Slides created by: Mario Berges, Anthony Rowe, Ethan Goldman

Carnegie Mellon

ICES CenSCIR

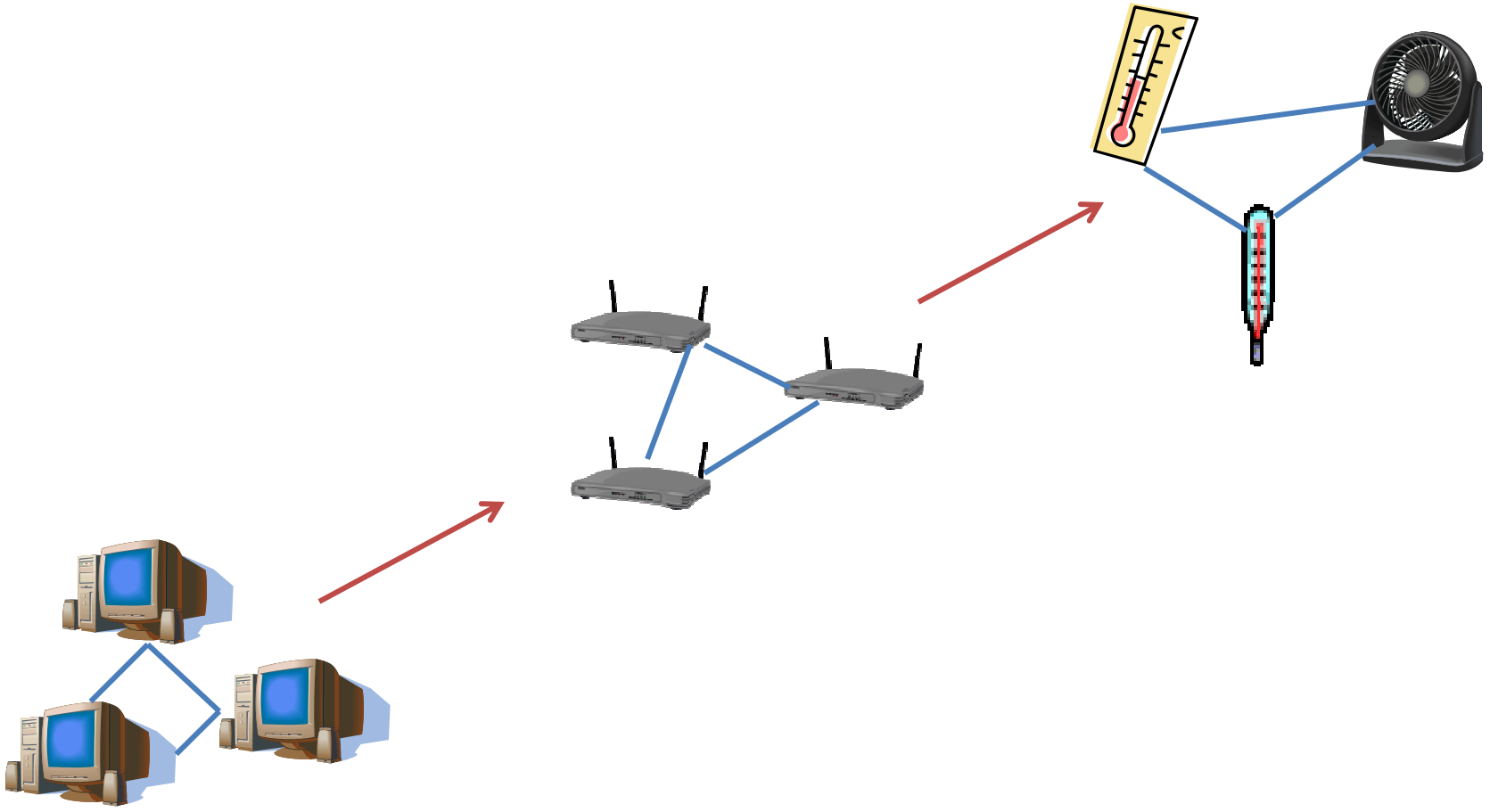
CenSCIR is leading a broad research project called Sensor Andrew

- "Living laboratory" for research
- A sensor network infrastructure that will enable the dense instrumentation of campus
- Sensor network that can support a broad set of applications
- Goal: Carnegie Mellon "the most sensed" campus

Carnegie Mellon

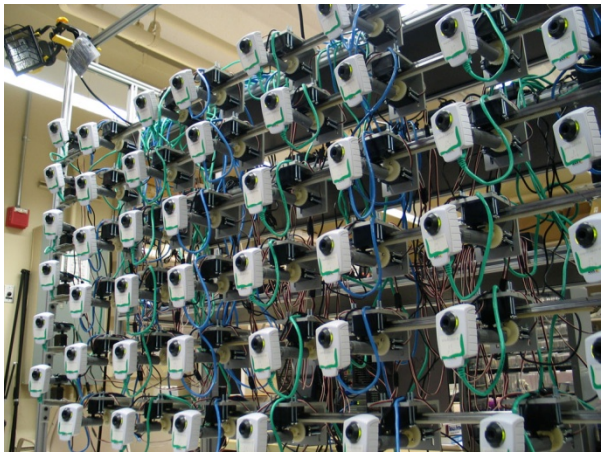
ICES CenSCIR

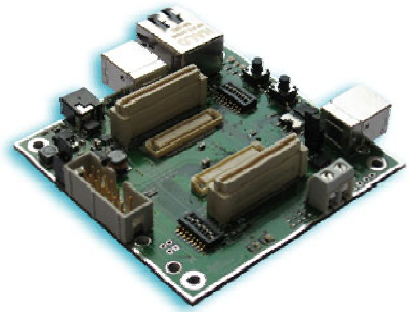
Origins

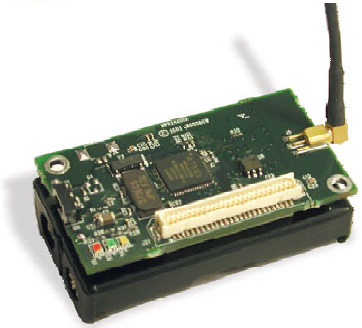


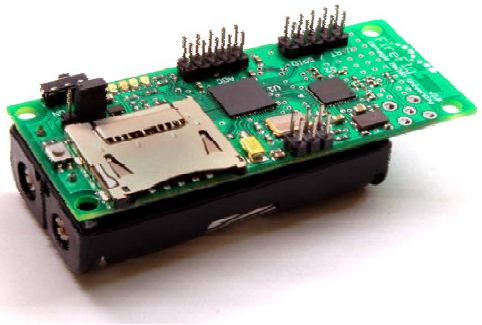
Carnegie Mellon

ICES CenSCIR









How do we share sensor-data at internet scale?

Carnegie Mellon

ICES CenSCIR

What is Sensor Andrew?

easy to use

scalable

campus-wide

sensor network

sensors

provides security

control

multi-disciplinary

sharing data

ubiquitous

infrastructure

applications

users

extensible

maintains privacy

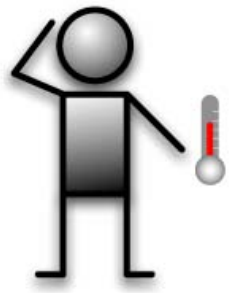
monitoring

Carnegie Mellon

ICES

CenSCIR

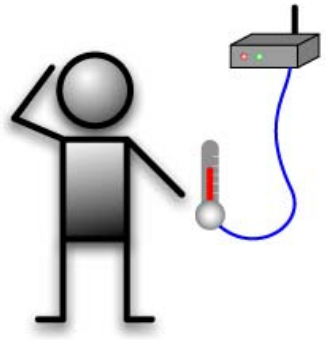
What is Sensor Andrew?

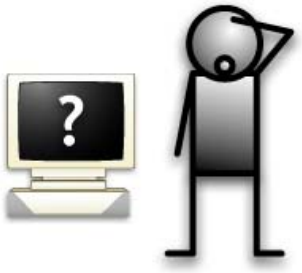


Carnegie Mellon

ICES CenSCIR

What is Sensor Andrew?

