Chapter 1: IIoT Fundamentals and Components
Chapter 2: IIoT Application Architecture and Design

- Device
- Gateway
- Cloud
Connect your existing devices which speak their existing protocols without making expensive changes to them.

Create a module pipeline which accomplishes your specific scenario.

Only send necessary data to the cloud for further analytics, dashboarding, or storage.

---

**Cloud Computing Layer**

- Aggregates data summaries from multiple fog nodes
- Performs deeper analysis on larger data set and sends application rules to fog nodes

**Fog Computing Layer**

- Transient Storage for immediate data
- Real-time analytics and control based on application rules provided by cloud layer

**Edge Computing Layer**

- Captures user interactions and sends feeds to Fog node
- Performs Intelligent Actions based on real time control signals from Fog nodes
Chapter 3: IIoT Edge Development

![Diagram of IIoT Edge Development]

```javascript
em:receiver melnikaite$ npm start
> receiver@1.0.0 start /Users/melnikaite/projects/iot-book/http/receiver
> node index.js

```

```json
{
  temperature: '26.00', humidity: '31.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '31.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '31.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '31.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '31.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '32.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '32.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '35.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '35.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '38.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '38.00', device: 'raspberry'
}
{
  temperature: '27.00', humidity: '38.00', device: 'raspberry'
}
{
  temperature: '27.00', humidity: '38.00', device: 'raspberry'
}
{
  temperature: '27.00', humidity: '38.00', device: 'raspberry'
}
{
  temperature: '26.00', humidity: '42.00', device: 'raspberry'
}
```
pi@raspberrypi:~/sensor/dht_12 $ sudo node index.js
{ temperature: '0.00', humidity: '0.00', device: 'raspberry' }
Failed to send to http://172.16.32.193:8080
{ temperature: '0.00', humidity: '0.00', device: 'raspberry' }
Failed to send to http://172.16.32.193:8080
{ temperature: '26.00', humidity: '31.00', device: 'raspberry' }
Failed to send to http://172.16.32.193:8080
{ temperature: '26.00', humidity: '31.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '31.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '31.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '31.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '32.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '32.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '32.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '32.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '35.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '35.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '38.00', device: 'raspberry' }
{ temperature: '26.00', humidity: '38.00', device: 'raspberry' }
{ temperature: '27.00', humidity: '38.00', device: 'raspberry' }

![Diagram of sensor, hub, and receiver with XD-80, Raspberry Pi, and PC labeled.]
pi@raspberrypi:~ $ sudo node device4.1.js
light status: true
Failed to send data to ws://172.16.32.193:8000
light status: true
Failed to send data to ws://172.16.32.193:8000
light status: true
Failed to send data to ws://172.16.32.193:8000
light status: true
light status: false
light status: false
light status: false
light status: true
light status: true
light status: true
light status: true
light status: true
light status: true
light status: true
light status: true
light status: true
light status: true
light status: true
light status: true
light status: true
^C
em:receiver melnikaite$
Sensor → Hub → Receiver

Simulator → Raspberry Pi → PC or Cloud

```
em:hub melnikaite$ npm start

> hub@1.0.0 start /Users/melnikaite/projects/iot-book/modbus/hub
> node index.js

onConnect
321
onDestroy
em:hub melnikaite$
```
Sensor

 Simulator

 Hub

 Raspberry Pi

 Receiver

 PC or Cloud

 sensor — node • npm MANPATH=/Users/melnikaite/.nvm/versions/node/v6.9.4...

 node • npm...ite/.rvm/bin ★ ...us/hub — --bash --login node • npm...aite/.rvm/bin ...

!em: sensor melnikaite$ npm start

 > sensor@1.0.0 start /Users/melnikaite/projects/iot-book/modbus/sensor
 > node index.js

 { nb_bits: 2,
   nb_input_bits: 0,
   nb_input_registers: 0,
   nb_registers: 5,
   tab_bits: [ 0, 1 ],
   tab_input_bits: [],
   tab_input_registers: [],
   tab_registers: [ 0, 0, 321, 0, 0 ] }

 onQuery
 [ false, true ]
 onQuery
 [ false, false ]
 onQuery
 [ false, false ]

 receiver — node • npm MANPATH=/Users/melnikaite/.nvm/versions/node/v6.9....

 node • npm...ite/.rvm/bin ★ ...us/hub — --bash --login node • npm...aite/.rvm/bin ...

