Viewing Suggestions

1. These slides are dense in sections and some of the terminology is newly coined.

   The new terminology is covered in the Glossary section at the bottom of each slide:

2. If running in a browser, the focus will shift to the audio clips. It becomes Irksome.

   In that case, download this slides, open in Acrobat. Audio clips will run in browser.

3. Key Takeaways are in Audio Clips with an Orange Border, near top right corner

4. Verbose Sections have their Associated Audio with a Blue Border

5. References and Citations are on the Anchor Page

6. All Suggestion on how to cover this material "better" are appreciated. Thank you.

Francis daCosta, Jan 2024

Next wave of the Internet is Machines-to-Machines Ecosystems ….

Man Oriented Ecosystem

Not (really) Resource Constrained

- Lots more Processor, Memory, Protocol stacks
- Human Oriented Consumption (external)
- Assumed often "Always On"
- Centrally-managed naming (MACID, et al)

Machine Oriented Ecosystem

Often Resource Constrained

- Often Limited or no processor, memory, etc.
- Consumption for local use (Internal)
- Many remote with Intermittent power
- Built by millions of manufacturers worldwide

… many *cannot* *afford* traditional IP protocol overhead …..

IoT Data Characteristics

- Typically Machine to Machine (M2M) status messaging which is:

- **Terse** – not oriented to humans.. Data packets can be 1-5 bytes. The IPV6 Header alone is **40** Bytes.


IPV6 Overhead = 40 bytes, 1 byte payload

Chirp Overhead = 4 bytes, 1 byte payload

- **Repetitive** but..

- *Individual messages not critical:* Decisions are based on large datasets of corroborated evidence

- Meaning comes from **combination** with other data sources: Data is shared out of Symbiosis

- "**Small Data**"- the data can be cryptic: Red/Yellow/Green/Black machine states take 2 bits only.

- Consumption and generation is mostly **Local**. Many legacy systems are running on PLC controllers.

- Usually **unidirectional**. Machine heart beats and terse status updates. OTA downloads infrequent.

- **Self-classified**: new concept, based on Nature

- **Discovery**, based on Nature and the value of independently corroborated ground truth.

1. "Light" Pollen uses all available transports. Only *specific* receivers decode "message"
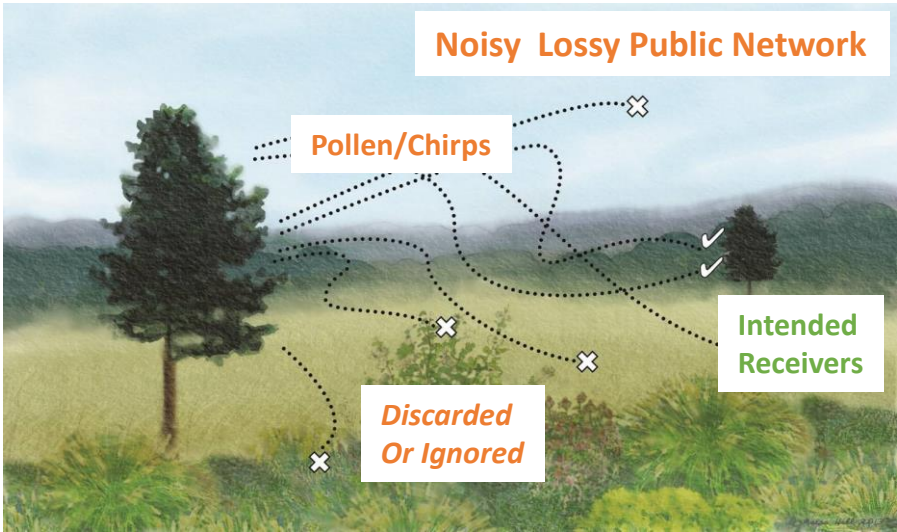
2. All birds derive some information, but only *specific* receivers can fully participate
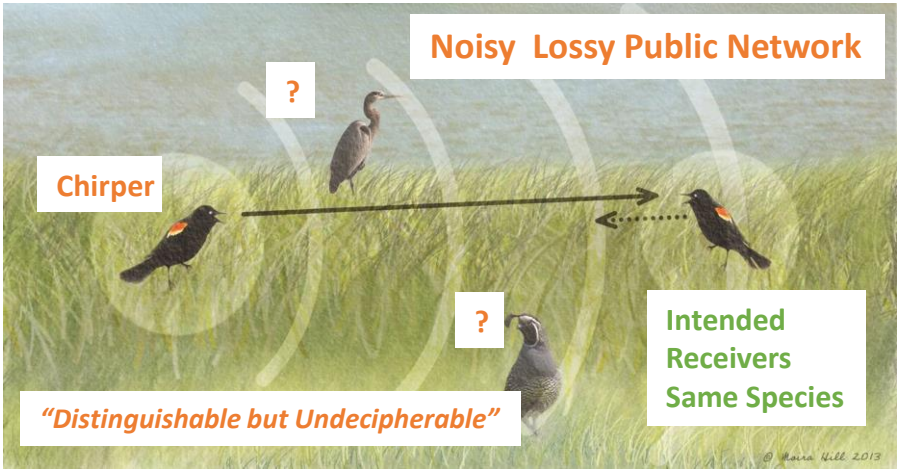
3. Self Classification Underlying event not seen but *affinities* are visible – "Pheromones")