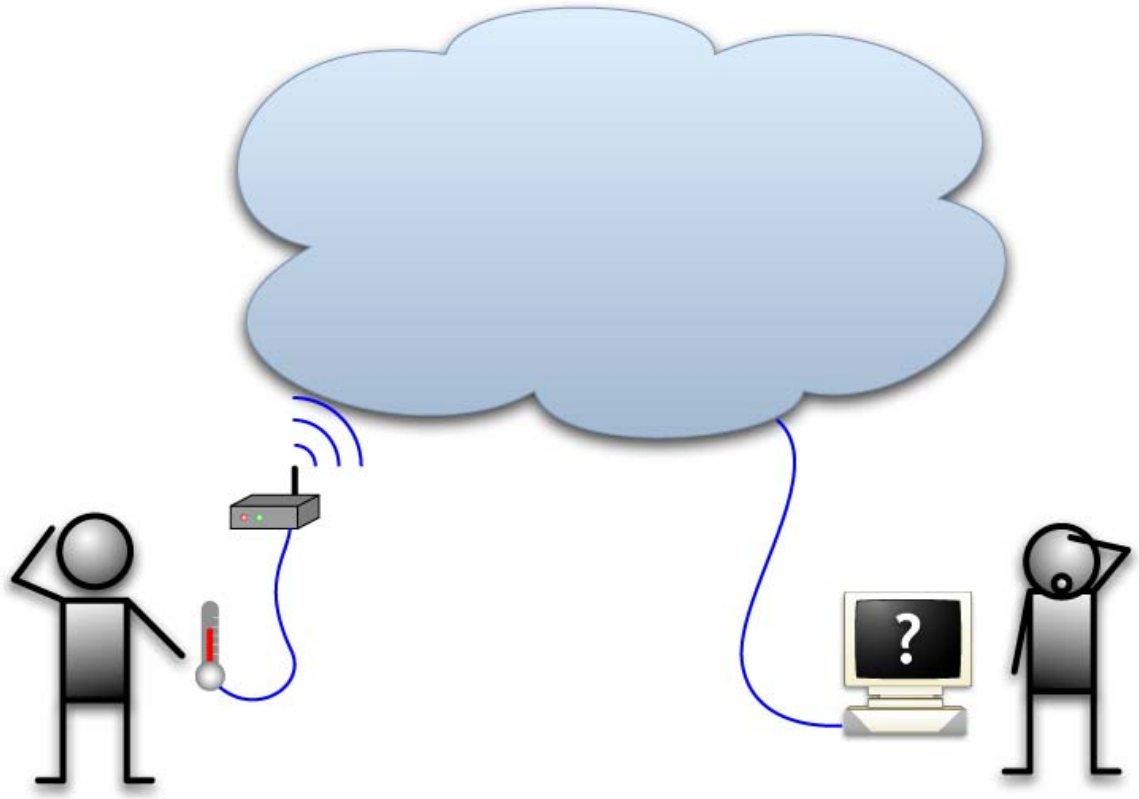




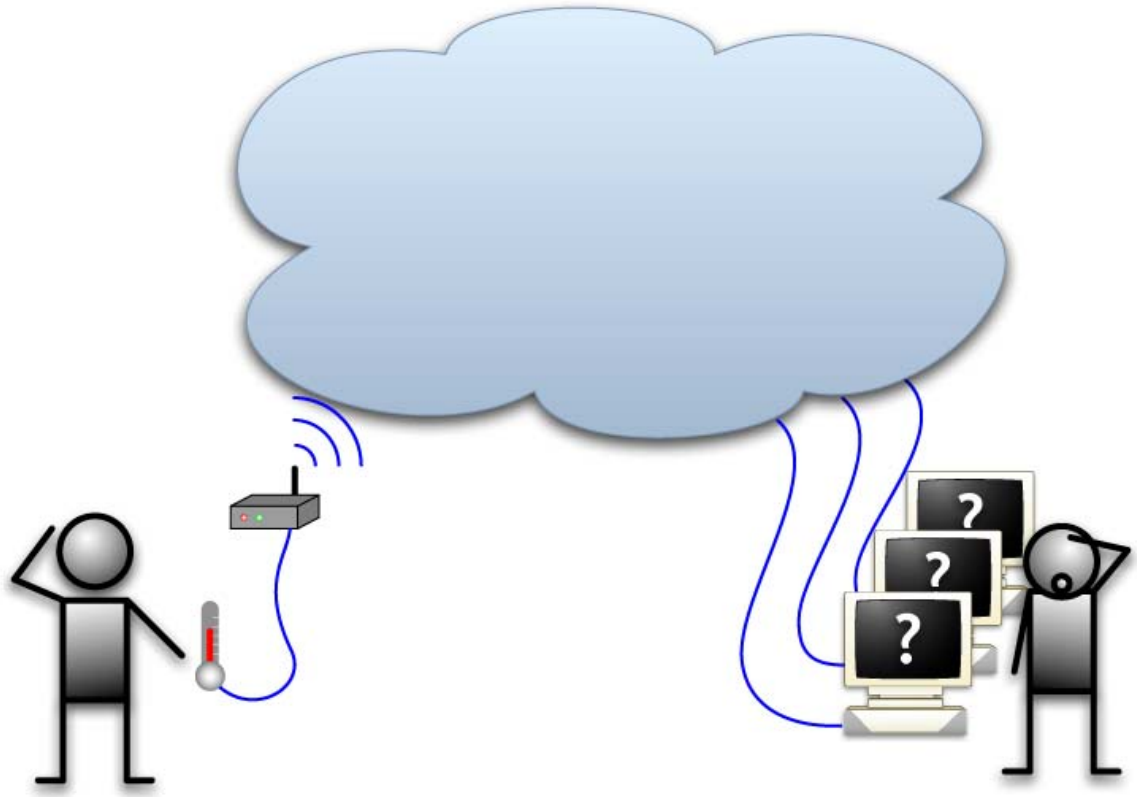
Carnegie Mellon

ICES CenSCIR

What is Sensor Andrew?



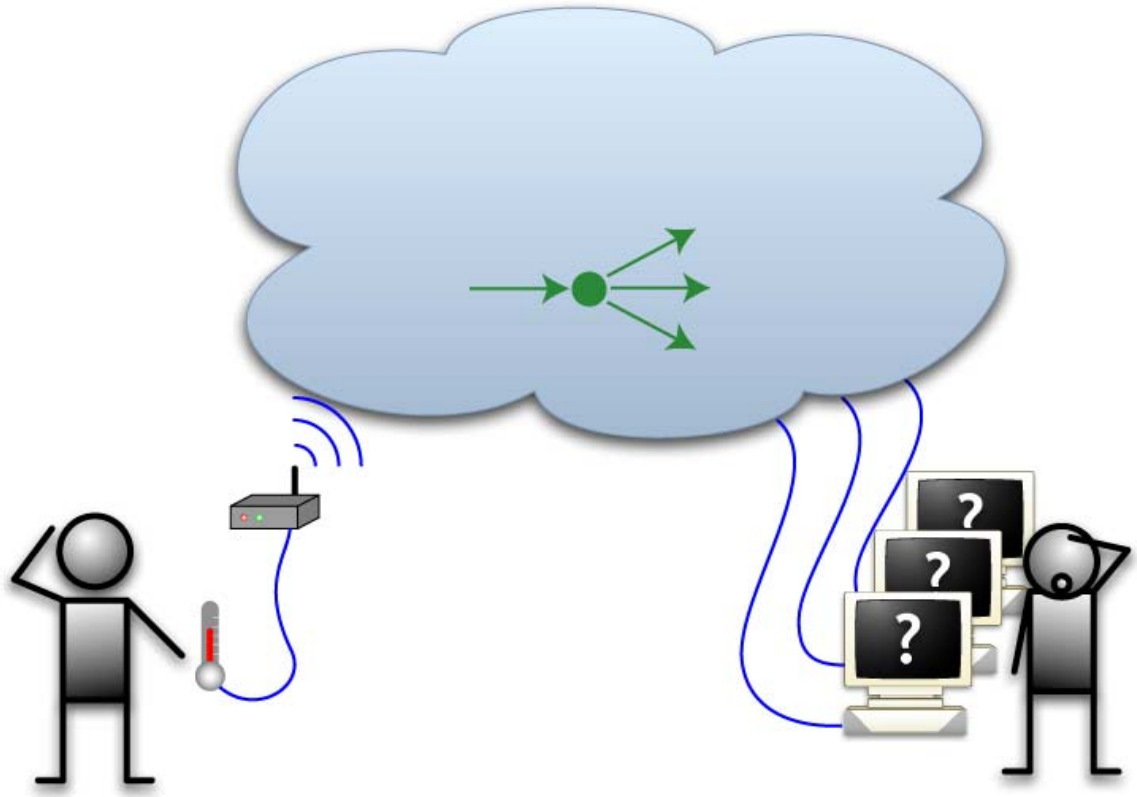
What is Sensor Andrew?



Carnegie Mellon

ICES CenSCIR

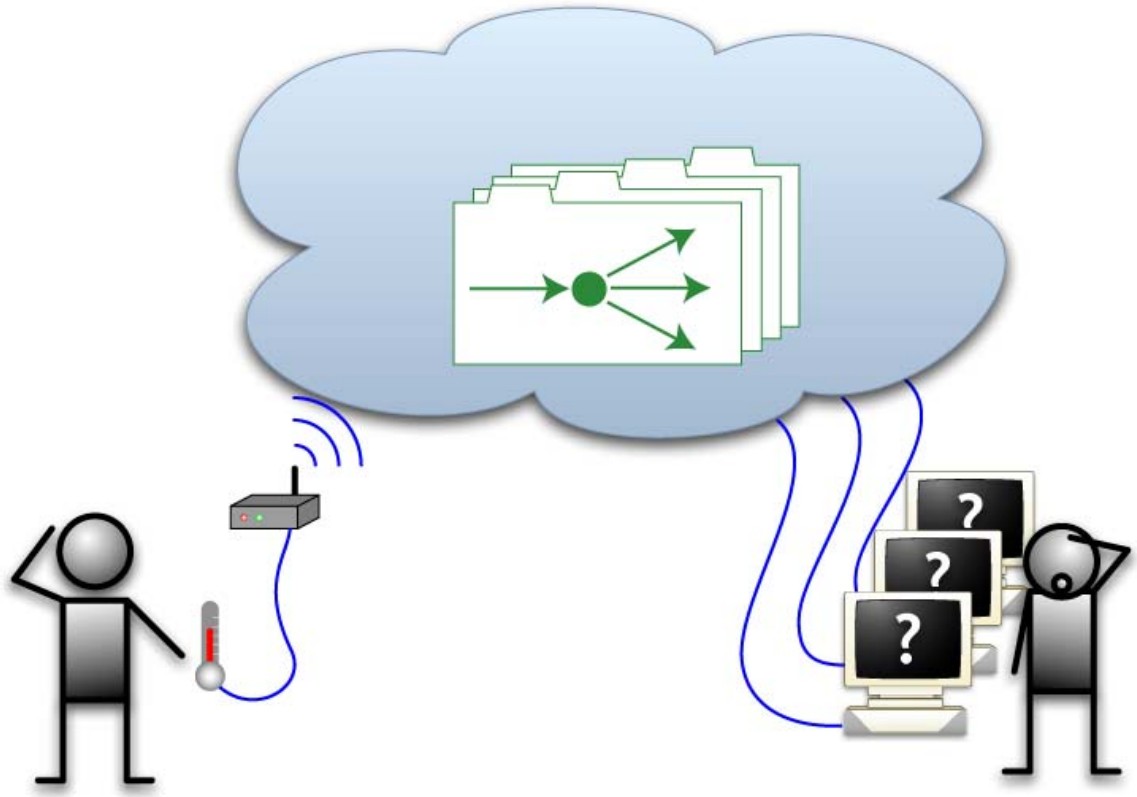
What is Sensor Andrew?



