BUILDING THE WEB OF THINGS

For the past 10 years, one layer at a time, one million things at a time...

Smarter products come with EVRYTHNG
<table>
<thead>
<tr>
<th><strong>Company Overview</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model:</strong></td>
</tr>
<tr>
<td><strong>History:</strong></td>
</tr>
<tr>
<td><strong>Founders:</strong></td>
</tr>
<tr>
<td><strong>Team:</strong></td>
</tr>
<tr>
<td><strong>Key Investors:</strong></td>
</tr>
</tbody>
</table>

Serving global brands that require SCALE
Market opportunity and inevitability

- 83 Million cars
- 2.3 Billion computing devices
- 6.0 Billion RFID chips
- 19 Billion microcontrollers

- 80 Billion apparel items
- 5-10 Trillion consumables

Source: ThinFilm (IDC; Gartner; World Bank; IMF; HIS; The Semiconductor Industry Association; OICA; IC Insights; Market Line; Apparel Market; Planet Forward; Companies & Markets)
EVRYTHNG - Smart Products Platform

MANUFACTURER & SUPPLY CHAIN APPS

CONSUMER MOBILE APPS

ANY PRODUCT, CHIP, SENSOR OR TAG

BI & CMS SYSTEMS

CLOUD & SOCIAL PLATFORMS

ENTERPRISE & SUPPLY CHAIN SYSTEMS

DEVELOPER TOOL

INTEGRATIONS

ADMINISTRATIONS & ANALYTICS

REAL-TIME DATA MANAGEMENT

SECURITY AND ACCESS CONTROL

CLOUD PLATFORM-AS-A-SERVICE

PRODUCT CONNECTION MANAGEMENT
Today’s menu

- From IoT to WoT
- Building the Web of Things
  - Layer
  - Discussion
  - Case-study @ EVRYTHNG
From IoT to WoT
Bootstraping the Web of Things
What are the Things in the IoT?

- **Tags**
  - NFC/RFID Tag
  - QR Code

- **Devices**
  - Arduino
  - iBeacon/BLE

- **Machines**
  - Philips Hue
  - Raspberry Pi

- **Environments**
  - Smart Building
  - Smart Car
  - Smart City

*Computational power and complexity*
Research Question: «How can the Web be leveraged to ease the development of Internet of Things applications and bring it closer to non-specialists?»
The Web of Things vs the IoT

"I hate my life!"

"Easy-peasy!"
Building the Web of Things
Layer by Layer
The Web of Things Architecture

Layer 4: COMPOSE
- Systems Integration
- IFTTT
- Node-RED
- Automated UI Generation
- WoT-a-Mashup
- Physical Mashups
- Web Applications

Layer 3: SHARE
- Social Networks
- Delegated Authentication
- Social WoT
- API Tokens
- TLS
- Encryption
- DTLS
- OAuth
- JWT
- PKI

Layer 2: FIND
- REST Crawler
- HATEOAS
- Link Header
- Web Thing Model
- Search engines
- Schema.org
- Linked Data
- RDFa
- JSON-LD
- Semantic Web
- mDNS

Layer 1: ACCESS
- HTML
- JSON
- Web Hooks
- REST API
- Proxy
- HTTP
- WebSocket
- CoAP
- URI / URL
- Gateway

Networked Things
- NFC
- 6LoWPAN
- Thread
- Ethernet
- Wi-Fi
- QR
- Beacons
- Bluetooth
- ZigBee
- 3/4/5 G

Source: Building the Web of Things, book.webofthings.io
Creative Commons Attribution 4.0
0. The Network
1. Choose a Physical Protocol

1. Physical (Link)
   - Ethernet
   - Wi-Fi
   - IEEE 802.15.4

2. Data Link
   - Thread Stack
   - MAC

3. Network
   - Bluetooth Stack
   - 6LoWPAN IP (v4, v6)

4. Transport
   - Zigbee Stack
   - EnOcean Stack
   - TCP
   - UDP

5. Session

6. Presentation

7. Application

Internet Protocols Suite (TCP/IP)
- HTTP
- WebSocket
- MQTT
- MQTT-SN
- CoAP
- DNS
- XMPP
- TLS
- SSL

OSI
- Physical (Link)
- Data Link
- Network (Internet)
- Transport
- Session
- Presentation
- Application
2. Choose a Network Protocol: IPv4 VS IPv6, 6LoWPAN

**IPv4 Address**

146.200.15.222

- 8 bits
- $4 \times 8 \text{ bits} = 32 \text{ bits}
- $2^{32} = \sim 4.3 \text{ billion addresses}$