"Light" Chirps   "Small" Data Flows   "Big" Data Analysis
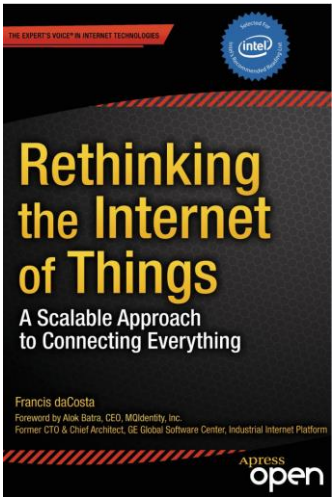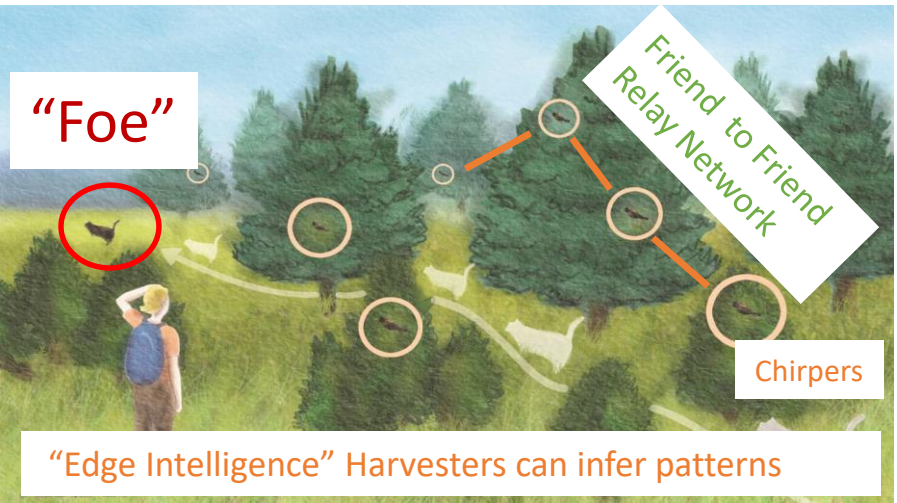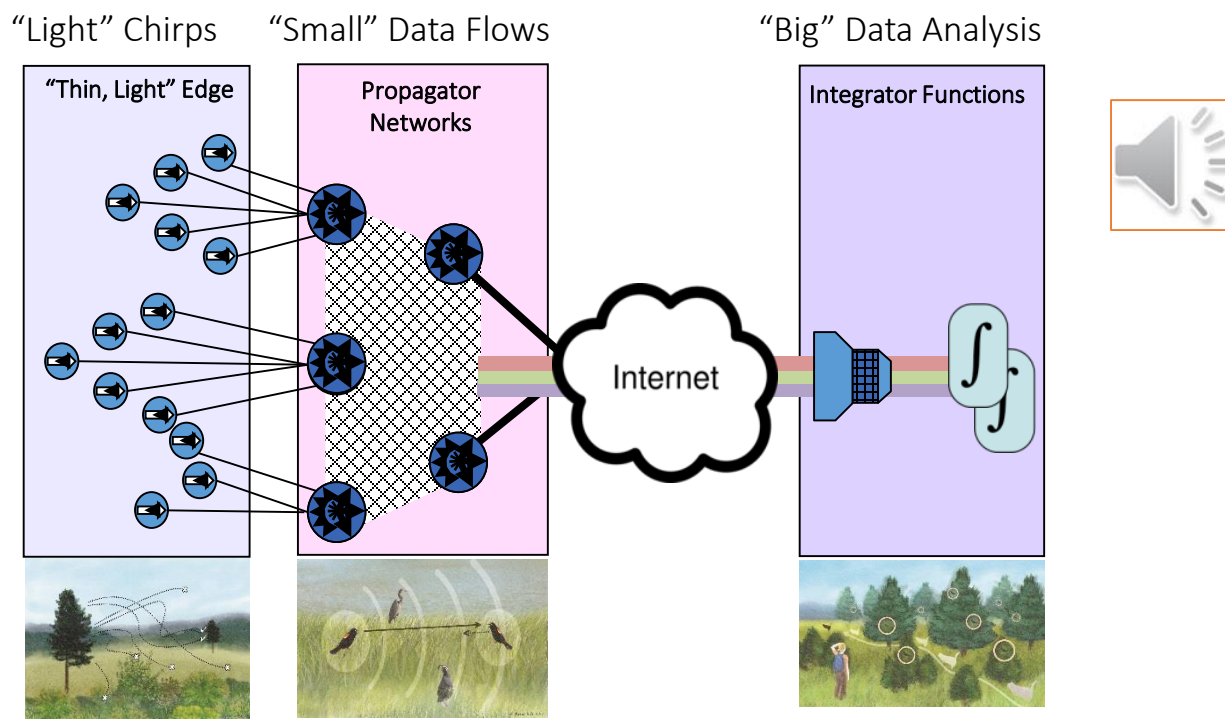
"Light" Chirps.   (Small Dumb Cheap Copious and Secure)

| PHY only (radio, et al) |
| Transmitter or Receiver |
| Modulator |
| Detector |

"Small" Data Flows   (Connected Devices do the Heavy Lifting)

| MAC/PHY (radio, et al) |
| Receiver |
| Transmitter |
| Memory |
| Operating System |
| Processor |
| I/O Logic |
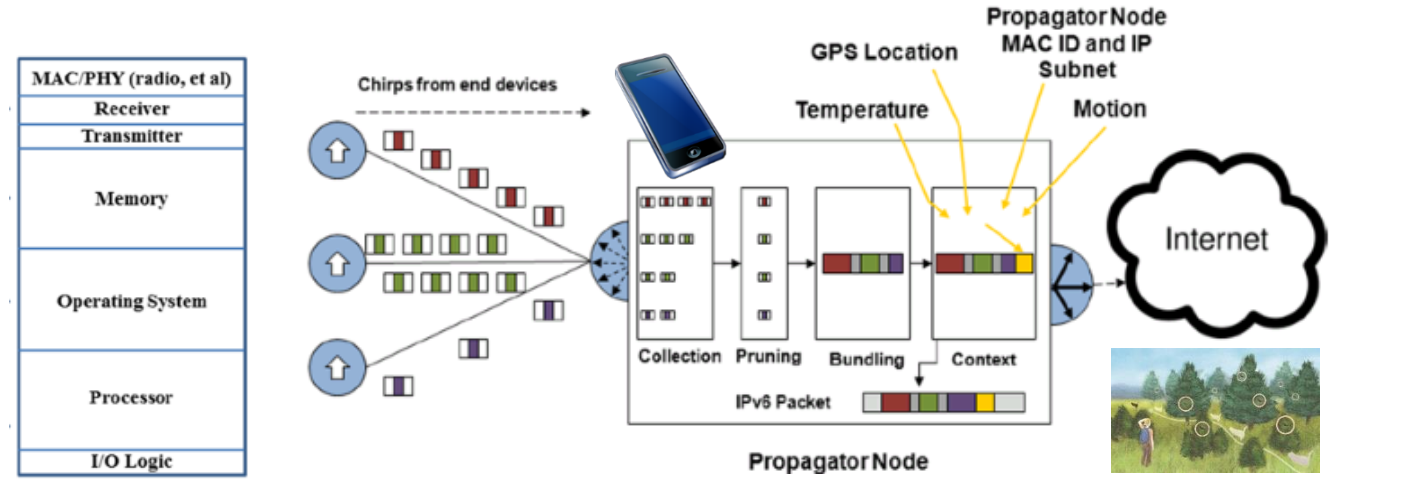
Trusted Sensor → Data Log → Trusted Propagator → Cloud

Sensor collects data

The FreeStyle Libre CGM sensors are small, unobtrusive, and easily applied[§5,13] to the back of your upper arm. The sensor continuously measures and stores glucose readings when worn on your body[1-3].

112↗

Glucose readings are displayed on your smartphone[¶#]**

The sensor sends your data to a device[¶#**††], where you can view your glucose readings, reports, and trends[1-3] anytime[‡‡], anywhere[§§].

1. Enterprise "Imprinted" Sensor

1. Sensor is applied

Most CGM sensors are tiny self-applied devices that sit partially beneath the skin.[1] They come with an inserter and have built-in adhesive to help them stick to your skin.[1]

2. Data is collected

The sensor measures glucose levels at regular intervals and stores the readings.

3. Scan using a reader or smartphone[‖]

A handheld reader allows you to scan the glucose sensor to collect your information. You may be able to use a smartphone app[‖] so that you don't have to carry around an extra device.