Carnegie Mellon

ICES CenSCIR

What is Sensor Andrew?



Carnegie Mellon

ICES CenSCIR

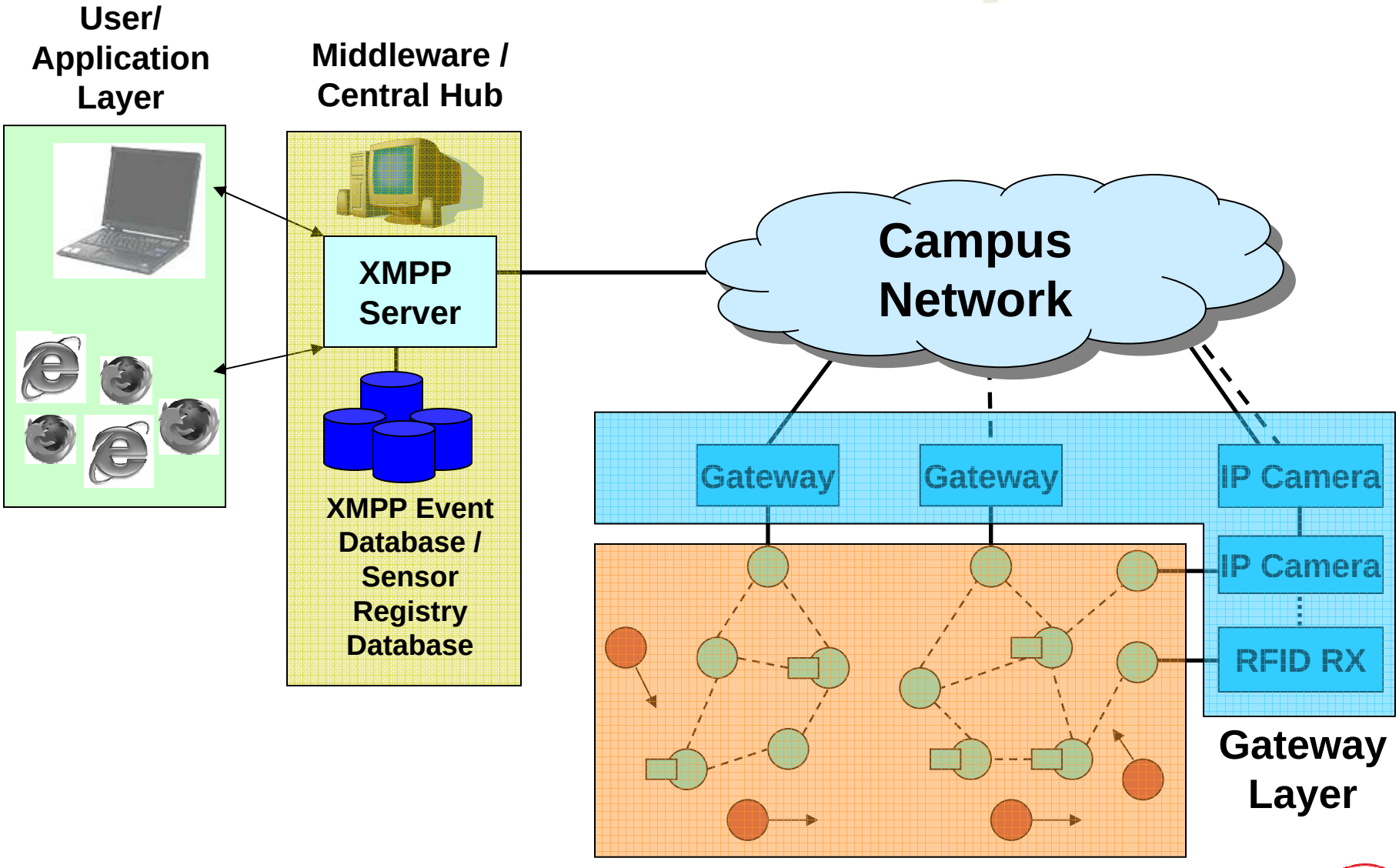
Design Goals

- Large-scale monitoring and control
- Easy to mange, configure and use
- Extensibility
- Security and Privacy
- True Infrastructure Sharing
- Evolvability
- Robustness

Communication Goals

- Standard Messaging Format
- Extensible Message Types
- Point-to-point and multicast messaging
- Data Tracking and/or Event Logging
- Encryption, Authentication, Access Control
- Internet Scale Performance

Sensor Andrew Layers



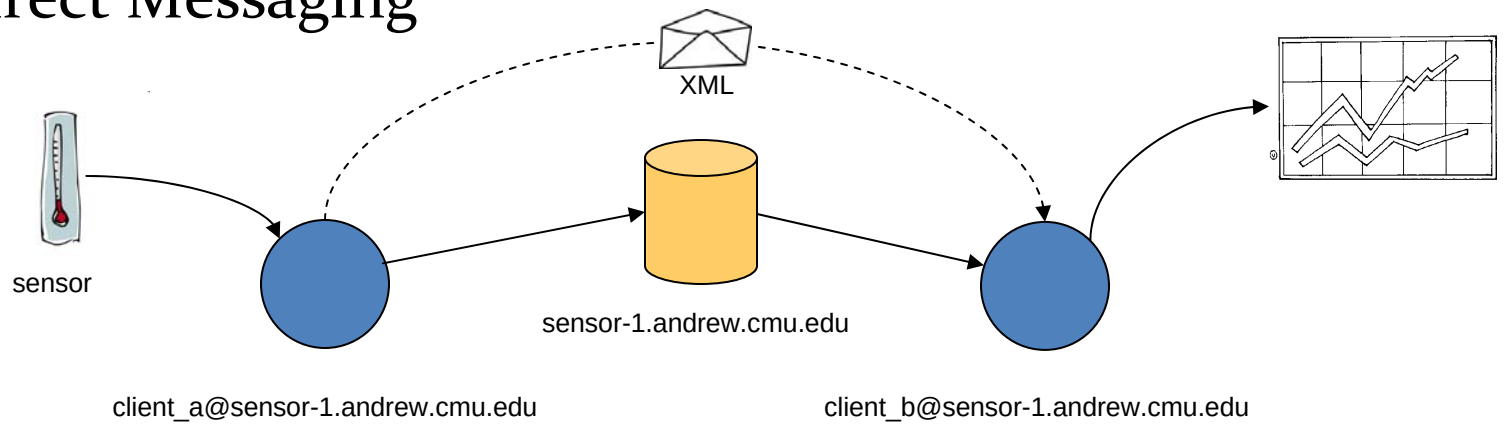
Carnegie Mellon

ices

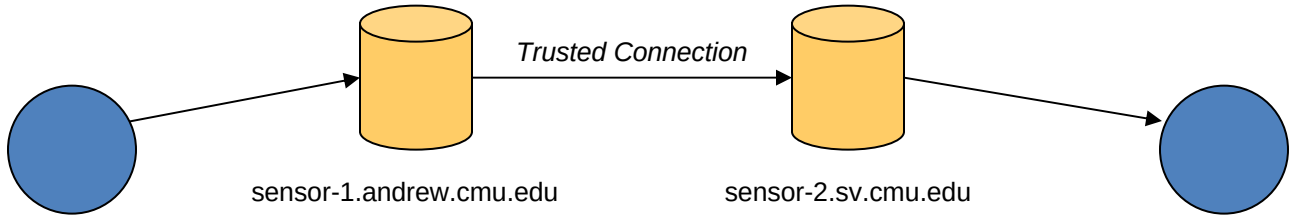
CenSCIR

XMPP Client Messaging

- Direct Messaging



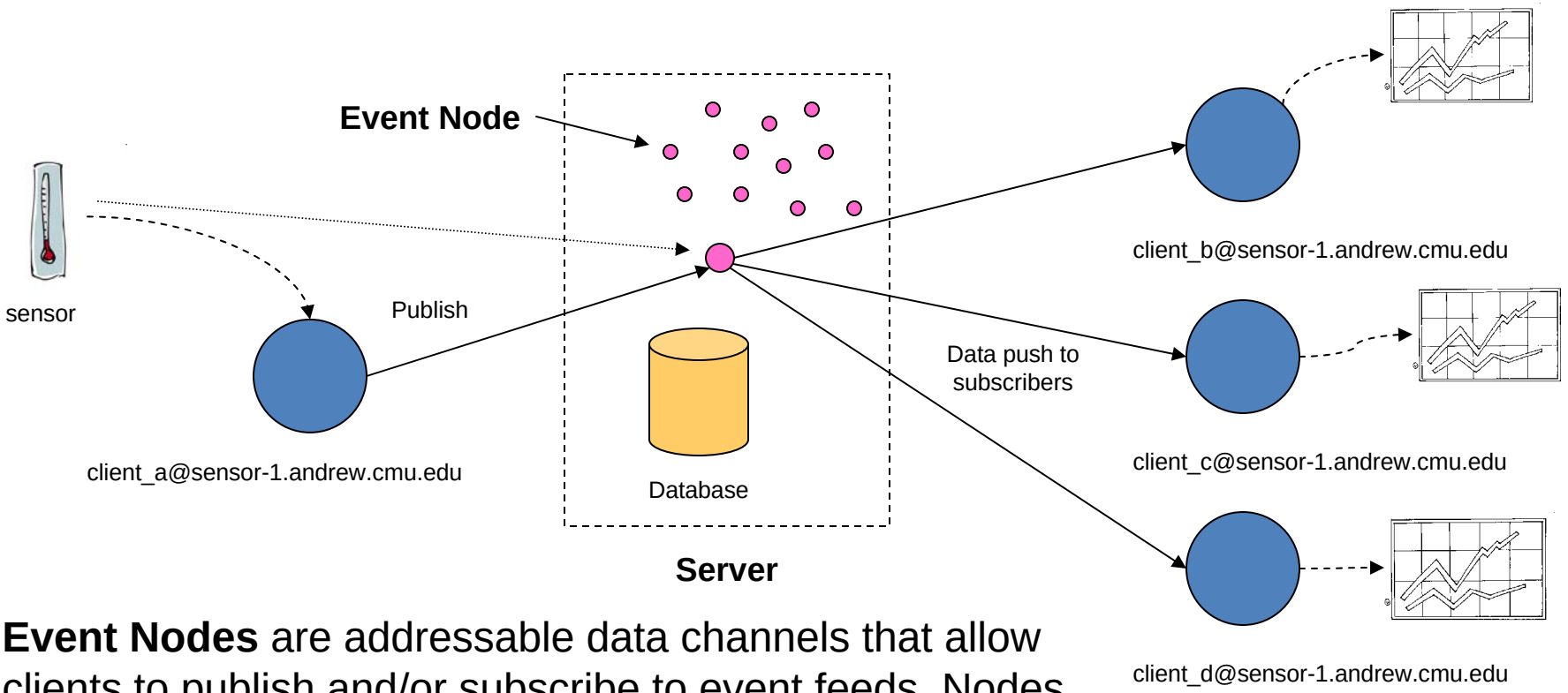
- Server-to-Server Messaging (beyond just local campus)