**IPv6 Address**

2001:db8:0:1234:0:567:8:1

- 16 bits
- $8 \times 16 \text{ bits} = 128 \text{ bits}
- $2^{128} \text{ addresses}$
3. Choose a Transport Protocol: TCP vs UDP
Dom Perignon Button

- Get Champagne at the click of a button
- Drop it in a room!
  - Thick walls
  - Wi-Fi requiring browser login
- 2G – SMS connectivity
Discussion: Layer 0 – The Network

- Fragmentation needs to be resolved to fulfill the promise of the IoT!
- There will not be one protocol to rule them all!
  - Power consumption & battery power
  - Environment of a deployment
- Convergence to Internet Protocols
  - IPv6 (6loWPAN, TCP/UDP)
- Consolidations will take place (and need to!):
  - PAN: Zigbee 3.0 – Thread - Wi-Fi HaLo
  - LPWAN: SigFox – 5G
- Pick the simplest path:
  - ~10 years for routers to be rolled out
1. Access Layer

- Networked Things
- NFC
- 6LoWPAN
- Thread
- Ethernet
- Wi-Fi
- 3/4/5 G
- Bluetooth
- ZigBee
- QR
- Beacons

Layer 1

ACCESS

HTML
JSON
WebSockets
Proxy
MQTT
CoAP

REST API
HTTP

Proxy
WebSockets
Gateway
URI / URL
Web Hooks

1. Access Layer
Basic principle: a URL for each Thing

http://tn.gg/JANFvB4u
And a RESTful API

Root URL of Raspberry Pi devices.webofthings.io/pi

- **actuators**
  - /actuators

- **LEDs**
  - /leds

- **LED #**
  - /{led#}

- **sensors**
  - /sensors
  - **light sensor**
    - /light
  - **temperature sensor**
    - /temperature

- **buttons**
  - /buttons

- **acceleration**
  - /accel

- **gyroscope**
  - /tilt

Demo!
The real-world is event driven

Client App   Web Thing   Sensor via GPIO

Request →

Response ←

Request ↓

Response ←

New Value ←

Request ↓

Response ←

New Value ←

Request ↓

Response ←

New Value ←
The Web has WebSockets

Demo!
WebSocket Upgrade in the browser
Not all devices can speak HTTP and WebSockets, can they?
Not all devices can speak HTTP and WebSocket

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Application</td>
<td>MQTT</td>
<td>MQTT-SN</td>
<td>CoAP CoRE</td>
</tr>
<tr>
<td>3. Transport</td>
<td>TCP</td>
<td>UDP</td>
<td>UDP</td>
</tr>
<tr>
<td>2. Network (Internet)</td>
<td>IP</td>
<td>Not specified</td>
<td>6LoWPAN</td>
</tr>
<tr>
<td>1. Physical (Link)</td>
<td>Not specified</td>
<td>Not specified</td>
<td>IEEE 802.15.4</td>
</tr>
</tbody>
</table>
Integration pattern: direct communication
Integration pattern: Gateway

Web Thing Clients

Wi-Fi Router

HTTP/WS via Wi-Fi

HTTP/WS via Wi-Fi

Wi-Fi Router

Bluetooth

CoAP

WoT API

Web Things
Integration pattern: Cloud

WT Clients
HTTP/WS via Wi-Fi

Cloud Service
HTTP/WS via 3G/4G

WoT API

MQTT

WAN/Internet

Web Things

Your Local Network at Home
iHome uses EVRYTHNG for their next-gen family of smart home products

- Launched SmartPlug in July 2015, with suite of other products in development
- One of just 5 initial HomeKit certified products
- Uses out-the-box Marvell toolkit for devices with MQTT support
- Integrated with SmartThings, Wink and Nest, and with iHome CRM and support system
- Android and iOS apps for setup, creating scenes, timers and granting access to other users

Watch video
Gooee uses EVRYTHNG to sell Lighting-as-a-Service

- Transforms dumb lights into smart services
- Smart bulb for remote control, with motion sensors for retail traffic monitoring & security
- Energy management & lower maintenance costs
- Greater control and flexibility
THNGHUB: a WoT Gateway

- Multi-protocol support including Zigbee, Bluetooth, WiFi, Ethernet
  - Modular protocols support via additional plugins
  - Any language (DAL)
- Local Web API via HTTP/REST, WebSockets, M2M API via MQTT
- Local version of EVRYTHNG’s Reactor™ Rules Engine
- Runs on Linux (ARM or x86/64), supports most gateways architectures
- Deployed as secure, virtualized docker containers on any Linux appliance
IoT World’s Largest Deployment