2. Enterprise Propagator

5.3↗

LibreView

3. "Enterprise" Trusted Subscribers and Apps

Trusted Device ➔ Trusted Propagator Network➔ Private Cloud

Noisy Lossy Public Network

2. Propagators Friend to Friend Network

AIR QUALITY MONITOR

ECHO

WILDFIRE ALERT

PET TRACKER

NOISE EMISSION MONITOR

SMART YARD LIGHTING

RING FLOODLIGHT CAM

2

RING VIDEO DOORBELL

ENVIRONMENTAL CONTROL

CO_2 SENSOR

RING SPOTLIGHT CAM

CONNECTED VEHICLE

1

REMOTELY-MANAGED STREET LIGHT

SIDEWALK BRIDGE PRO

1. IoT Device ("Edge") New Friend

3. AWS IoT Core -- Cloud Messaging Services 3

How it works

With AWS IoT Core for Amazon Sidewalk, developers and device makers can experience a simple setup, onboarding, and registration of devices. Sidewalk-enabl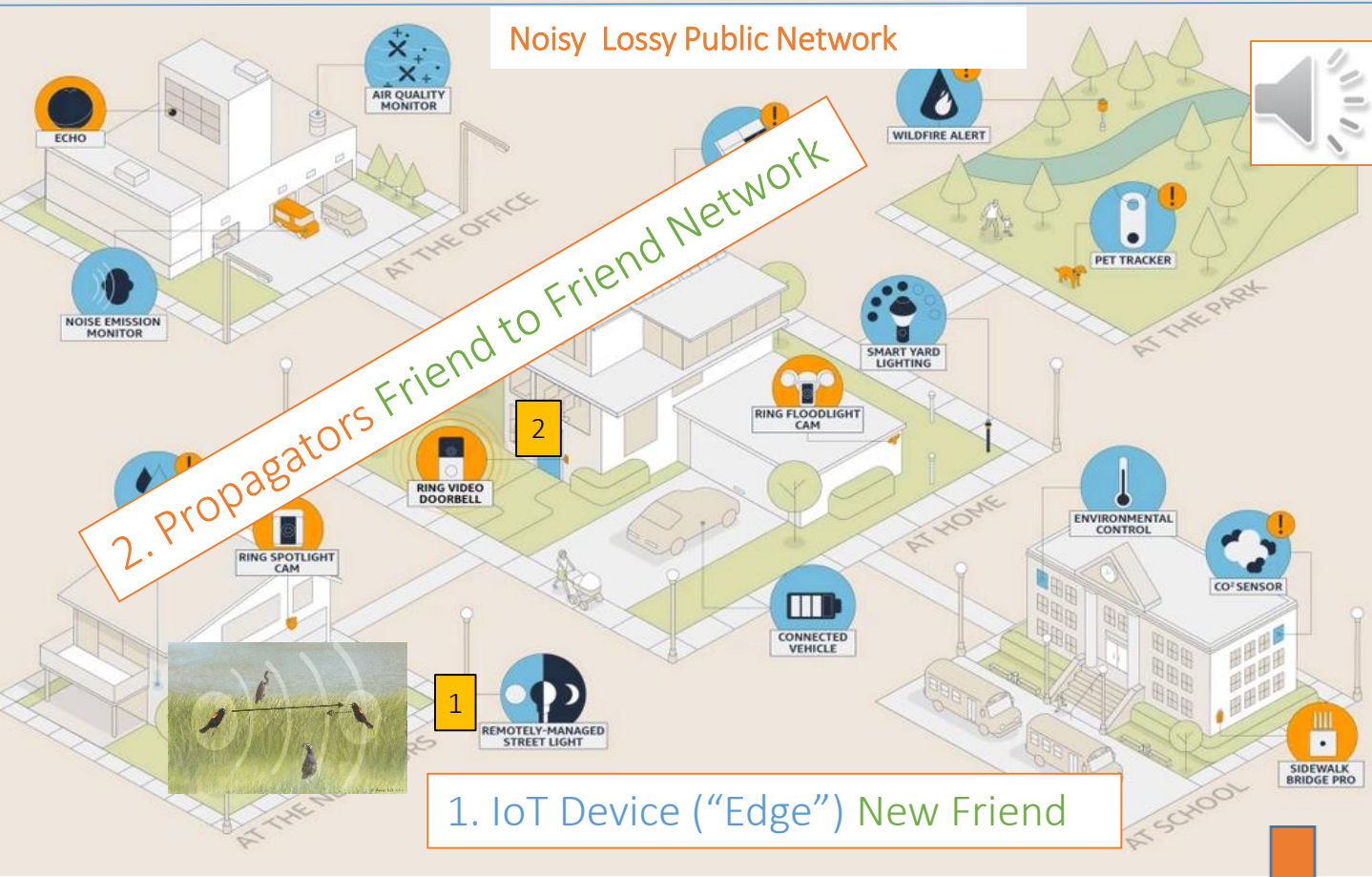ed partner devices come pre-provisioned with security certificates required to establish an encrypted connection with AWS IoT services. From the AWS IoT Console, developers can easily configure their Sidewalk-enabled devices and access a range of AWS services to build scalable solutions.

Hosted Pub/Sub Based Services (e.g. Amazon IoT Core)

IoT device →

Amazon Sidewalk
IoT devices can use the free-to-connect Amazon Sidewalk network to create long-range connected solutions.

Amazon Sidewalk Bridges
Sidewalk Bridges are devices, such as Amazon Echo and Amazon Ring products, that provide connections to Amazon Sidewalk network. Users can enable or disable Amazon Sidewalk functionality on these bridge devices.

AWS IoT Core for Amazon Sidewalk
Sidewalk devices come pre-provisioned with security certificates to securely and quickly connect to AWS using AWS IoT Core for Amazon Sidewalk feature.

AWS IoT Core Rules Engine
Messages are routed through AWS IoT Core Rules Engine to other AWS Services.

Amazon SageMaker

Amazon S3

Additional AWS services

AWS Services

Trusted Device ➔ Trusted Propagators ➔ Logically Connected Clouds

Group Edge Intelligence From Friend NATO Forces

"British"
Private Community

3 "Cloud-VPN" DTN Back-haul

"American"
Private Community

2 Trusted Pigeons

2 Trusted Pigeons

RBWN DTN Region 2

1

1
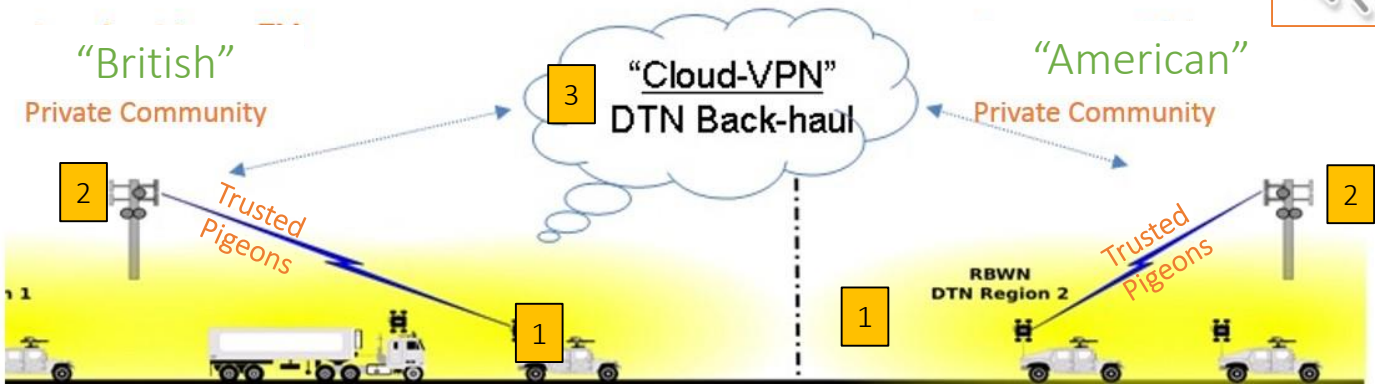
Military USAF & NAVWAR Disruption Tolerant Networks, Field Tested.