Carnegie Mellon

ICES CenSCIR

XMPP Publish / Subscribe

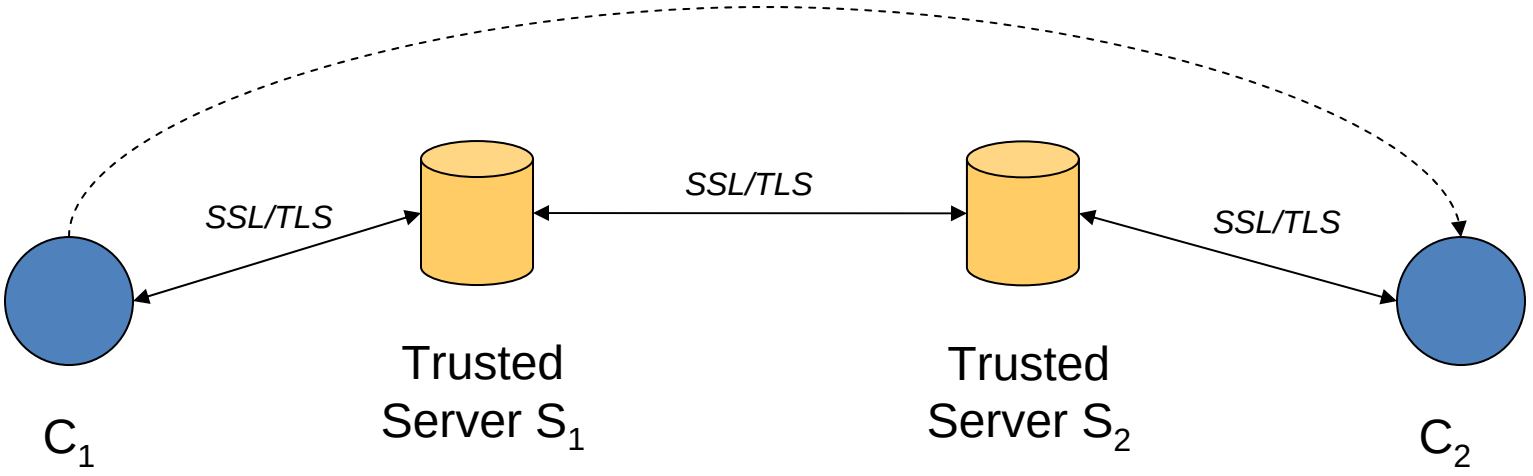


Event Nodes are addressable data channels that allow clients to publish and/or subscribe to event feeds. Nodes may also maintain a history of events and provide other services that supplement the pure pubsub model.

Carnegie Mellon

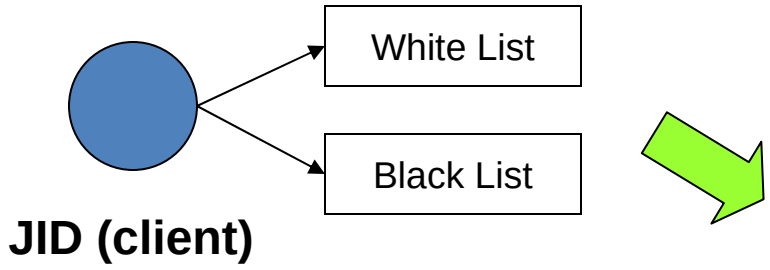
ICES CenSCIR

XMPP Security



- C_1 wants to send a message to C_2…
- C_1 establishes a session with S_1
- Likewise C_2 sets up a session with S_2
- S_1 and S_2 trust each other
- S_2 passes message to C_2 marking in the source field that S_1 vouches it came from C_1

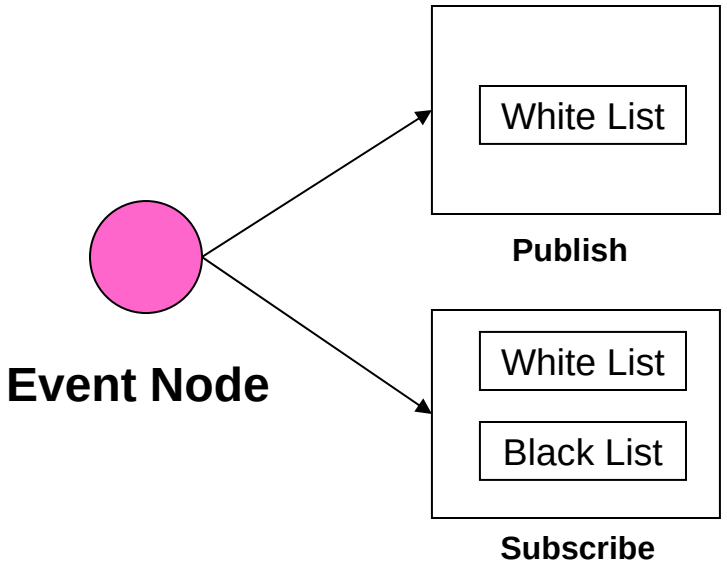
XMPP Access Control



Full JID allows access control across trusted domains.

client_a@sensor.andrew.cmu.edu
client_b@gmail.com
Client_c@jabber.andrew.cmu.edu
roster_a @ sensor.andrew.cmu.edu

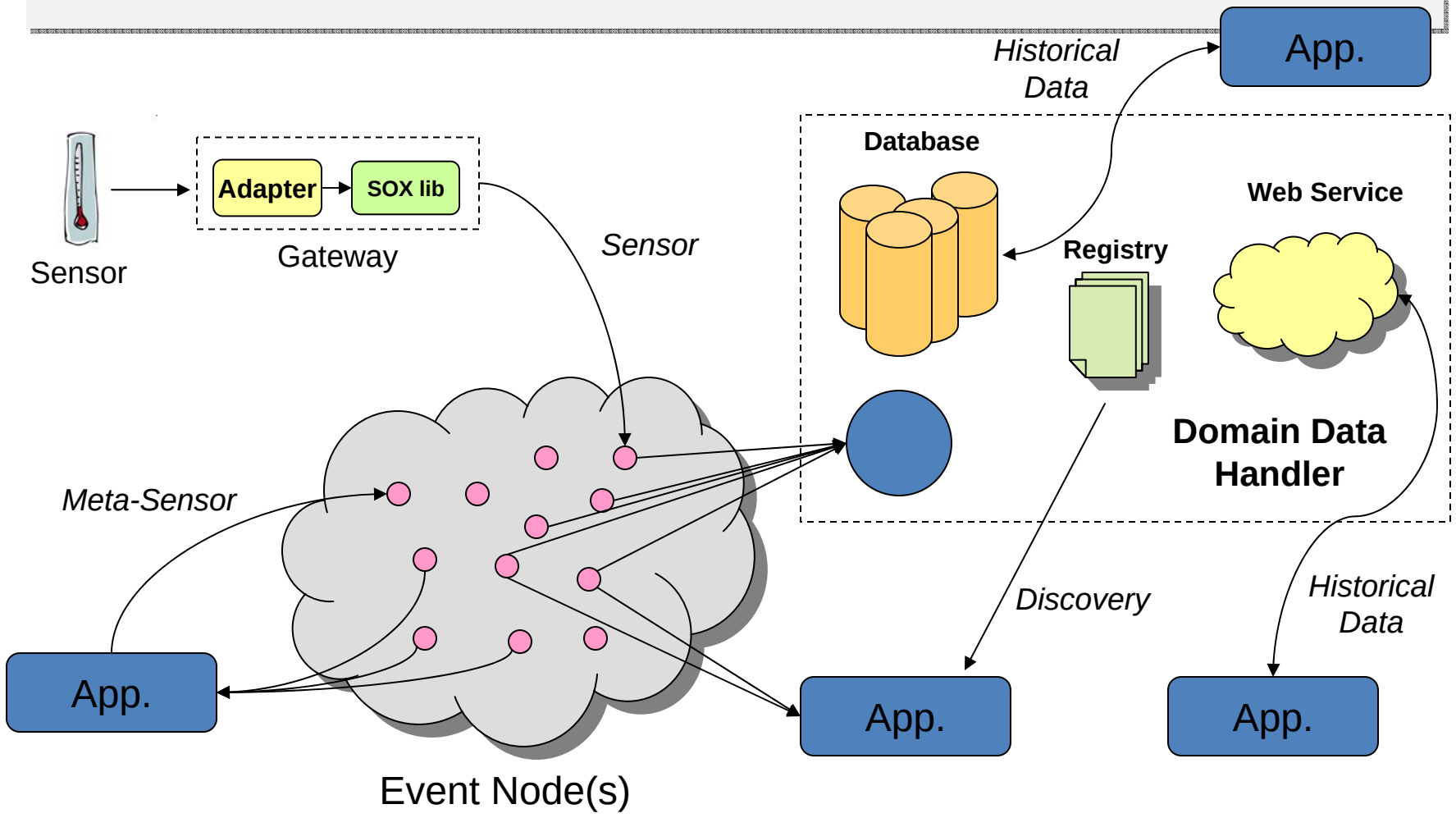
Rosters are groups of JIDS that can be used in access control lists.