IN THE NEWS

AVERY DENNISON AND EVRYTHNG SWITCH ON THE APPAREL INDUSTRY WITH 10 BILLION PRODUCTS IN WORLD’S LARGEST IOT DEPLOYMENT

Apr 18, 2016

Apparel and footwear products from the world’s largest brands have the power to be born digital and given unique, item-level digital identities with the Janela™ Smart Products Platform

- 10 Billion products through Avery Dennison labels
- Apparel products are “born digital”
- Based on the simple concepts of URLs and Web API for each thing
Discussion: Layer 1 - Access

- All protocols need to meet at the Application Layer
  - The Web
- Other protocols can be translated to HTTP/WS
  - MQTT (quality of service, remote actuation)
  - CoAP (battery, low power)
- Pick the simplest path:
  - Simplicity **does** matter!
  - UDP NAT traversal issues, etc.
2. Find Layer

Layer 2
- REST Crawler
- HATEOAS
- Link Header
- Web Thing Model
- Search engines
- JSON-LD
- Linked Data
- RDFa
- REST API
- Gateway MQTT CoAP

Layer 1
- HTML
- JSON
- Web Hooks
- WebSockets
- REST API
- HTTP
- MQTT
- CoAP

Networked Things
- NFC
- 6LoWPAN
- Thread
- Ethernet
- Wi-Fi
- QR
- Beacons
- Bluetooth
- ZigBee
- 3/4/5G

ACCESS
Web Thing Model

http://model.webofthings.io
http://gateway.webofthings.io
Web Things Model

Web Thing Clients

- Native Mobile App
  - Discovers Web Thing

- Web App
  - Create Actions
  - Read / Subscribe to Properties

- Web Thing
  - Control Non-Web Things

Web Thing

- URL: `http://gateway.webofthings.io`
- /model
  - Name, Description, Tags
  - Actions/Properties model

- /actions
  - ledState
  - reboot
  - displayText

- /properties
  - `Temperature`: 1,221
    - Time Online: 00:05:59
  - `Light`: 579
    - Humidity: 33.99%

- /things
  - Health Monitor
    - LilyPad

Non-Web Devices

- Bluetooth
- ZigBee
An example

Demo!
Semantic Smart Home Integrations

**Platform integrations** via Cloud to Cloud modules
- Enables devices to be managed by EVRYTHNG & accessible and controlled from 3rd party systems and apps. e.g. Nest, Wink or SmartThings
- Integration module includes:
  - User & device mapping
  - Synchronization
  - Custom Logic

**Local integrations** via THNGHUB Local Cloud Gateway
- Enables interoperability with Homekit & Weave
- Connection to 3rd party products via device APIs
Using Web protocols is the first step towards true interoperability
  - Helps “opening” devices, freeing them through APIs
Web protocols cover the “How” not the “What”
  - Great for humans, challenging for machines
  - Semantic Web, JSON-LD
  - First W3C proposal for the Semantic Web of Things: http://model.webofthings.io
  - Schema.org
  - See also https://www.w3.org/WoT/IG/
### 3. Share Layer

<table>
<thead>
<tr>
<th>Layer 1: ACCESS</th>
<th>Layer 2: FIND</th>
<th>Layer 3: SHARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML</td>
<td>JSON</td>
<td>REST Crawler</td>
</tr>
<tr>
<td>Web Hooks</td>
<td>Web Thing Model</td>
<td>Link Header</td>
</tr>
<tr>
<td>URI / URL</td>
<td>REST API</td>
<td>HATEOAS</td>
</tr>
<tr>
<td>REST API</td>
<td>JSON-LD</td>
<td>RDFa</td>
</tr>
<tr>
<td>MQTT</td>
<td>HTTP</td>
<td>RDFa</td>
</tr>
<tr>
<td>CoAP</td>
<td>TLS</td>
<td>Linked Data</td>
</tr>
<tr>
<td>DTLS</td>
<td>OAuth</td>
<td>JSON-LD</td>
</tr>
<tr>
<td>JWT</td>
<td>API Tokens</td>
<td>REST Crawler</td>
</tr>
<tr>
<td>PKI</td>
<td>Social WoT</td>
<td>JSON-LD</td>
</tr>
<tr>
<td>Encryption</td>
<td>Social Networks</td>
<td>RDFa</td>
</tr>
<tr>
<td>Delegated Authentication</td>
<td>Linked Data</td>
<td>Linked Data</td>
</tr>
</tbody>
</table>

**Networked Things:**
- NFC
- 6LoWPAN
- Thread
- Ethernet
- Wi-Fi
- 3/4/5 G

**Social Networks:**
- Social WoT
- Delegated Authentication
- OAuth
- JWT
- PKI
- Encryption

**API Tokens:**
- OAuth
- JWT
- PKI

**Layer 3: SHARE**
- Social Networks
- Delegated Authentication
- API Tokens
- TLS
- DTLS

**Layer 2: FIND**
- REST Crawler
- HATEOAS
- Link Header
- Search engines
- Schema.org
- Linked Data
- Semantic Web
- mDNS