NAVWAR

Ministry of Defence — Defence Research and Development Canada

Department of the Navy

United States Coast Guard — U.S. Department of Homeland Security

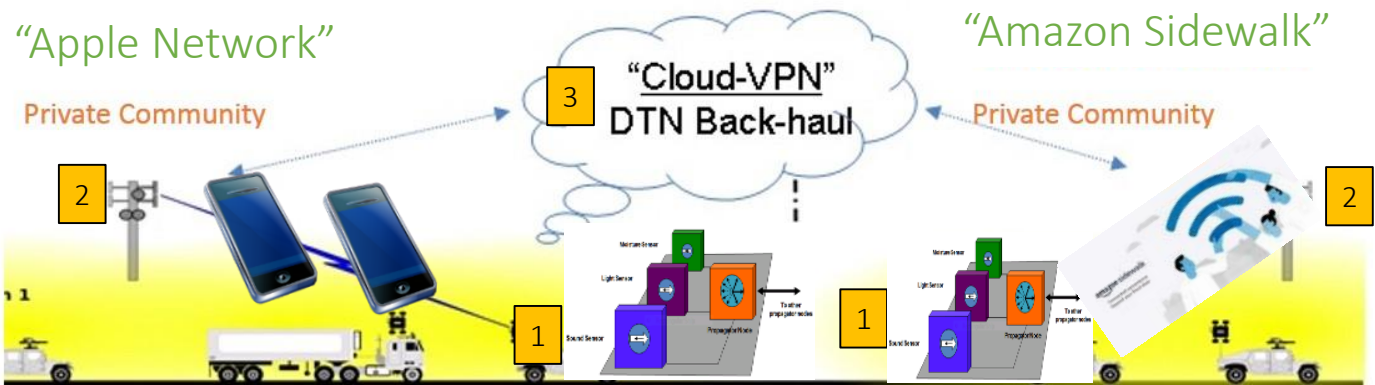Sandia National Laboratories

Industry SHARP Modular Mesh with Sub 1 GHz Modem. Field Tested.

SHARP

A Globally Scalable version of Messaging across Closed Systems.
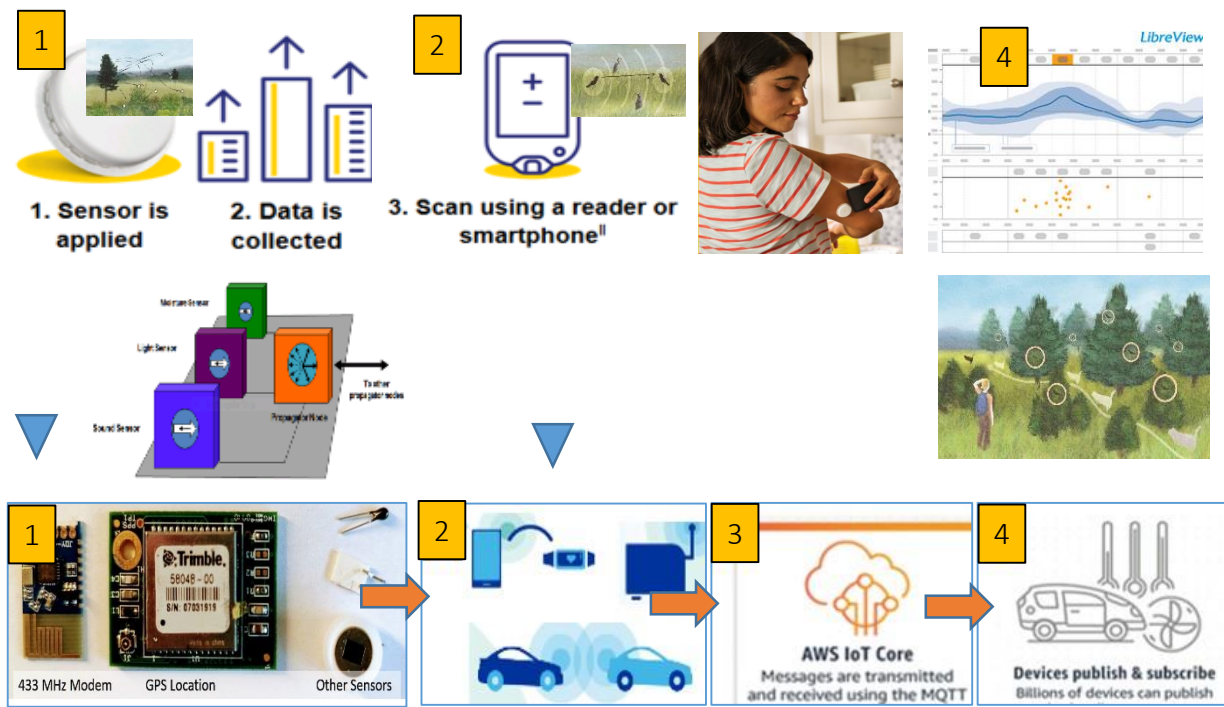*Enterprises now define and create their own imprinting protocols.*

Group Edge Intelligence From Friend Simple Edge Devices

"Apple Network"
Private Community

3 "Cloud-VPN" DTN Back-haul

"Amazon Sidewalk"
Private Community

2

2

1

1

Trusted Device → Trusted Propagators → Logically Connected Clouds

Introducing Soft Chips™ – An Enterprise Imprinted Protocol Agnostic "Sensor Patch"

1. Sensor is applied **2. Data is collected** **3. Scan using a reader or smartphone"**

1. 433 MHz Modem   GPS Location   Other Sensors

2

3 AWS IoT Core — Messages are transmitted and received using the MQTT

4 Devices publish & subscribe — Billions of devices can publish

Soft Chips™ are Enterprise versions of Medical Sensor Patch work flow (above).

*Imprinted Sensor ->Data Logs -> Wakes up when Propagator (Carrier Pigeon) Arrives.*

Carrier Pigeon Tags Data Log -> Cloud -> Publish

Soft Chips – a packaged set of sensors – are programmed through imprinting (establishing Provenance).

  1. Sensor Patch is non-invasively attached on Edge Asset being tracked or monitored.
  2. Cheap Wireless Serial Modems connect to available opted-in Carrier Pigeons – Cars, Drones, Phones.
  3. Sensor based status logs are tagged by Carrier pigeon and forwarded to IoT messaging brokers.
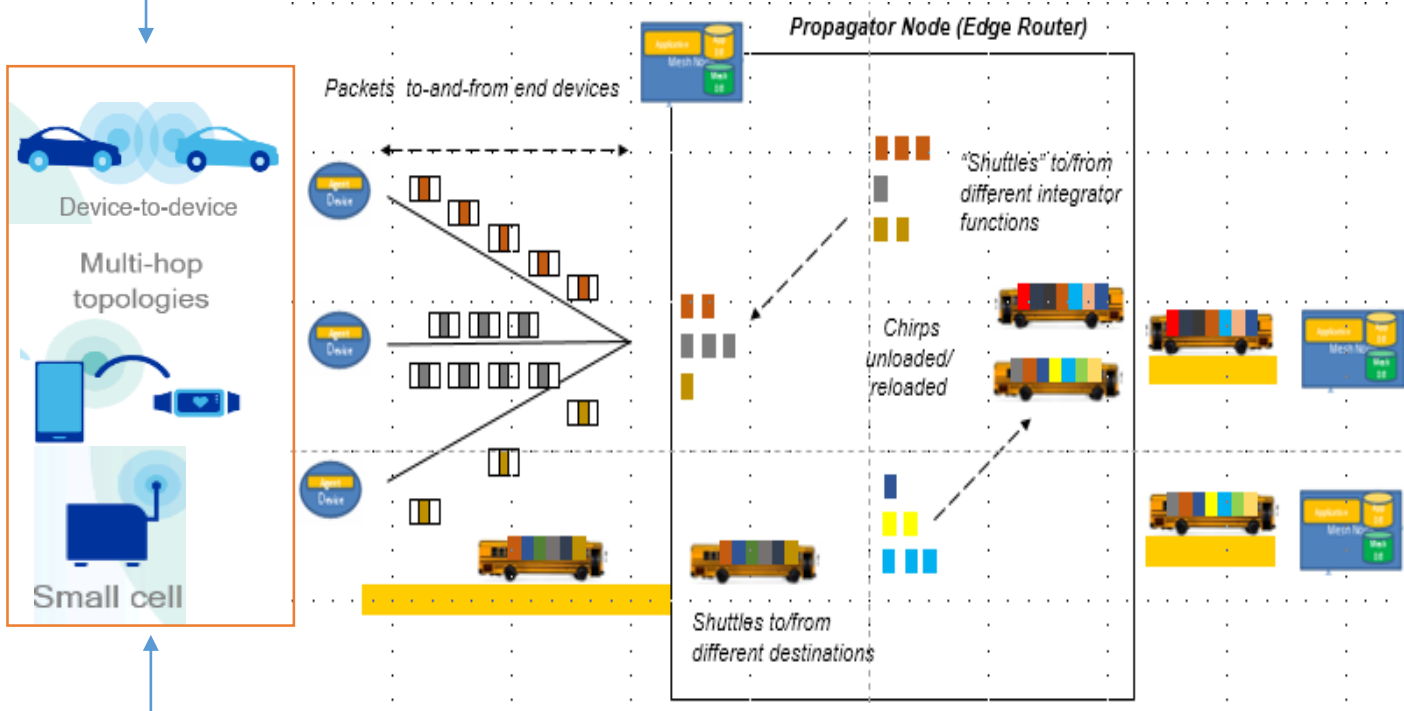 4. Publish Subscribe messaging brokers distribute to all interested subscribers- human and digital.