Carnegie Mellon

ICES CenSCIR

Sensing



Carnegie Mellon

ICES CenSCIR

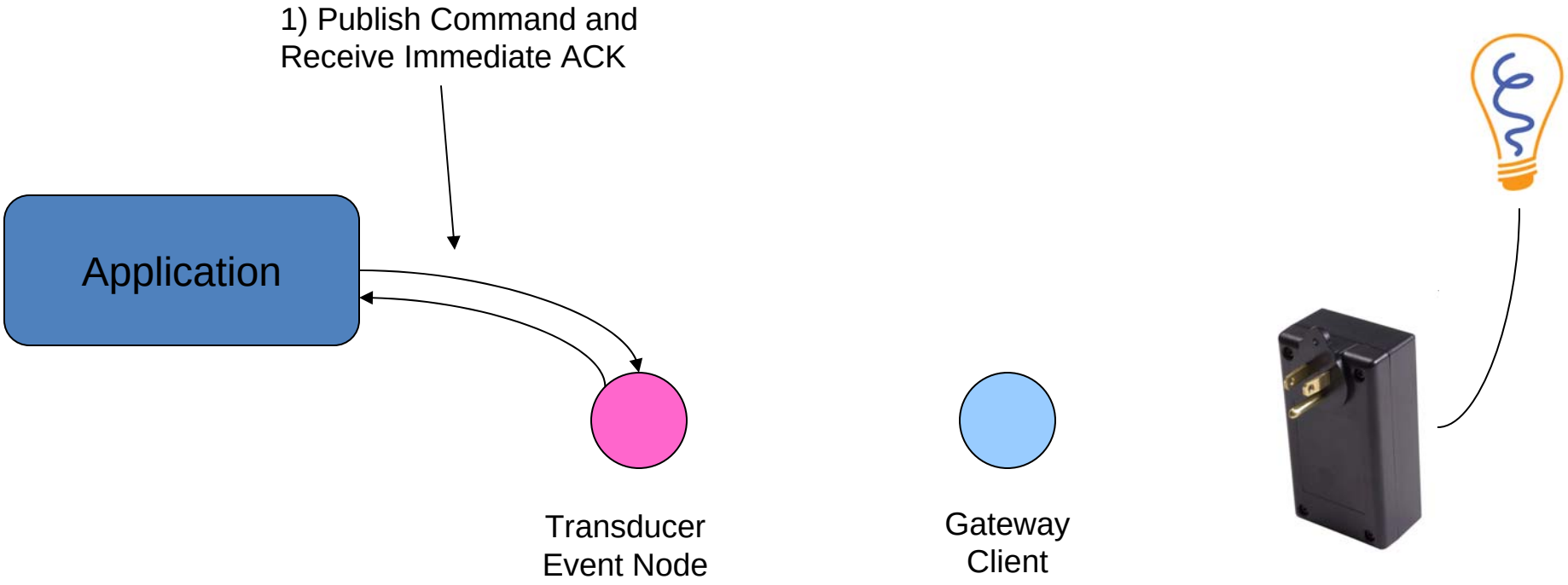
Actuation

Application

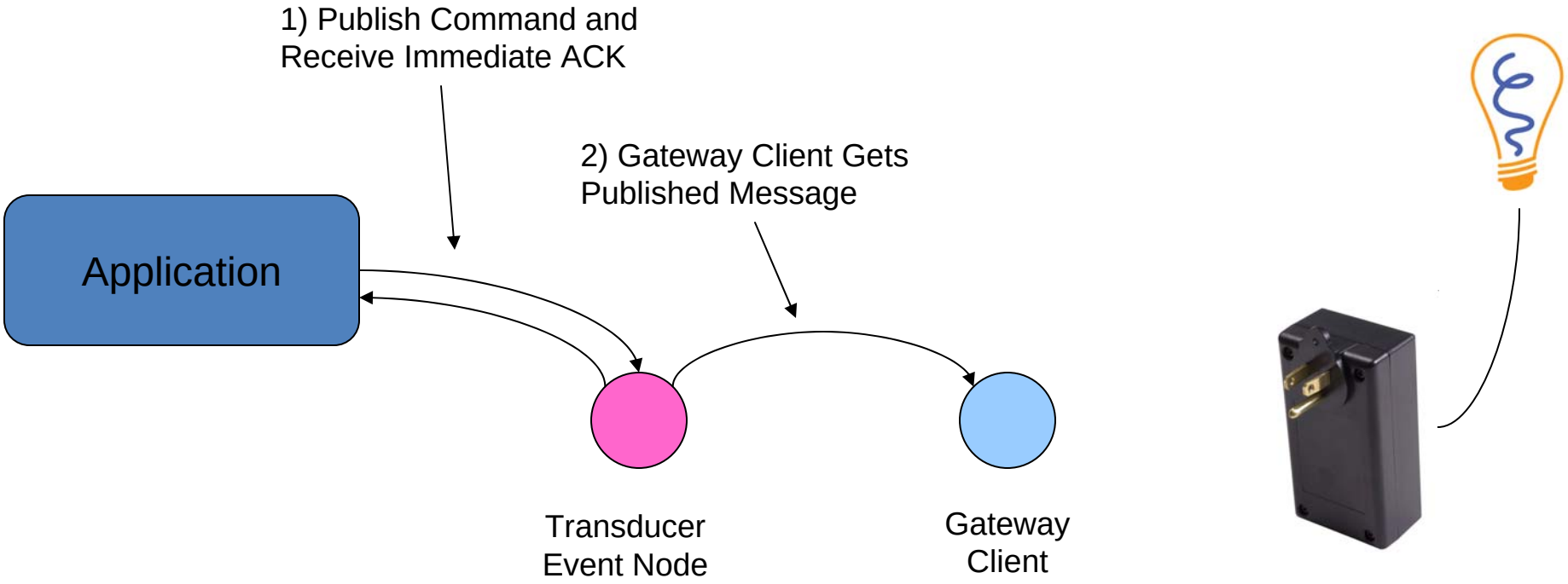
Transducer
Event Node

Gateway
Client

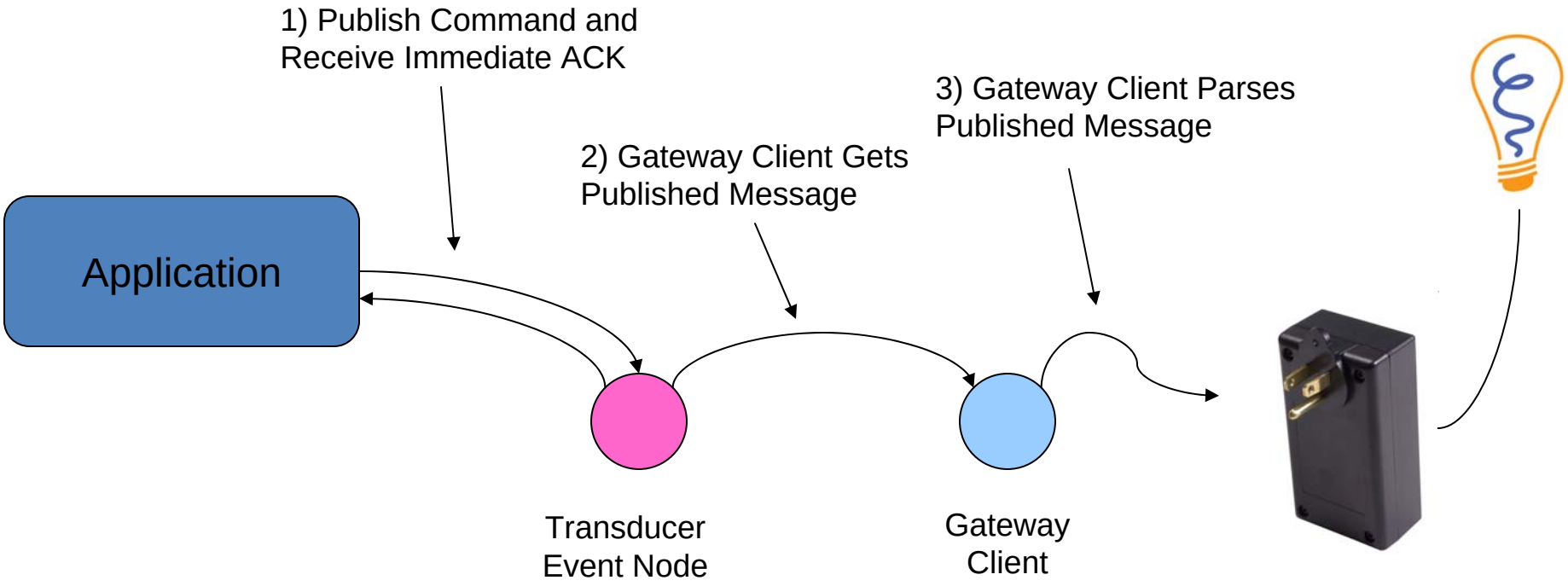
Actuation


1) Publish Command and
Receive Immediate ACK
Application
Transducer
Event Node
Gateway
Client