**Layer 1: ACCESS**
- HTML
- JSON
- Web Hooks
- URI / URL
- REST API
- HTTP
- MQTT
- CoAP
- NFC
- 6LoWPAN
- Thread
- Ethernet
- Wi-Fi
- 3/4/5 G
The Social Web of Things

Social Network OAuth Authorization Server

1. Login

2. Gets Token

3. GET /temp + token

Client App

devices.webofthings.io:5050

Auth Proxy

HTTPS token

Local Network

/pir Passive Infrared Sensor

192.168.1.18:8484 Web Thing

/temp Temperature Sensor
Discussion: Layer 3 - Share

- Security by obscurity never helps
  - Better off with open protocols!
- Technical challenges
  - TLS can be heavy for resource constrained devices
  - See DTLS, TLS on UDP for constrained devices
- Things on the Web = Things on the Web!
  - DDoS attacks
  - UDP flooding / TCP SYN attacks
  - Hacking the physical world
    - E.g., Shodan, Baby Monitors
4. Compose Layer

Layer 4: COMPOSE
- Systems Integration
  - Networked Things
  - Bluetooth
  - ZigBee
  - QR
  - NFC
  - Beacons
- 6LoWPAN
- Thread
- Wi-Fi
- Ethereum
- 3/4/5G
- JSON
- HTML
- HTTP
- URI / URL
- REST API
- HTTP
- CoAP
- JSON-LD
- RDFa
- Linked Data
- Semantic Web
- Search engines
- REST Crawler
- HATEOAS
- Link Header
- Social WoT
- OAuth
- API Tokens
- TLS
- DTLS
- JWT
- PKI
- Encryption
- Social Networks
- Delegated Authentication
- OAuth
- Node-RED
- IFTTT
- REST API
- Automation
- UI Generation
- Web Applications
- WoT-a-Mashup
- Physical Mashups
- mDNS

Layer 3: SHARE
- OAuth
- Delegated Authentication
- API Tokens
- Social Networks
- TLS
- DTLS
- Encryption
- JWT
- PKI
- RDFa
- Linked Data
- Semantic Web
- Search engines
- REST Crawler
- HATEOAS
- Link Header
- Social WoT
- OAuth
- API Tokens
- TLS
- DTLS
- Encryption
- Social Networks
- Delegated Authentication
- OAuth
- Node-RED
- IFTTT
- REST API
- Automation
- UI Generation
- Web Applications
- WoT-a-Mashup
- Physical Mashups
- mDNS

Layer 2: FIND
- REST Crawler
- HATEOAS
- Link Header
- Social WoT
- OAuth
- API Tokens
- TLS
- DTLS
- Encryption
- Social Networks
- Delegated Authentication
- OAuth
- Node-RED
- IFTTT
- REST API
- Automation
- UI Generation
- Web Applications
- WoT-a-Mashup
- Physical Mashups
- mDNS

Layer 1: ACCESS
- HTML
- JSON
- Web Hooks
- Gateway
- REST API
- HTTP
- CoAP
- JSON-LD
- RDFa
- Linked Data
- Semantic Web
- Search engines
- REST Crawler
- HATEOAS
- Link Header
- Social WoT
- OAuth
- API Tokens
- TLS
- DTLS
- Encryption
- Social Networks
- Delegated Authentication
- OAuth
- Node-RED
- IFTTT
- REST API
- Automation
- UI Generation
- Web Applications
- WoT-a-Mashup
- Physical Mashups
- mDNS

Networked Things
- NFC
- QR
- 6LoWPAN
- Bluetooth
- Thread
- Ethernet
- Wi-Fi
- Scan
- Thread
- 3/4/5G
Composing the real-world: Physical Mashups

http://node-red.org
Physical Mashups with IFTTT

Recipe Title
If new tweet by specific user @wotbook, then make a web request

http://ifttt.com
The Reactor

- **Usage:** reacting to events
  - Generate alerts for users via apps, SMS, calls
  - Generate events in other systems
  - Physical mashups

- **Massively scalable scripting run-time**
  - Unlimited horizontal scale
  - Ephemeral virtual instances

- **Secure**
  - Data isolation
  - One event, one instance

- **Flexible**
  - Node.js runtime
  - Full NPM access

https://developers.evrythng.com/
20% off “Building the Web of Things” with code “guid20evry”
See: http://book.webofthings.io

Free e-book with sample chapters