Cheap sensors like microphones with rudimentary ant-like intelligence can detect "bad" sounds from motor bearings and drive predictive maintenance. Massively Scalable and Sustainable Iot

1. Enterprise Imprinted Soft Chips (e.g. Sensor Patch, RFID++)
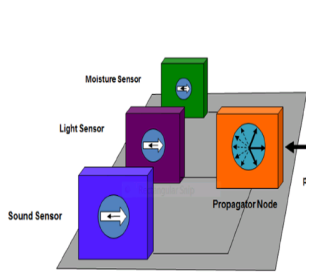
- **Minimal**, Legacy Friendly Protocol & Hardware.
- **Imprinted** at Birth – Zero trust based.
- **Distinguishable** but Undecipherable Chirps.
- **Piggybacks** on Ubiquitous Messaging Networks (SMS)
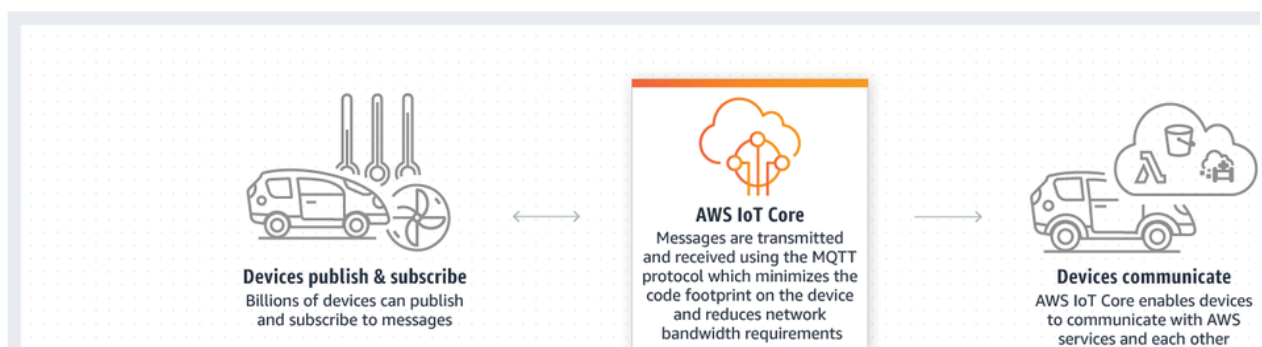- **Small** Dumb Cheap Copious + Secure → "Massive IoT"

Moisture Sensor
Light Sensor
Sound Sensor
Propagator Node

2. Enterprise Imprinted Propagators (& Shuttle Bus Services)

Device-to-device

Multi-hop topologies

Small cell

Propagator Node (Edge Router)

Packets to-and-from end devices

"Shuttles" to/from different integrator functions

Chirps unloaded/ reloaded

Shuttles to/from different destinations

3. Protocol Agnostic Cloud Orchestrators

Message Broker   **Mirror Device State**   **Built-in Alexa**   **LoRaWAN Devices**   **Amazon Sidewalk Devices**

Devices publish & subscribe
Billions of devices can publish and subscribe to messages

AWS IoT Core
Messages are transmitted and received using the MQTT protocol which minimizes the code footprint on the device and reduces network bandwidth requirements

Devices communicate
AWS IoT Core enables devices to communicate with AWS services and each other

Above: Key Challenges to "Massive IoT" (Qualcomm) : Low Energy, Low Complexity, High Density + High Security.

The only systems on earth that have ever scaled to the size & scope of the Internet of things are natural systems: pollen distribution, ant colonies, redwoods, and so on. This presentation outlines how Massive IoT may be achieved by rethinking last mile connectivity.

Today's last mile is crippled by proprietary transport protocols, not sustainable. Edge➔Cloud thinking focused on smarts at the radio end to manage collision avoidance by segmenting collision domains in RF space (channel diversity) and time (time reservation slots). Current IoT Radios have to be "smart" because they use phones and computers as carrier pigeons to connect. These devices, intended for humans are rechargeable and thus not energy constrained. They can afford power hungry communication protocols. In order to communicate, the IoT edge devices must then also conform to protocols and RF channels supported – or provide private networks.

Conversely, in Cloud->Edge thinking, we let Clouds manage ant-like reprogrammable intelligence – purpose driven, minimal processing and thus low power usage and cheap. A 433 MHz wireless modem costs $0.10. A WiFi and Bluetooth wireless chipset costs $20. Also CSMA/CA protocols are inherently inefficient and insecure without "heavy" encryption on top of a heavy protocol. An IP header is 40 bytes vs. 1-5 bytes for Chirp headers.

Chirp protocols and their minimal hardware cleanly cut through all Gordian knots in Fig. above.

The Edge *can* be small dumb cheap and copious**.** Think dense sensor grids for forest fires, climate preparedness.

Key Points addressed in this presentation relate to "Cloud➔Edge" thinking.

  A.   Global-Scale "Edge" challenges are: simplicity, cost, energy & (as always) security.
  B.   Chirpers don't need heavy OSI stack -> minimal power and cost for connectivity.
  C.   Software Driven Mesh for the Edge -> Moves Chirping Edge Intelligence to Cloud.
  D.   Trusted walled gardens become globally relevant through our imprinted chipsets.
  E.   Massive IoT – burgeons with Cloud messaging and Distributed AI ➔ Globally Relevant.

Please see Chirp Primer - intended as a prelude for these slides - for more.
Thank you for your consideration. Your feedback is welcomed. Francis daCosta Jan 2024.

Introduction, "Rethinking IoT"   Intel Press, 2013.   177K Free downloads.   Jolt Award   Amazon reviews

I didn't set out to develop a new architecture for the Internet of Things (IoT). Rather, I was thinking about the implications of control and scheduling within machine social networks in the context of Metcalfe's Law. The coming tsunami of machine-to-machine interconnections could yield tremendous flows of information – and knowledge.

Once we free machine social networks (comprised of sensors and other devices) from the drag of human interaction, there is tremendous potential for creating autonomous communities of machines that require occasional interaction or reporting to humans.

The conventional wisdom is that the expansive address space of IPv6 solves the IoT problem of myriad end devices. But the host-to-host assumptions fossilized into the IP protocol in the 1970s fundamentally limited its utility for the very edge of the IoT network.

As the Internet of Things expands exponentially over the coming years, it will be expected to connect to devices that are cheaper, dumber, and more diverse. Traditional networking thinking will fail for multiple reasons.