!em: receiver melnikaite$ npm start

 > receiver@1.0.0 start /Users/melnikaite/projects/iot-book/modbus/receiver
 > node index.js

 { device: 'sensor1', timestamp: '1495713113784', reg2: '321' }
em:sensor  $ npm start

> sensor@1.0.0 start /Users/username/projects/iot-book/opcua/sensor
> node index.js

Endpoint: opc.tcp://username:4334/UA/resourcePath

em:receiver $ npm start

> receiver@1.0.0 start /Users/username/projects/iot-book/opcua/receiver
> node index.js

{ device: 'sensor1',
  timestamp: '1533900069892',
  Variable1: '100' }
{ device: 'sensor1',
  timestamp: '1533900070389',
  Variable1: '37' }
{ device: 'sensor1',
  timestamp: '1533900070891',
  Variable1: '55' }
{ device: 'sensor1',
  timestamp: '1533900070891',
  Variable1: '76' }
{ device: 'sensor1',
  timestamp: '1533900071094',
  Variable1: '97' }
{ device: 'sensor1',
  timestamp: '1533900072396',
  Variable1: '92' }
{ device: 'sensor1',
  timestamp: '1533900072396',
  Variable1: '85' }
{ device: 'sensor1',
  timestamp: '1533900073401',
  Variable1: '60' }
{ device: 'sensor1',
  timestamp: '1533900073401',
  Variable1: '17' }
em:hub $ npm start
> hub@1.0.0 start /Users/.../projects/iot-book/opcua/hub
> node index.js

Connection successful
Variable1 = 24
subscription id: 398664
Variable1 = 100
Variable1 = 37
Variable1 = 55
Variable1 = 76
Variable1 = 97
Variable1 = 92
Variable1 = 85
Variable1 = 60
Variable1 = 17
Successfully finished
em:hub $
Chapter 4: Data for IIoT

Industrial Big Data: Internet of things is not about things
Data-Modelling Challenges: Multiple Personas
Achieving Scalability and Multitenancy
<table>
<thead>
<tr>
<th>Batch Data Integration</th>
<th>Real-Time data Integration</th>
<th>Big data Integration</th>
<th>Data Virtualization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large data sets</td>
<td>Small amounts of data transfer</td>
<td>Very large data sets</td>
<td>Across data stores and data sets</td>
</tr>
<tr>
<td>Point to point</td>
<td>Point to point</td>
<td>Not Point to point</td>
<td>Not Point to point</td>
</tr>
<tr>
<td>Groups of data</td>
<td>Single business transactions</td>
<td>Processing to the data</td>
<td>No intermediate persistent store</td>
</tr>
<tr>
<td>Asynchronous - sent periodically</td>
<td>Synchronous</td>
<td>Only smaller sets of results get integrated</td>
<td>In memory integration</td>
</tr>
<tr>
<td>Tightly coupled</td>
<td>Tightly coupled</td>
<td>Loosely coupled</td>
<td>Loosely coupled</td>
</tr>
<tr>
<td>Careful orchestration of changes required</td>
<td>Logical design solutions to reduce P2P limitations</td>
<td>Master data/keys in structured and metadata tags in unstructured</td>
<td>Truly cross data platform and store integration in-memory</td>
</tr>
</tbody>
</table>