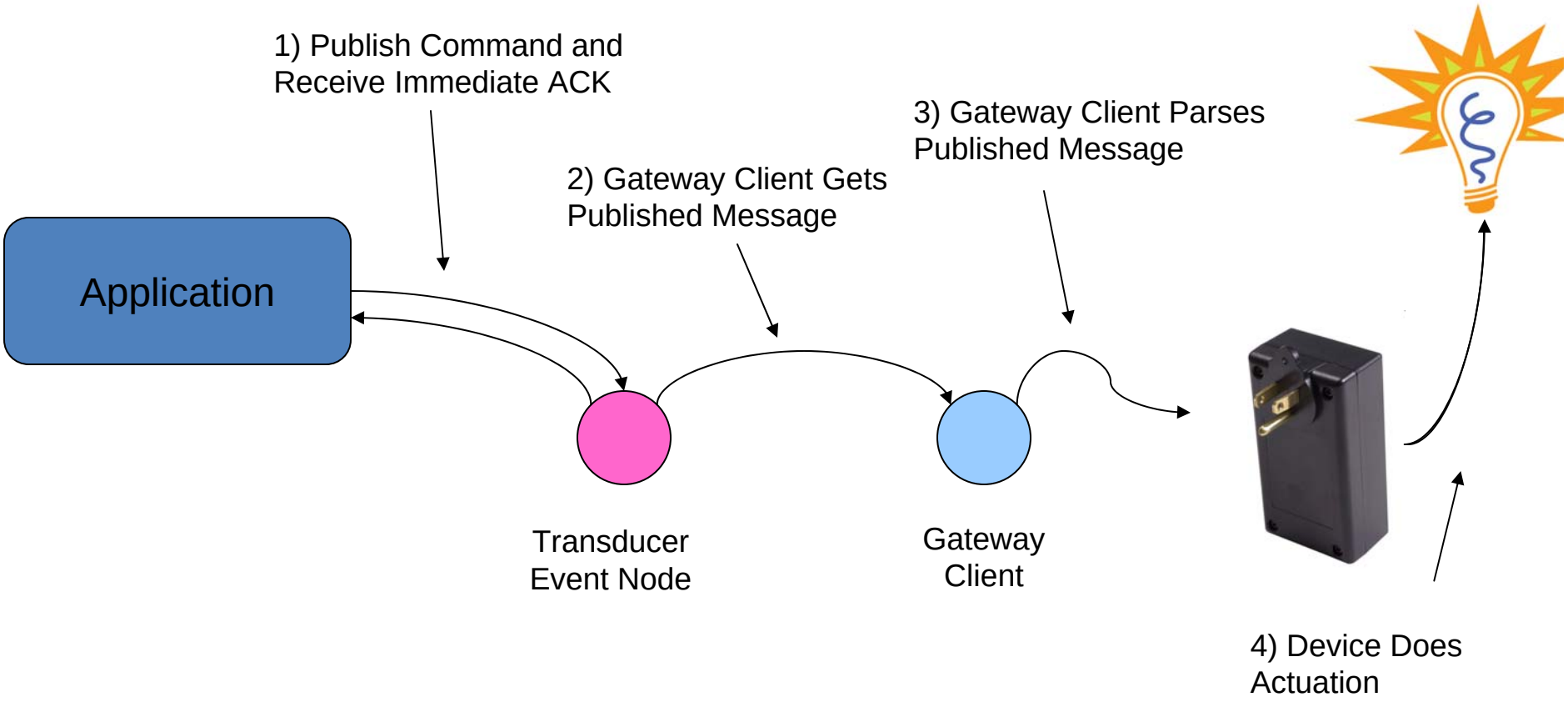
Actuation


1) Publish Command and
Receive Immediate ACK
2) Gateway Client Gets
Published Message
Application
Transducer
Event Node
Gateway
Client

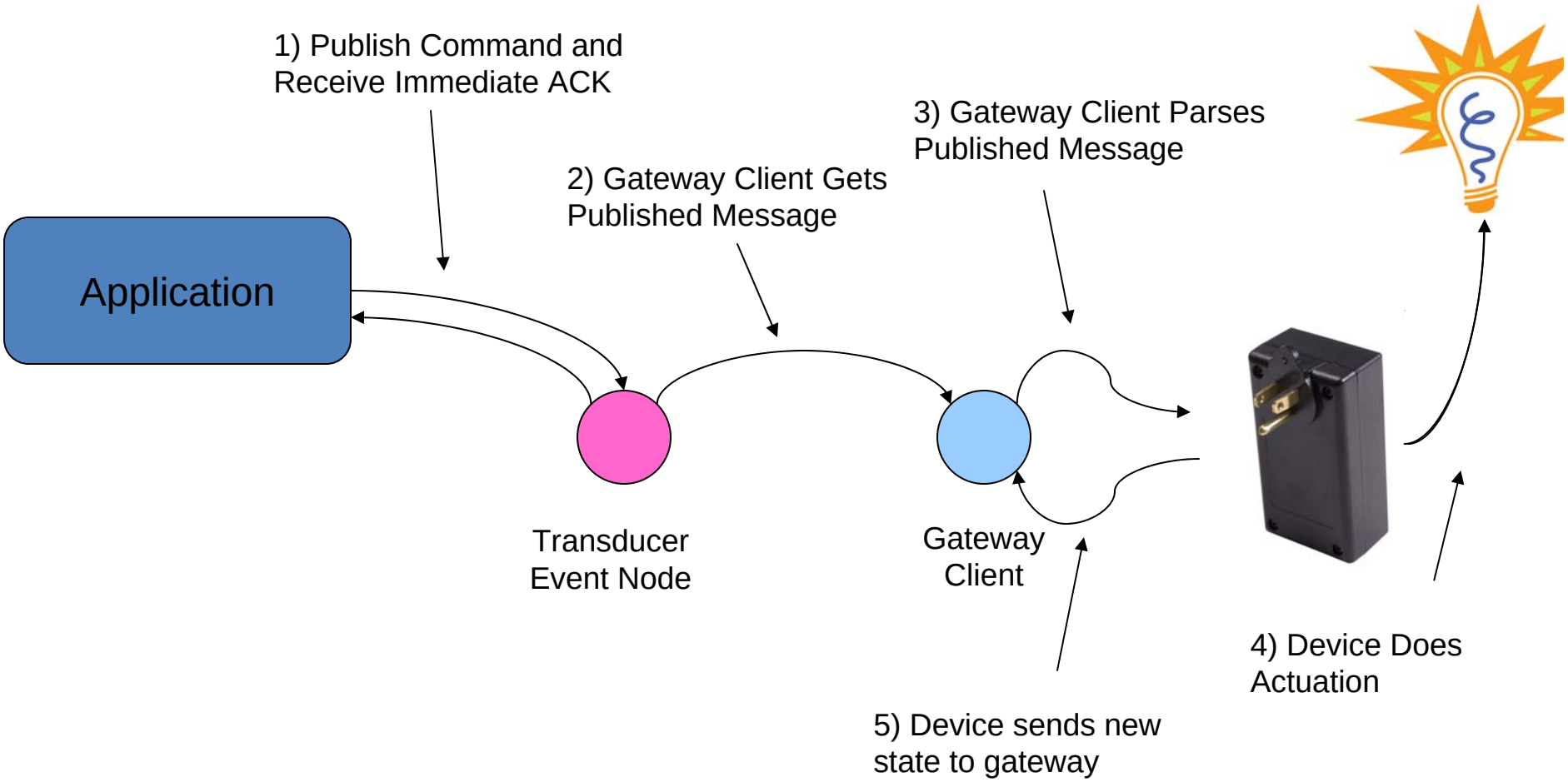
Actuation


1) Publish Command and
Receive Immediate ACK
2) Gateway Client Gets
Published Message
3) Gateway Client Parses
Published Message
Application
Transducer
Event Node
Gateway
Client