First, although IPv6 provides an address for these devices, the largest population of these appliances, sensors, and actuators will lack the horsepower in terms of processors, memory, and bandwidth to run the bloated IP protocol stack. It simply does not make financial sense to burden a simple sensor with the protocol overhead needed for host-to-host communications.

Second, the conventional implementation of IP protocols implies networking knowledge on the part of device manufacturers: without centrally authorized MAC IDs and end-to-end management, IP falls flat. Many of the hundreds of thousands of manufacturers, building moisture sensors, streetlights lack the expertise to implement legacy network technology in traditional ways.

Third, the data needs of the IoT are completely different from the global Internet. Most of the communications will be terse machine-to-machine interchanges that are largely asymmetrical, with much more data flowing in one direction (sensor to server, for example) than in the other. And in most cases, losing an individual message to an intermittent or noisy connection will be no big deal. Unlike the traditional Internet, which is primarily human-oriented (and thus averse to data loss), much of the Internet of Things traffic will be analyzed over time, not acted upon immediately. Most of the end devices will be essentially autonomous, operating independently whether anyone is "listening" or not. More

Fourth, when there are real-time sensing and response loops needed in the Internet of Things, traditional network architectures with their round-trip control loops will be problematic. Instead, a way would be needed to engender independent local control loops managing the "business" of appliances, sensors, and actuators while still permitting occasional "advise and consent" communications with central servers.

Finally, and most importantly, traditional IP peer-to-peer relationships lock out much of the potential richness of the Internet of Things. There will be vast streams of data flowing, many of which are unknown or unplanned. Only a publish/subscribe architecture allows us to tap into this knowledge by discovering interesting data flows and relationships. And only a publish/subscribe network can scale to the tremendous size of the coming Internet of Things.
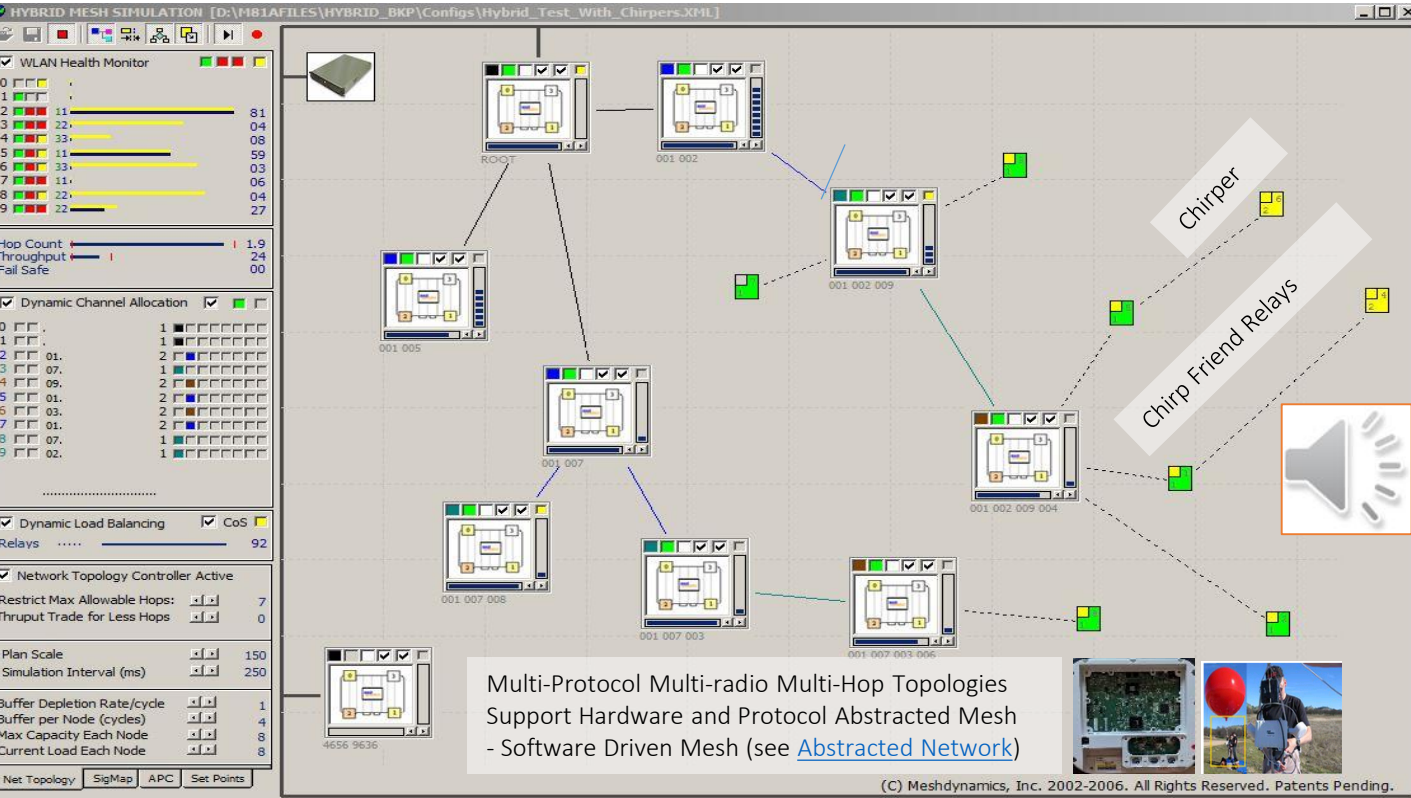
The only systems on earth that have ever scaled to the size and scope of the Internet things are natural systems: pollen distribution, ant colonies, redwoods, and so on. From examining these natural systems, I developed the three-tiered IoT architecture described in this book: simple end devices; networking specialist propagator nodes, and information-seeking integrator functions.