**Data Integration Types**
<table>
<thead>
<tr>
<th></th>
<th>Hub and Spoke</th>
<th>Request and Reply</th>
<th>Pub and Sub</th>
<th>Two-Phase Commits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used for real-time data integration</td>
<td>Point to point or through hub</td>
<td>Pub and sub information</td>
<td>All effected systems update together or fail</td>
<td></td>
</tr>
<tr>
<td>One interface created for each additional system</td>
<td>Synchronous or asynchronous</td>
<td>Requires orchestration system</td>
<td>Leveraged for transactional systems</td>
<td></td>
</tr>
<tr>
<td>All data transformed into canonical model</td>
<td>Need not have a common model</td>
<td>Requires tracking of requests</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Canonical model definition requires careful design</td>
<td>Get information or acknowledgement back</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Data Integration Types**
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Governance Council, Data Owners</td>
<td>Business Analyst</td>
<td>Data Quality Analyst</td>
<td>Data Steward</td>
<td>Data Steward</td>
<td>Security Steward</td>
<td>Compliance Steward</td>
<td>Operational Steward</td>
</tr>
<tr>
<td>- Data Ownership</td>
<td>- Metadata Based</td>
<td>- Data Sampling</td>
<td>- Data Quality Assessment</td>
<td>- Data Analysis</td>
<td>- Data Standards</td>
<td>- Business Terms</td>
<td>- Data Dictionary</td>
</tr>
<tr>
<td>- Business Owner</td>
<td>- Content Based</td>
<td>- Percentage</td>
<td>- %Good</td>
<td>- %Valid</td>
<td>- Names</td>
<td>- Profitability</td>
<td>- Sources</td>
</tr>
<tr>
<td>- External Owners</td>
<td>- Every nth Record</td>
<td>- Data Validity</td>
<td>- Data Types</td>
<td>- Match</td>
<td>- Address</td>
<td>- Cost of Goods Sold</td>
<td>- Targets</td>
</tr>
<tr>
<td>- Data Usage</td>
<td>- References</td>
<td>- Ranges</td>
<td>- Fuzzy</td>
<td>- Exact</td>
<td>- Formats</td>
<td>- Business Models</td>
<td>- Repositories</td>
</tr>
<tr>
<td>- Restrictions</td>
<td>- Deterministic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Data Retention</td>
<td>- Merge</td>
<td>- Business Processes</td>
<td>- Data Models</td>
<td>- Data Analysis</td>
<td>- Business Terms</td>
<td>- Business Models</td>
<td>- Architecture</td>
</tr>
<tr>
<td>- Duration</td>
<td>- Survive</td>
<td>- Order to Cash</td>
<td>- Data Classification</td>
<td>- Data Standard</td>
<td>- Profitability</td>
<td>- Reporting</td>
<td>- Conceptual</td>
</tr>
<tr>
<td>- Regulations</td>
<td>- Best Match</td>
<td>- Procure to Pay</td>
<td>- Public</td>
<td>- Names</td>
<td>- Cost of Goods Sold</td>
<td>- Analytics</td>
<td>- Logical</td>
</tr>
<tr>
<td>- Data Archival</td>
<td>- Cleansed Data</td>
<td></td>
<td>- Internal</td>
<td>- Data Analysis and</td>
<td>- Forecasting</td>
<td></td>
<td>- Physical</td>
</tr>
<tr>
<td>- Offline/Online</td>
<td></td>
<td></td>
<td>- Sensitive</td>
<td>Table</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Data Backup and Restoration</td>
<td></td>
<td></td>
<td>- Restricted</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Business Continuity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Point in Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Data Governance - CTQs and Priorities**
Chapter 5: Advanced Analytics for the IIoT
ML Analytic Lifecycle

In a typical analytic lifecycle, the first step is to develop an analytic by training an model with the datasets and generating the model.

Develop
(Train, Test, Validate)

Publish the Model

Continuous monitoring of the performance for accuracy and other metrics. Based on the outcome retrain the model for better accuracy.

Monitor and Re-Train

Redeploy the new Model

Configure

Configure the analytic for different parameters and data sources.

Deploy

Run the analytic with model settings at scale in a computing environment such as spark for predictions, classification or for data analysis.

Deploy the model with the configurations
Chapter 6: Developing Your First Application for IIoT
$ npm start

> timeseries@1.0.0 start /Users/<username>/projects/iot-book/timeseries
> node index.js

26
82
62
9
18
17
91
95
24

{ attachments: [],
  alternative: null,
  header:
    { 'message-id': '<1499771164791.0.33523@em.local>',
      date: 'Tue, 11 Jul 2017 14:06:04 +0300',
      from: '<username@domain.com>',
      to: '<username@domain.com>',
      content: 'text/plain; charset=utf-8',
      text: 'variable1 exceeded 91 rpm' }