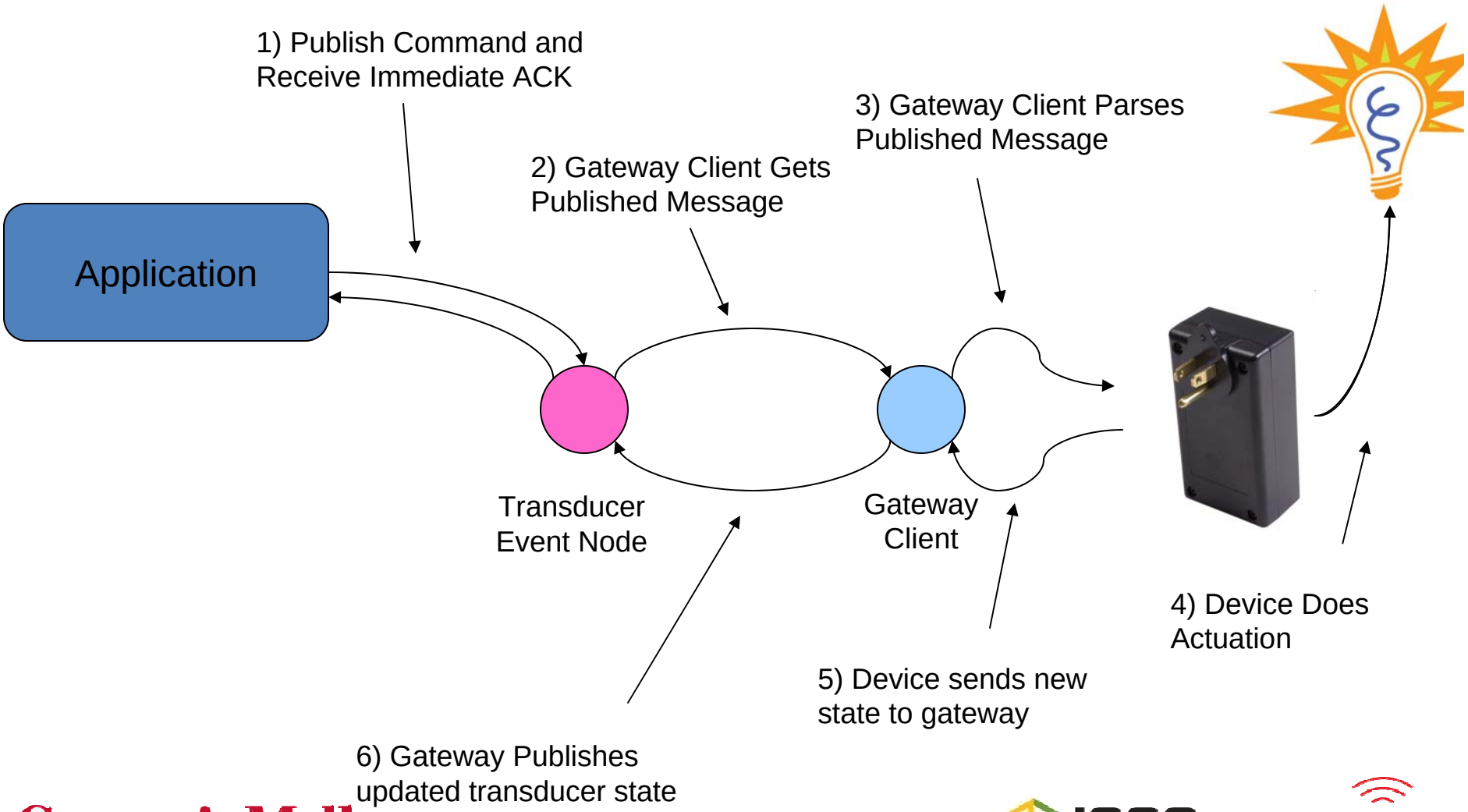
Actuation


1) Publish Command and
Receive Immediate ACK
2) Gateway Client Gets
Published Message
3) Gateway Client Parses
Published Message
Application
Transducer
Event Node
Gateway
Client
4) Device Does
Actuation

Actuation


1) Publish Command and
Receive Immediate ACK
2) Gateway Client Gets
Published Message
3) Gateway Client Parses
Published Message
Application
Transducer
Event Node
Gateway
Client
4) Device Does
Actuation
5) Device sends new
state to gateway

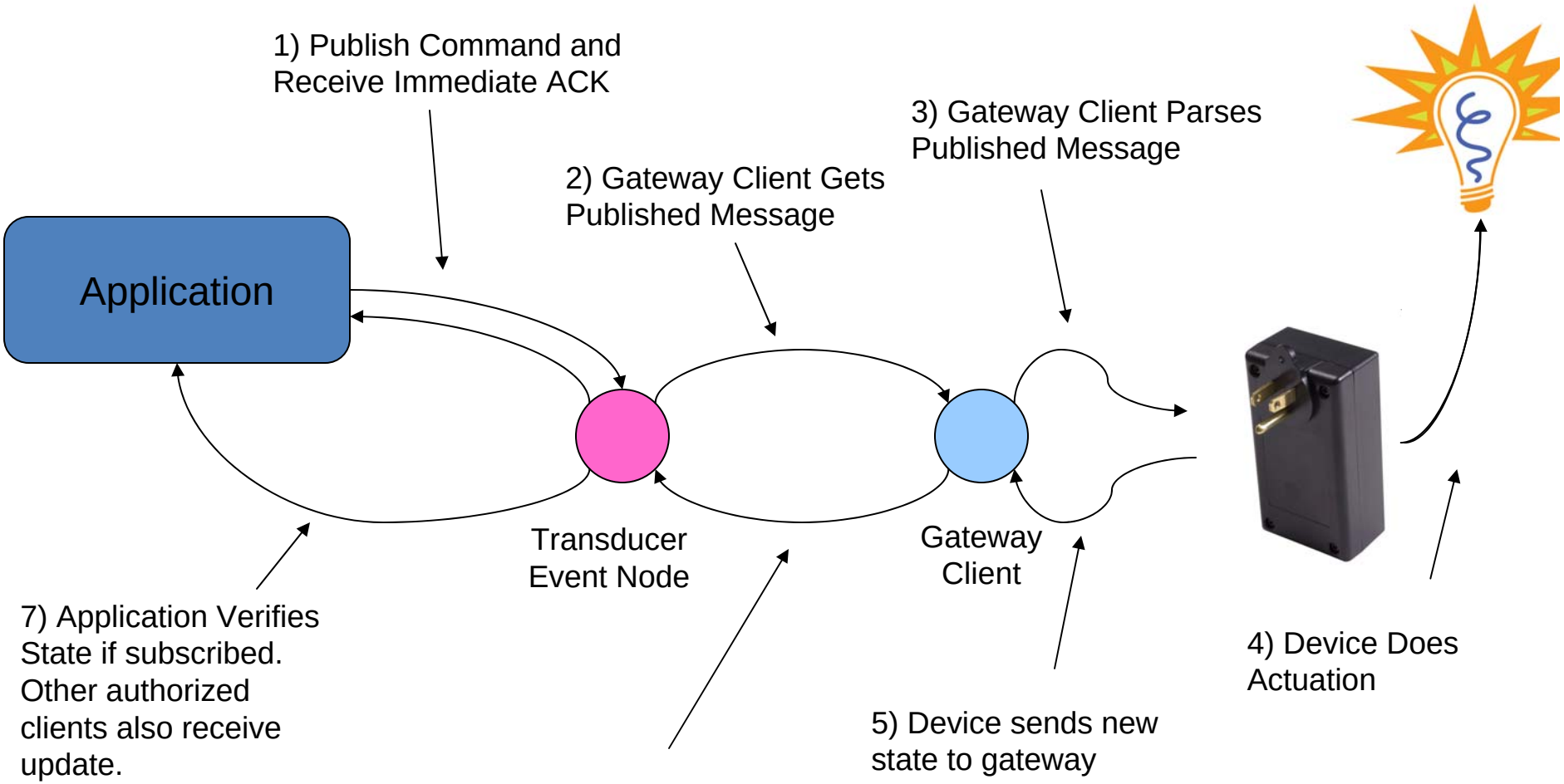
Actuation



Carnegie Mellon

ICES CenSCIR

Actuation



Carnegie Mellon

ICES CenSCIR

Sensor Over XMPP

- SOX Library
  - Command Line Tools w/ Examples
- Implementaion in various languages:
  - C
  - .NET
  - LabVIEW
  - Java
  - Python
- Implementation/testing in various platforms:
  - MAC
  - Linux
  - Gumstix
- Common transport XML schema

Example: Adding a new sensor

Example: Adding a new sensor

SOX

```
/* Prepare the SOX message */
msg = create_sox_message();
msg_add_device_installation(msg,
                            (gchar *) "TED",
                            (gchar *) ted.device_reg_id,
                            (gchar *) "watts",
                            (gchar *) "Obtained with TED Adapter",
                            (gchar *) time_str);

msg_add_transducer_installation(msg,
                                (gchar *) "TED",
                                (gchar *) "Power",
                                (gchar *) "TED-Power",
                                (gchar *) ted.power_transducer_reg_id,
                                FALSE);

sprintf( (char *) watt_value, "%d", watts);
msg_add_value_to_transducer(msg,
                            (gchar *) "TED",
                            (gchar *) "TED-Power",
                            watt_value,
                            watt_value,
                            (gchar *) time_str);

msg_add_transducer_installation(msg,
                                (gchar *) "TED",
                                (gchar *) "Voltage",
                                (gchar *) "TED-Voltage",
                                (gchar *) ted.voltage_transducer_reg_id,
                                FALSE);

sprintf( (char *) vrms_value, "%d", vrms);
msg_add_value_to_transducer(msg,
                            (gchar *) "TED",
                            (gchar *) "TED-Voltage",
                            vrms_value,
                            vrms_value,
                            (gchar *) time_str);

/* Write the SAGA structure to the database */
if(ted.saga_support && counter % (ted.storing_interval) == 0)
  write_power(saga_power);

/* Publish SOX message */
            if (publish_sox_message(connection, ted.event_node_id, msg)) {
                    error_msg("Failed to publish XMPP message.");
                    break;
            }
delete_sox_message(msg);
```

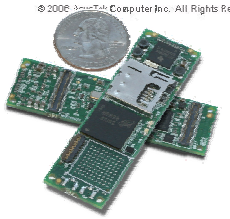
Carnegie Mellon

ICES CenSCIR

Example: Adding a new sensor

SOX

```
/* Prepare the SOX message */
msg = create_sox_message();
msg_add_device_installation(msg,
                            (gchar *) "TED",
                            (gchar *) ted.device_reg_id,
                            (gchar *) "watts",
                            (gchar *) "Obtained with TED Adapter",
                            (gchar *) time_str);

msg_add_transducer_installation(msg,
                                (gchar *) "TED",
                                (gchar *) "Power",
                                (gchar *) "TED-Power",
                                (gchar *) ted.power_transducer_reg_id,
                                FALSE);

sprintf( (char *) watt_value, "%d", watts);
msg_add_value_to_transducer(msg,
                            (gchar *) "TED",
                            (gchar *) "TED-Power",
                            watt_value,
                            watt_value,
                            (gchar *) time_str);

msg_add_transducer_installation(msg,
                                (gchar *) "TED",
                                (gchar *) "Voltage",
                                (gchar *) "TED-Voltage",
                                (gchar *) ted.voltage_transducer_reg_id,
                                FALSE);

sprintf( (char *) vrms_value, "%d", vrms);
msg_add_value_to_transducer(msg,
                            (gchar *) "TED",
                            (gchar *) "TED-Voltage",
                            vrms_value,
                            vrms_value,
                            (gchar *) time_str);

/* Write the SAGA structure to the database */
if(ted.saga_support && counter % (ted.storing_interval) == 0)
  write_power(saga_power);

/* Publish SOX message */
          if (publish_sox_message(connection, ted.event_node_id, msg)) {
                  error_msg("Failed to publish XMPP message.");
                  break;
          }
delete_sox_message(msg);
```



