shown that it has the potential to overcome the data exchange myth
  - Application development was easy and flexible
- Limitation?
  - Lack of flexibility of choosing a historian of choice
  - Debugging can be a bit cumbersome
  - Lack of management tools
  - Lack of diagnostic tools
  - No PC or Mac versions

Questions?

Srinivas.Katipamula@pnnl.gov

VOLTTRON - https://transactionalnetwork.org/
http://github.com/volttron

PNNL Developed Applications (Transactional Network)

- Smart Monitoring and Diagnostic System - http://buildingsystems.pnnl.gov/building/smds.stm
- Proactive Diagnostics - http://buildingsystems.pnnl.gov/building/afdd.stm
- Automated Demand Response - http://buildingsystems.pnnl.gov/building/adr.stm