{ attachments: [],
  alternative: null,
  header:
    { 'message-id': '<1499771165813.1.33523@em.local>',
      date: 'Tue, 11 Jul 2017 14:06:05 +0300',
      from: '<username@domain.com>',
      to: '<username@domain.com>',
      content: 'text/plain; charset=utf-8',
      text: 'variable1 exceeded 95 rpm' }
sensor1

![Graph showing sensor data from 22. Jul to 25. Jul](image)

```json
[{
  "time": "2017-08-17T09:08:11.628Z",
  "count_variable1": 32,
  "mean_variable1": 44.40625,
  "median_variable1": 35.5,
  "mode_variable1": 67,
  "spread_variable1": 92,
  "stddev_variable1": 28.706365906334906,
  "sum_variable1": 1421,
  "first_variable1": 35,
  "last_variable1": 85,
  "max_variable1": 95,
  "min_variable1": 3,
  "percentile_variable1": 5
}]
```

![Histogram showing count and mean values](image)
## Panel Title

<table>
<thead>
<tr>
<th>Time</th>
<th>sensor1.mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-08-17 13:29:00</td>
<td>51.93</td>
</tr>
<tr>
<td>2017-08-17 13:28:00</td>
<td>54.27</td>
</tr>
<tr>
<td>2017-08-17 13:27:00</td>
<td>44.12</td>
</tr>
<tr>
<td>2017-08-17 13:26:00</td>
<td>51.32</td>
</tr>
<tr>
<td>2017-08-17 13:25:00</td>
<td>47.97</td>
</tr>
<tr>
<td>2017-08-17 13:24:00</td>
<td>48.43</td>
</tr>
</tbody>
</table>

### Panel Title

- **Panel Title alert**
  - **OK** for 6 minutes ago

### Panel Title

![Graph](image)
<table>
<thead>
<tr>
<th>Time</th>
<th>Subject</th>
<th>Body</th>
</tr>
</thead>
<tbody>
<tr>
<td>11:31</td>
<td>Alert</td>
<td>variable1 exceeded 92 rpm</td>
</tr>
<tr>
<td>11:19</td>
<td>Alert</td>
<td>variable1 exceeded 96 rpm</td>
</tr>
<tr>
<td>11:19</td>
<td>Alert</td>
<td>variable1 exceeded 91 rpm</td>
</tr>
<tr>
<td>11:19</td>
<td>Alert</td>
<td>variable1 exceeded 93 rpm</td>
</tr>
<tr>
<td>11:19</td>
<td>Alert</td>
<td>variable1 exceeded 90 rpm</td>
</tr>
</tbody>
</table>
Integration Settings

Post to Channel
Messages that are sent to the incoming webhook will be posted here.

Webhook URL
Send your JSON payloads to this URL.

Descriptive Label
Use this label to provide extra context in your list of integrations (optional).

Customize Name
Choose the username that this integration will post as.

Customize Icon
Change the icon that is used for messages from this integration.

Preview Message
Here's what messages from this integration will look like in Slack.

Save Settings
### Notifications

<table>
<thead>
<tr>
<th>Send to</th>
<th>Email</th>
<th>Slack</th>
</tr>
</thead>
</table>

**Message**

Notification message details...
variable1 exceeded 92 rpm

incoming-webhook APP 11:49 AM
variable exceeded 93 rpm

incoming-webhook APP 12:18 PM

[Alerting] Test notification
Someone is testing the alert notification within grafana.

<table>
<thead>
<tr>
<th>High value</th>
<th>Higher Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>200</td>
</tr>
</tbody>
</table>

Error message
This is only a test

Grafana v4.4.1 | Today at 12:18 PM (30kB)
[Alerting] Panel Title alert

**Metric name**
sensor1.mean

**Value**
51.979
Chapter 7: Deployment, Scale, and Security

- **Trusted Identity**
- **Trusted Boot**
- **Trusted Update**
- **Trusted Firmware**
- **Trusted Operation**

**Device**

**Gateway**

**Cloud**

**Embedded security modules**
- Efficient, fast, ultra-secure, tested
- Any CPU/OS & trust anchor
- Abstracted API for easy dev & code re-use