Carnegie Mellon

ICES CenSCIR

Example: Adding a new sensor

XMPP Server: sensor.andrew.cmu.edu
XMPP Server Port: 5223
XMPP PubSub Server: pubsub.sensor.andrew.cmu.edu
XMPP Server SSL Fingerprint: blah
username: marioberges@sensor.andrew.cmu.edu
Event Node: 00000803
Verbose: YES

initialized xmpp client
Data from node '00000803'
<item id="0mp9t3WDyxWwW2k"> <System>
<DeviceInstallation id="00000803" regid="105" type="JigaWatt" description="A Firefly Node with Power" timestamp="2009-12-01T00:01:17">
<TransducerInstallation name="Voltage1" id="0001" regid="360" canActuate="false">
<TransducerValue typedValue="120.952377" rawValue="381" timestamp="2009-12-01T00:01:17"></TransducerValue>
</TransducerInstallation>
<TransducerInstallation name="Current1" id="0002" regid="361" canActuate="false">
<TransducerValue typedValue="1.888889" rawValue="34" timestamp="2009-12-01T00:01:17"></TransducerValue>
</TransducerInstallation>
<TransducerInstallation name="TruePower1" id="0003" regid="362" canActuate="false">
<TransducerValue typedValue="169.132736" rawValue="9556" timestamp="2009-12-01T00:01:17"></TransducerValue>
</TransducerInstallation>
<TransducerInstallation name="ApparentPower1" id="0004" regid="unknown" canActuate="false">
<TransducerValue typedValue="228.465591" rawValue="228.465591" timestamp="2009-12-01T00:01:17"></TransducerValue>
</TransducerInstallation>
<TransducerInstallation name="PowerFactor1" id="0005" regid="363" canActuate="false">
<TransducerValue typedValue="1.000000" rawValue="1.000000" timestamp="2009-12-01T00:01:17"></TransducerValue>
</TransducerInstallation>
<TransducerInstallation name="Energy1" id="0006" regid="364" canActuate="false">
<TransducerValue typedValue="177946.171875" rawValue="10053959" timestamp="2009-12-01T00:01:17"></TransducerValue>
</TransducerInstallation>
<TransducerInstallation name="State1" id="0007" regid="366" canActuate="false">
<TransducerValue typedValue="1" rawValue="1" timestamp="2009-12-01T00:01:17"></TransducerValue>
</TransducerInstallation>
<TransducerInstallation name="Freq1" id="0008" regid="unknown" canActuate="false">
<TransducerValue typedValue="59" rawValue="59" timestamp="2009-12-01T00:01:17"></TransducerValue>
</TransducerInstallation>
</DeviceInstallation>
</System>
</item>
Got data from node <00000803>

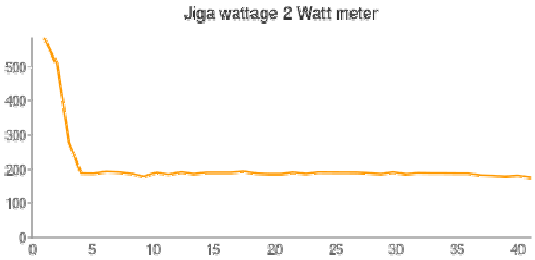
Example: Adding a new sensor

Device

```
/* Prepare the SOX message */
msg = create_sox_message();
msg_add_device_installation(msg,
                            (gchar *) "TED",
                            (gchar *) ted.device_reg_id,
                            (gchar *) "watts",
                            (gchar *) "Obtained with TED Adapter",
                            (gchar *) time_str);

msg_add_transducer_installation(msg,
                                (gchar *) "TED",
                                (gchar *) "Power",
                                (gchar *) "TED-Power",
                                (gchar *) ted.power_transducer_reg_id,
                                FALSE);

sprintf( (char *) watt_value, "%d", watts);
msg_add_value_to_transducer(msg,
                            (gchar *) "TED",
                            (gchar *) "TED-Power",
                            watt_value,
                            watt_value,
                            (gchar *) time_str);

msg_add_transducer_installation(msg,
                                (gchar *) "TED",
                                (gchar *) "Voltage",
                                (gchar *) "TED-Voltage",
                                (gchar *) ted.voltage_transducer_reg_id,
                                FALSE);

sprintf( (char *) vrms_value, "%d", vrms);
msg_add_value_to_transducer(msg,
                            (gchar *) "TED",
                            (gchar *) "TED-Voltage",
                            vrms_value,
                            vrms_value,
                            (gchar *) time_str);

/* Write the SAGA structure to the database */
if(ted.saga_support && counter % (ted.storing_interval) == 0)
  write_power(saga_power);

/* Publish SOX message */
            if (publish_sox_message(connection, ted.event_node_id, msg)) {
                    error_msg("Failed to publish XMPP message.");
                    break;
            }
delete_sox_message(msg);
```

SOX

Gateway



Application

Carnegie Mellon

ICES CenSCIR

Proposed Solution

Problem

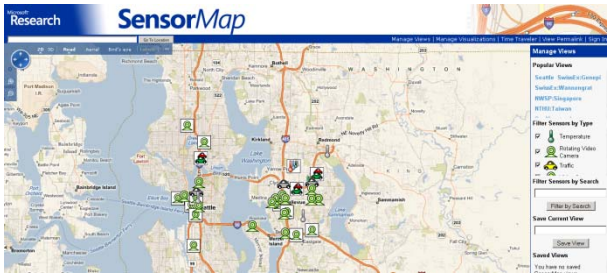
- Communication Requirements
- Meta-Data Operations
- Heterogeneity

Solution

- XMPP
- Data Handler
  - Registry
  - Historical DB
- SOX Library, SOX Adapters

Similar Projects

- **Microsoft:** SenseWeb & SensorMap
- Pachube
- **Intel:** IrisNet
- **OGC:** SensorML & TransducerML
- **Sensorpedia**



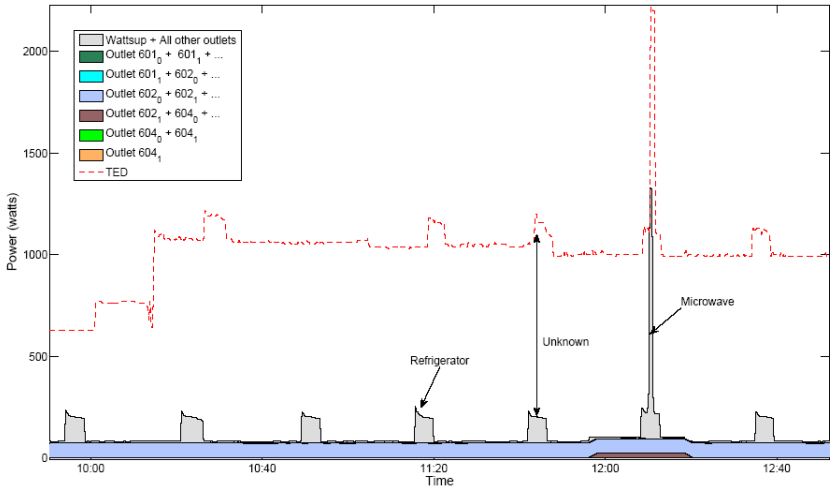
Differences

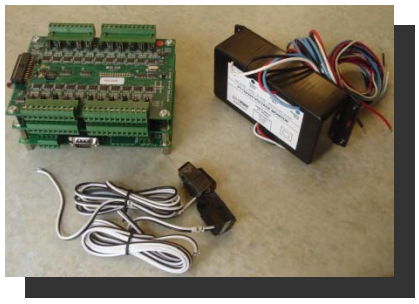
- Fully heterogeneous
- Support for Actuation
- Non-centralized architecture
- Highly scalable and adaptable
  - Not purpose-specific
  - Evolvable
  - Easy to expand and reconfigure

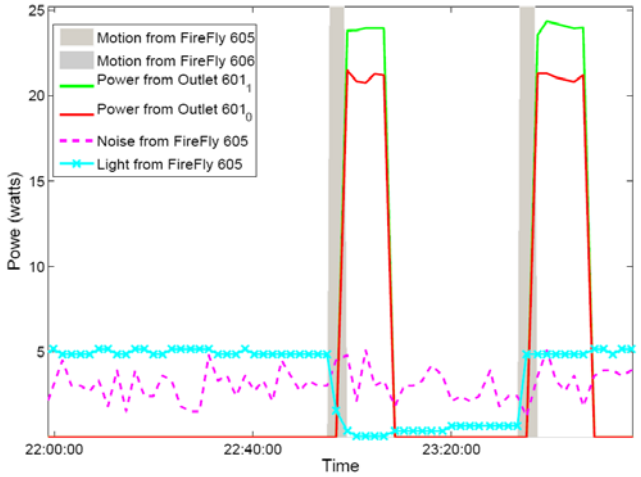
Current Applications

- Building Energy Monitoring
  - For research
  - For Facility Management Operations
- Pipeline Monitoring
- Green Roof Project
- Event-Notification System
- Web-portal

Energy Monitoring







Other Projects

Integrating Building Automation Systems (Intelligent Workplace)

Pipeline Monitoring (Leakage Detection)

(Green Roof)

Other Projects

Demonstration

http://sensor.andrew.cmu.edu/

Future Development

- **User interfaces**
- **Complete the API**
- **Self-Describing Data**
- **Expanding the coverage**
- **Larger user-base**
- **New Applications**

The End

Questions?