In these pages, I'll explain why terse, self-classified messages, networking overhead isolated to a specialized tier of devices, and publish subscribe relationships formed are the only way to fully distill the power of the coming Internet of Things.
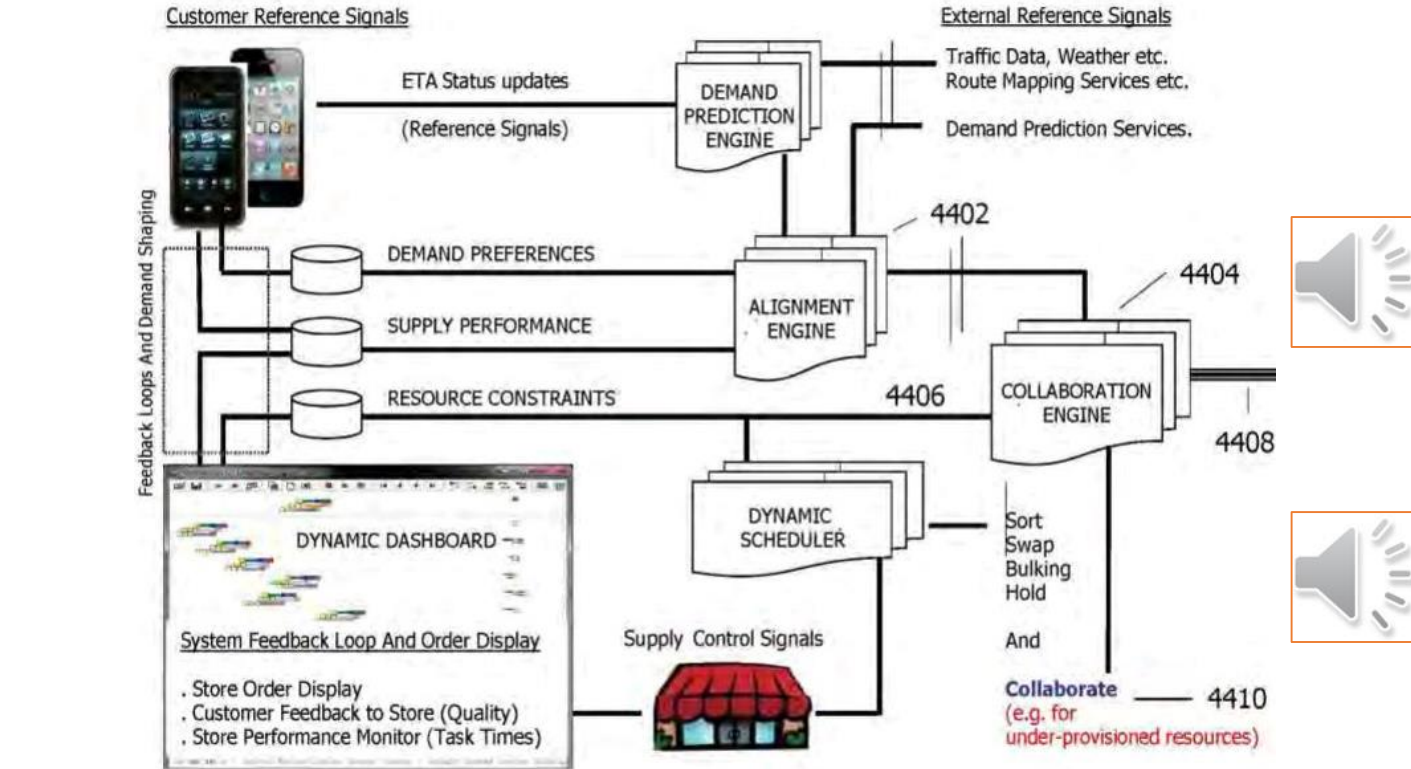
Francis daCosta Santa Clara, California, 2013

### Rapid Convergence of Distributed AI & SDN → Cloud Orchestration → Massive IoT

A. Adaptive Last Mile Mesh Networking Architectures driven by Subscriber Demands   Enlarge   Animation

Multi-Protocol Multi-radio Multi-Hop Topologies Support Hardware and Protocol Abstracted Mesh - Software Driven Mesh (see Abstracted Network)

(C) Meshdynamics, Inc. 2002-2006. All Rights Reserved. Patents Pending.

B. Cloud Orchestrated Work Flow Scheduling - Aligning Supply and Demand Chains Ver. 2012   Animation

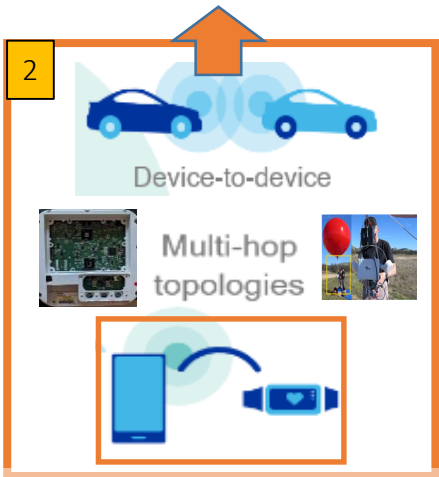## Imprinted Device → Imprinted Pigeons → Message Brokers → Subscribers
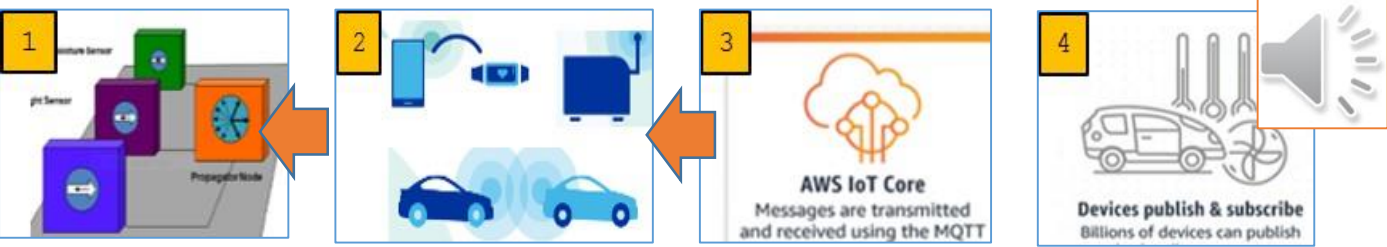
Imprinted Sensor -> Logs -> Modems -> Pigeons -> Tagging -> Cloud -> Publish -> Actionable Edge Intelligence

Enterprise Imprinted Soft Chip

Enterprise Imprinted Pigeons

Cloud Orchestrator (hosted or App)

Exemplary Soft Chips™ Imprinting Work Flow

A. Sensor Pack Order Fulfillment (Enterprise supplies Soft Chips)
1. Customer selects Enterprise Provided Soft Chip e.g. Sensor Patch.
2. Enterprise delivers unit with QR Code
3. Customer attaches Sensor Patch to Legacy Edge Asset.
4. Enterprise App on Phone takes picture of installed Sensor Patch
5. Asset is Fingerprinted, with phone camera, e.g. Alitheon

B. Use Carrier Pigeon to Scan Local RF channels (Collision Avoidance)
1. Enterprise phone radio or USB radio scans RF interference.
2. Information relayed to cloud and/or Phone App has User entry.
3. Initial RF Channel and Schedule proposed and accepted.

C. First Power Up, Pairings, Establishing Provenance Chain
1. Customer removes battery insulation. Unit powers up.
2. Unit searches for "Mother" per Enterprise first time "Birthing".
3. Chirp aware receiver radio – on carrier Pigeon – responds.
4. Switch to private channels. (Control Plane Messaging)
5. Carrier Pigeon App also switches to private Channel.
7. Chirp_ID, Instance_ID, imprinted – Unit is "birthed" in situ.
8. Time/Location Synced to Pigeon. Unit given or infers time slot.
9. Pairing and Provenance established. Joins Chirp-ID Group.

D. Delivery of data logs per schedules defined in Imprints
1. Carrier pigeon visits at scheduled delivery time range
2. Carrier pigeon sends beacons on Data Plane Channel
2. Unit listen-sensor detects trusted Carrier pigeon in range.
3. Pigeon stores broadcasted data logs sent per time slot allocated
4. Pigeon tags received data logs container for audit trail.

E. Delivery of next Imprinted Code settings
1. Per Schedule, Unit listens on Control Plane Channel for Pigeon Arrival
2. Pigeon broadcasts to all listening. Mobile drone covers entire Grid.
3. Audit trail logs updated to cloud via pigeon and Connected Devices.
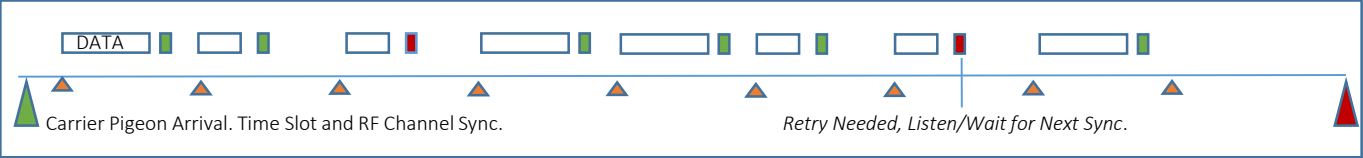
0. Select Channels Needed (worst case A,B,C,D)