**Cloud platform for IoT security operations**
- Policy management
- Monitoring
- Security Analytics

**IIoT Application Tiers**

**IIoT Key Security Areas**

**A. Secure Device (Hardware)**
1. Device intelligence
2. Edge processing

**B. Secure Communications**
3. Device initiated connections
4. Messaging control

**C. Secure Cloud**
5. Identification, authentication and encryption

**D. Secure Lifecycle Management**
6. Remote control and updates of devices
```
// dumpObj:
dumpObj: function( spec ) {
    var val = "<undefined>";
    try {
        val = eval("this."+spec).toString();
    }
    catch( exception ) {
    }
    this.dump( spec + "=" + val + "\n" );
},
```

Review the arguments of this "eval" call to make sure they are validated.

7 months ago ▼ L989 ▼

Vulnerability △ Critical ▲ Open ▼ Not assigned ▼ 30min effort Comment

收受cwe, owasp-a3 ▼
### Vulnerability Name

<table>
<thead>
<tr>
<th>Vulnerability Name</th>
<th># Issues</th>
<th># Fixed</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross-Site Request Forgery</td>
<td>2</td>
<td>0/2</td>
<td><img src="image" alt="Medium" /></td>
</tr>
<tr>
<td>Cross-Site Scripting (XSS)</td>
<td>4</td>
<td>0/4</td>
<td><img src="image" alt="Critical" /></td>
</tr>
<tr>
<td>Cross-Site Scripting in HTM...</td>
<td>1</td>
<td>1/1</td>
<td><img src="image" alt="Low" /></td>
</tr>
<tr>
<td>SQL Injection</td>
<td>2</td>
<td>0/2</td>
<td><img src="image" alt="Low" /></td>
</tr>
<tr>
<td>Unencrypted password form</td>
<td>1</td>
<td>1/1</td>
<td><img src="image" alt="Low" /></td>
</tr>
<tr>
<td>XPath Injection</td>
<td>3</td>
<td>0/3</td>
<td><img src="image" alt="Low" /></td>
</tr>
<tr>
<td>ASP.NET DEBUG Method E...</td>
<td>1</td>
<td>1/1</td>
<td><img src="image" alt="Low" /></td>
</tr>
<tr>
<td>Clickjacking</td>
<td>1</td>
<td>1/1</td>
<td><img src="image" alt="Low" /></td>
</tr>
<tr>
<td>Path Traversal</td>
<td>1</td>
<td>1/1</td>
<td><img src="image" alt="Low" /></td>
</tr>
<tr>
<td>Response splitting</td>
<td>1</td>
<td>1/1</td>
<td><img src="image" alt="Low" /></td>
</tr>
<tr>
<td>Directory listing is enabled.</td>
<td>1</td>
<td>1/1</td>
<td><img src="image" alt="Low" /></td>
</tr>
</tbody>
</table>

### Security Risk

<table>
<thead>
<tr>
<th>Security Risk</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Risk Level</td>
<td><img src="image" alt="High" /></td>
<td><img src="image" alt="Medium" /></td>
<td><img src="image" alt="Low" /></td>
<td><img src="image" alt="None" /></td>
</tr>
</tbody>
</table>

### License Risk

<table>
<thead>
<tr>
<th>License Risk</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Risk Level</td>
<td><img src="image" alt="High" /></td>
<td><img src="image" alt="Medium" /></td>
<td><img src="image" alt="Low" /></td>
<td><img src="image" alt="None" /></td>
</tr>
</tbody>
</table>

### Operational Risk

<table>
<thead>
<tr>
<th>Operational Risk</th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Risk Level</td>
<td><img src="image" alt="High" /></td>
<td><img src="image" alt="Medium" /></td>
<td><img src="image" alt="Low" /></td>
<td><img src="image" alt="None" /></td>
</tr>
</tbody>
</table>
Graph Results

Name: Graph Results

Write All Data to a File

Filename: 

Graphs to Display: Data, Average, Median, Deviation, Throughput

No of Samples: 526
Latest Sample: 469
Average: 301
Deviation: 115
Throughput: 892.544802/1680777/minute
Median: 297
Scale Cube

Y-Axis Scalability: Split by Service

Z-Axis Scalability: Segment by Customer

POD 1
POD 2
POD 3
POD 4

X-Axis Scalability: Replicate & LB

<table>
<thead>
<tr>
<th>Tier</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web</td>
<td>Replicate Web Servers &amp; Load Balance</td>
</tr>
<tr>
<td>App</td>
<td>Store Session in browser or separated Object Cache to horizontally scale app tier independent of web tier</td>
</tr>
<tr>
<td>DB</td>
<td>Use Read Replicas for read-only use cases like reporting, search, etc.</td>
</tr>
</tbody>
</table>
Chapter 8: Reliability, Fault Tolerance, and Monitoring IIoT Applications
Cloud IIoT Application

Platform Microservices
Asset Service, Analytics Service and Timeseries service, User Management Service and Others

Reliability by Design and Development
- App Microservices
- App Microservice
- App Microservice

Reliability Provided by Platform Infrastructure

State in Monolithic approach

State in Microservices approach

stateless presentation services
stateless service
stateful services with separate stores
The diagram illustrates a circuit breaker mechanism in a service-oriented architecture. The flow begins with a client sending a request to a circuit breaker.

- **Circuit Closed**: The circuit breaker is closed, allowing the request to pass through to the service.

- **Circuit Open**: If the circuit breaker receives too many failed requests, it opens the circuit, blocking further requests. This is indicated by "Exceeds Thresholds." The counters are incremented during this phase.

- **Circuit Half Open**: After a period of time or after a certain number of successful requests, the circuit breaker transitionally opens, allowing requests to pass through partially. This is indicated by "Few Requests." The counters are reset during this phase.

- **Circuit Closed**: Once the circuit breaker has sufficient evidence that the service is healthy, it closes the circuit, allowing further requests to pass through.

The diagram shows the flow of requests and the decision-making process of the circuit breaker in maintaining service reliability.
The diagram illustrates an IoT (Internet of Things) system with a central IoT Device Manager at the cloud level. The system has four main components: Key Management, Asset Management, and Device Management. These components are interconnected with mutual authentication/TLS for secure communication.

- **Site 1**: IoT Edge Gateway connected to Key Management and Asset Management.
- **Site 2**: IoT Edge Gateway connected to Asset Management and Device Management.
- **Site 3**: IoT Edge Gateway connected to Device Management and Key Management.

Each connection from the sites to the components on the central level is secured with mutual authentication/TLS.
Chapter 9: Implementing IIoT Applications with Predix

```
$ cf bind-service ui asset-service-instance
Binding service asset-service-instance to app ui in org [org] / space dev as [as] ...
OK
TIP: Use 'cf restage ui' to ensure your env variable changes take effect
```

### Client Name

```
PUT ▼
https://predix-assec.run.aws-usw02-prdce.predix.io/locomotives/1

<table>
<thead>
<tr>
<th>Authorization</th>
<th>Headers (3)</th>
<th>Body</th>
<th>Pre-request Script</th>
<th>Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>form-data</td>
<td>x-www-form-urlencoded</td>
<td>raw</td>
<td></td>
<td>JSON (application/json)</td>
</tr>
</tbody>
</table>

```

```
{
  "url": "/locomotives/1",
  "type": "Diesel-electric",
  "model": "ESS44AC",
  "serial_no": "001",
  "emission_tier": "B4",
  "fleet": "/fleets/up-1",
  "manufacturer": "/manufacturers/GE",
  "engine": "/engines/V12-1",
  "installedOn": "01/12/2005",
  "dateIso": "2005-12-01T13:15:31Z",
  "attributes": {
    "lat": 33.914685,
    "lng": -117.253375
  }
}
```
PATCH  
https://predix-asset.run.aws-usw02-pr.ice.predix.io/locmotives/1

<table>
<thead>
<tr>
<th>Authorization</th>
<th>Headers (3)</th>
<th>Body</th>
<th>Pre-request Script</th>
<th>Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
1  
2  
3  
4  
5  
6  
7  
```

```
1: [{
    "op": "replace",
    "path": "/m/LatLng/lon",
    "value": -117.253375}
}
```

DELETE  
https://predix-asset.run.aws-usw02-pr.ice.predix.io/locmotives/1

<table>
<thead>
<tr>
<th>Authorization</th>
<th>Headers (3)</th>
<th>Body</th>
<th>Pre-request Script</th>
<th>Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```