A- Factory Default (Pre Birthing)
B- Temporal Instance Private Channel Switch and Settings for Birthing
C- Group Channel for Pigeon Sending Broadcast of New Schedules/Pgms (Control Plane)
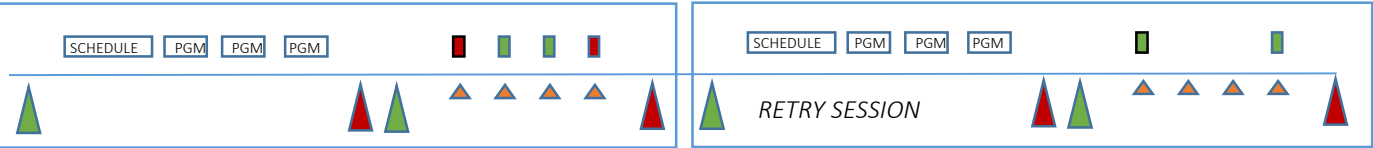D- Group Channel for Sending Timed Data Logs sent in broadcast (Data Plane)

1. Birthing Sequence

A   Power up unit, connect on Sub-1 GHz modem, transmit private channel & settings
B   Unit switches to private channel, imprinted with Chirp_ID, Chirp_Instance etc.
B/C   Birthed units receive Private/Group schedule & Pgms
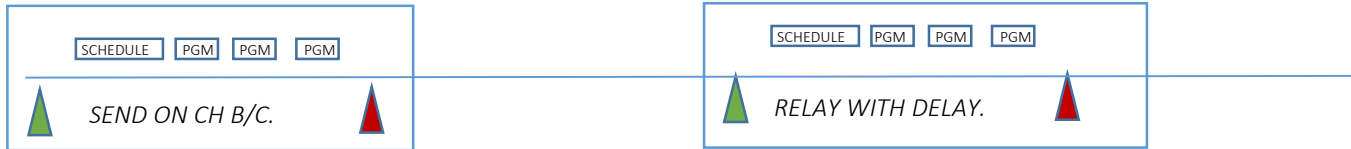B/C   Imprinted Units returns Status/ACK per reserved time slot in Group Schedule

2. Collecting Time Slotted Data Logs   Edge→Pigeon (Channel D, Data Plane)

DATA

Carrier Pigeon Arrival. Time Slot and RF Channel Sync.   *Retry Needed, Listen/Wait for Next Sync.*

3. Group Broadcast of New Schedules & Pgms   Pigeon→Edge (Channel C Control Plane)

SCHEDULE   PGM   PGM   PGM

SCHEDULE   PGM   PGM   PGM   *RETRY SESSION*

4. Internal Relay between Group Members   Edge→Edge (Wild Fire Mode Ch. B/C)

SCHEDULE   PGM   PGM   PGM   *SEND ON CH B/C.*

SCHEDULE   PGM   PGM   PGM   *RELAY WITH DELAY.*

5. Exemplary Terse Protocol Framing (for Minimal SMS-Like Messaging)

   1. Birthing: |Chirp_ID | Chirp_Instance |Version etc. |
               |1 Bytes = 255 Species| 1 Byte = 255 units | 3 Bytes | = Total 5

   2. Internal sensor data mapped to Black/Red/Yellow/Green = 2 bits suffice. 4 such sensor feeds = 1 Byte.
    Thus 100 recent samples (Circular Q) = 100 bytes.

   3. Data Logs |Chirp_ID|Chirp_Instance|Version| + 100 Byte ASCII CSV Data Log = 105 Bytes

   4. SMS Message Broker Payload - add GPS and UTC time for audit trails = 160 Byte Max

   5. Payload of a patrol drone servicing a 100 4-sensor Sensor Grid is 16 KB.

   6. Payload sent to ubiquitous SMS-like services become globally relevant.

A — Apps on Connected Devices

B — Simple Modem based Pigeons

C — Data Logs from Edge Sensors
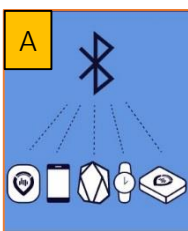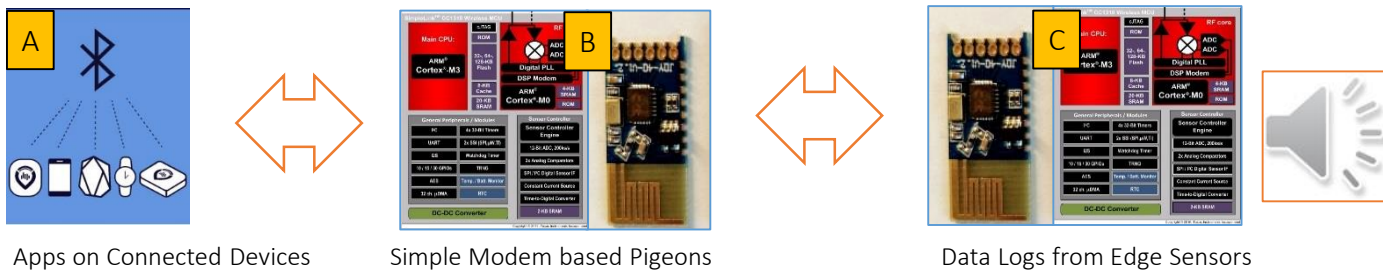
Apps on Connected Devices (Phone) "Orchestrates"
|
|- Downloads Configurable Exemplary Templates
|- Downloads Schedules/Pgms to Pigeon over BLE/USB
|- Uploads Data Logs back from pigeon over BLE/USB
|- Leverage Device->Message Brokers (SMS, IoT Core)
|- Leverage existing mesh networks e.g. BLE Mesh etc.
|- Extend or restrict network access as needed.
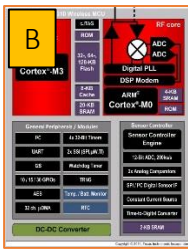
Carrier Pigeon Store and Forward Duties
|
|- Uploads Configured Templates for Schedules
|- Uploads associated programs for sensors selected
|- Store and Forward to Soft Chip per Schedules
|- Return with Data Logs and Upload to Connected Device
|- Sync UTC Time+Location on Soft Chips (Broadcast or private mode)
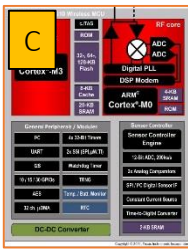
Soft Chip Harvests Sensor Data, waits for pickup
|
|- Uploads Schedules & Pgms from Pigeon over Modem
|- Runs wake up timers to collect data
|- Runs wake up timers to "listen" for pigeon arrival
|- Downloads data logs via Modem to Pigeon. Receives ACK
|- Syncs UTC Time+Location with Pigeons, resets timers
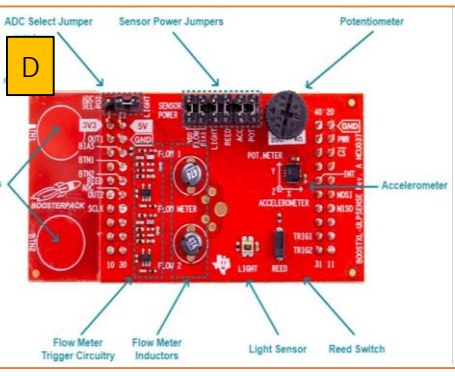
Extensive Development Environment for Prototyping
|
|- Prototype Enterprise Specific Applications
|- Supports multiple general purpose inputs
|- Experiment with IFTTT rules and Schedules
|- Emulate Field testing through multiple devices.
|- Generate user configurable templates Library
|- Leverage Community to share Apps and templates.
|- Engender IoT Systems Developers to Think "Thin"

Related to above: Sidewalk™ Toolkit   Launchpad   CC1312R7   SensorTag   FreeRTOS