```
```
Application Authorization

https://my-prefix.predix-analytics-ui.run.aws-usw02-pr.ice.predix.io/callback

Eugene has requested permission to access your account. If you do not recognize this application or its URL, you should click deny. The application will not see your password.

You can change your approval of permissions or revoke access for this application at any time from account settings. By approving access, you agree to Eugene’s terms of service and privacy policy.

Deny
Authorize
Analytic Input

```
{"timeseries": [[1501664960967,12,3],[1501664961973,48,3]]}
```

STATUS: COMPLETED

Analytic Output

```
{"result": 60}
```
Analytic Input

```json
{"timeseries": [[1501664960967, 10, 3], [1501664961973, 40, 3]]}
```

Analytic Output

```json
{"result": 50}
```

Analytic Deployment Configuration

- Memory (MB): 128
- Disk Quote (MB): 128
- Number of Instances: 1

Status: COMPLETED
```json
{
    "analyticId": "Zc5f51a3-b97a-4cab-84e4-e88c2fdd01df",
    "status": "COMPLETED",
    "message": "Analytic executed successfully."
}
```
[{
    "title": "Current value",
    "slug": "device-card",
    "attributes": {
        "name": "sensor1:variable1",
        "device": "/device/raspberry"
    },
    "createTimeStamp": "2017-10-26T13:07:23.014Z"
}]

[{
    "id": "HJmzRLJR-",
    "title": "Current value",
    "slug": "device-card",
    "attributes": {
        "name": "sensor1:variable1",
        "device": "/device/raspberry"
    },
    "createTimeStamp": "2017-10-26T13:07:23.014Z"
}]

[[]]
```json
{
    "title": "Deck Title",
    "cardOrder": [],
    "id": "deck-1"
}
```
```
{
  "id": "HJmzRLJR-",
  "tenantId": "b3e07566-273f-4d1e-85ce-a5d4b2632a27",
  "title": "Current value",
  "slug": "device-card",
  "attributes": {
    "name": "sensor1:variable1",
    "device": "/device/raspberry"
  },
  "createTimestamp": "2017-10-26T13:07:23.014Z"
}
```
Chapter 10: Best Practices for IIoT Applications

{  
  "IIOTEcoSystem":"gets a list of IIOT Devices",  
  "Version":"1.0",  
  "CompanyName":"IIOTPlatform",  
  "Devices": [  
    {  
      "deviceId": "10",  
      "name": "device-pump",  
      "asset-type": "pump",  
      "active": "yes"  
    },  
    {  
      "deviceId": "11",  
      "name": "device-liquifier",  
      "asset-type": "liquifier",  
      "active": "yes"  
    }  
  ]  
}
Application (Client) → Auth0 Authorization Server

Your API (Resource Server) → Application Specific Logs (Control flow, Trace, variables, exception)

Application (API etc)

Runtime (JVM, Python, Node.js)

Edge Device & Platform (Kubernetes, CloudFoundary, VM)

Application Specific Logs
Runtime logs (Objects, Memory allocation, threads, DB Connections, Garbage Collection, device heart beat etc)
Platform logs (load balancer, http connections, CPU, Memory)

Unstructured logs needs proper tracing with correlation ID

Continuous logs based on time series
Device

Log info

Service A → Service B → Service C

[Service A][Info] received request from [user id]...

... [Service B][Info] received request to...

... [Service C][Error] failed to process request...
[Service A][correlation-id][Info] received request from [user id]...
...
[Service B][correlation-id][Info] received request to...
...
[Service C][correlation-id][Error] failed to process request...

Log Analysis

<table>
<thead>
<tr>
<th>splunk</th>
<th>vs</th>
<th>logstash</th>
<th>kibana</th>
</tr>
</thead>
</table>

Application Performance Monitoring

<table>
<thead>
<tr>
<th>New Relic.</th>
<th>vs</th>
<th>Prometheus</th>
<th>Grafana</th>
</tr>
</thead>
</table>
Chapter 11: Future Direction of the IIoT

Solution Architecture