INTERNET OF THINGS – INTERNET IN THE SMALL

H. Schulzrinne
(+ Jan Janak & other CUCS IRT contributors)

COMSNETS 2016

Overview

- The Internet of Things is many things
  - Most of them aren't new or exciting
- The network part isn't (all that) exciting
  - but usable security, software and system integration are
- It's a software thing → multi-devices services, APIs, IFTTT, SECE, …
- About being a nice IoT citizen
- ShellShock for light switches?

Internet of Things (IoT)

- Also sometimes called
  - "industrial Internet" (subset)
  - "machine-to-machine" (M2M)
  - "cyber-physical systems" (NSF)
  - "smart city" (IBM)
- Concept dates to late 90's
  - Initially, tag physical objects (RFID)
  - Now, **networked** sensors and control of previously mechanical or electrical objects

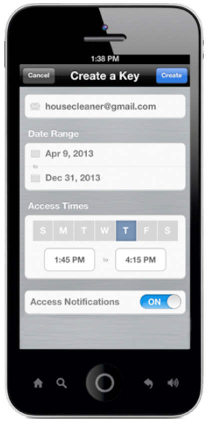
M2M/IoT/CPS is not…

- isn’t just about fancy thermostats and $199 door bells
- doesn’t always uses cellular networks
- is not always energy-constrained
- is not always cost-constrained
- doesn’t always use puny microcontrollers
- is not always run by large organizations
  - many small & mid-sized providers
  - usually embedded into other products

Internet of Things

- Mostly about devices, not the Internet
- Network part not really new or exciting
- Software-controlled networked devices
- Challenges:
  - lack of UI → usability
  - lack of UI → usable security
  - integration (service & APIs)
  - programming – beyond a single device

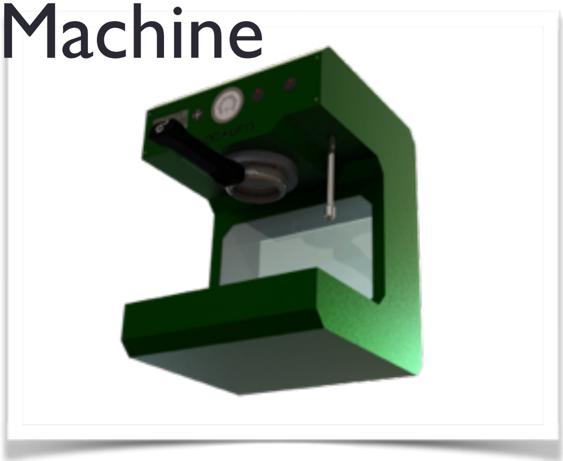
Natural evolution



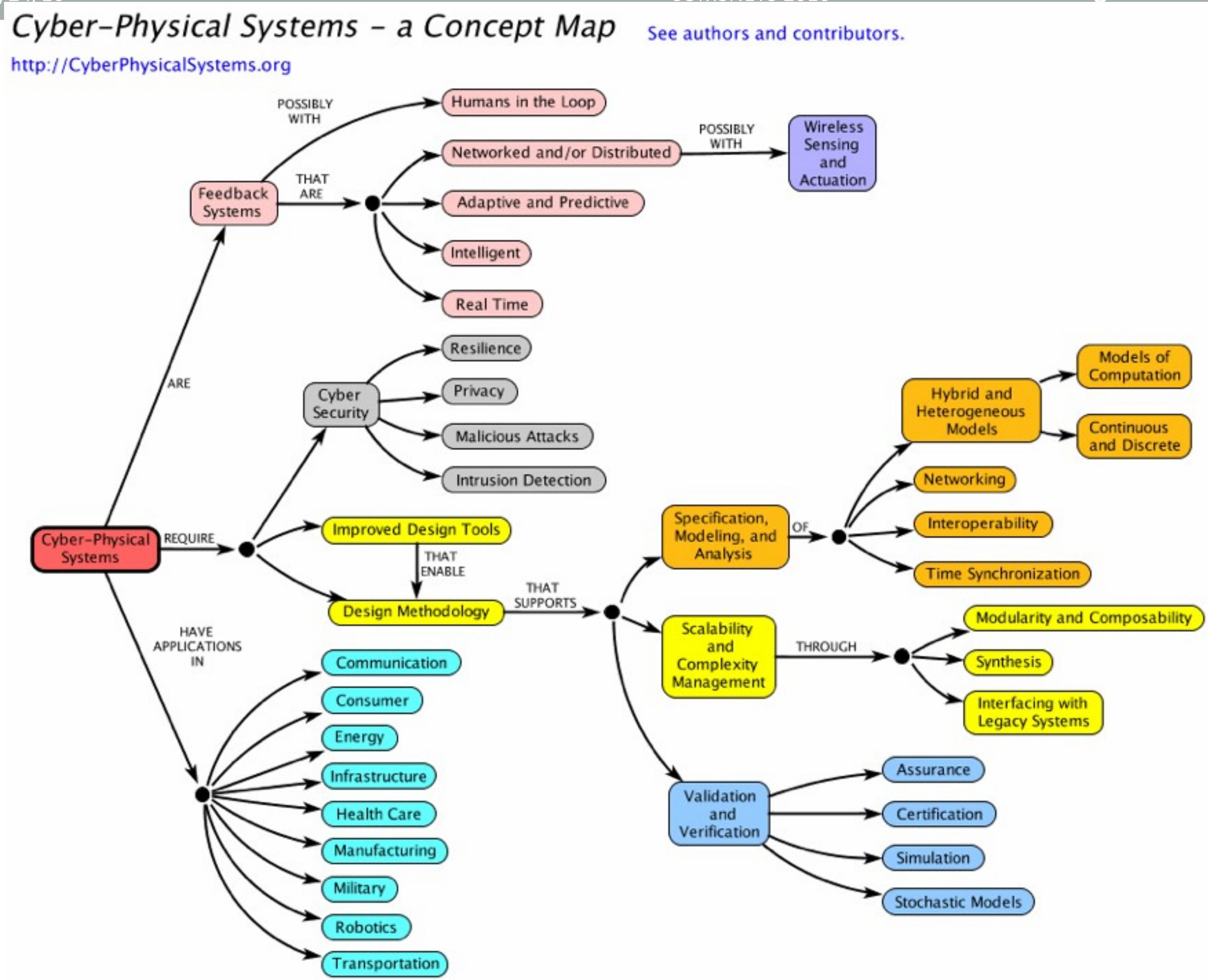
Software Defined Devices

Sous vide cooking

PID Controlled Espresso Machine



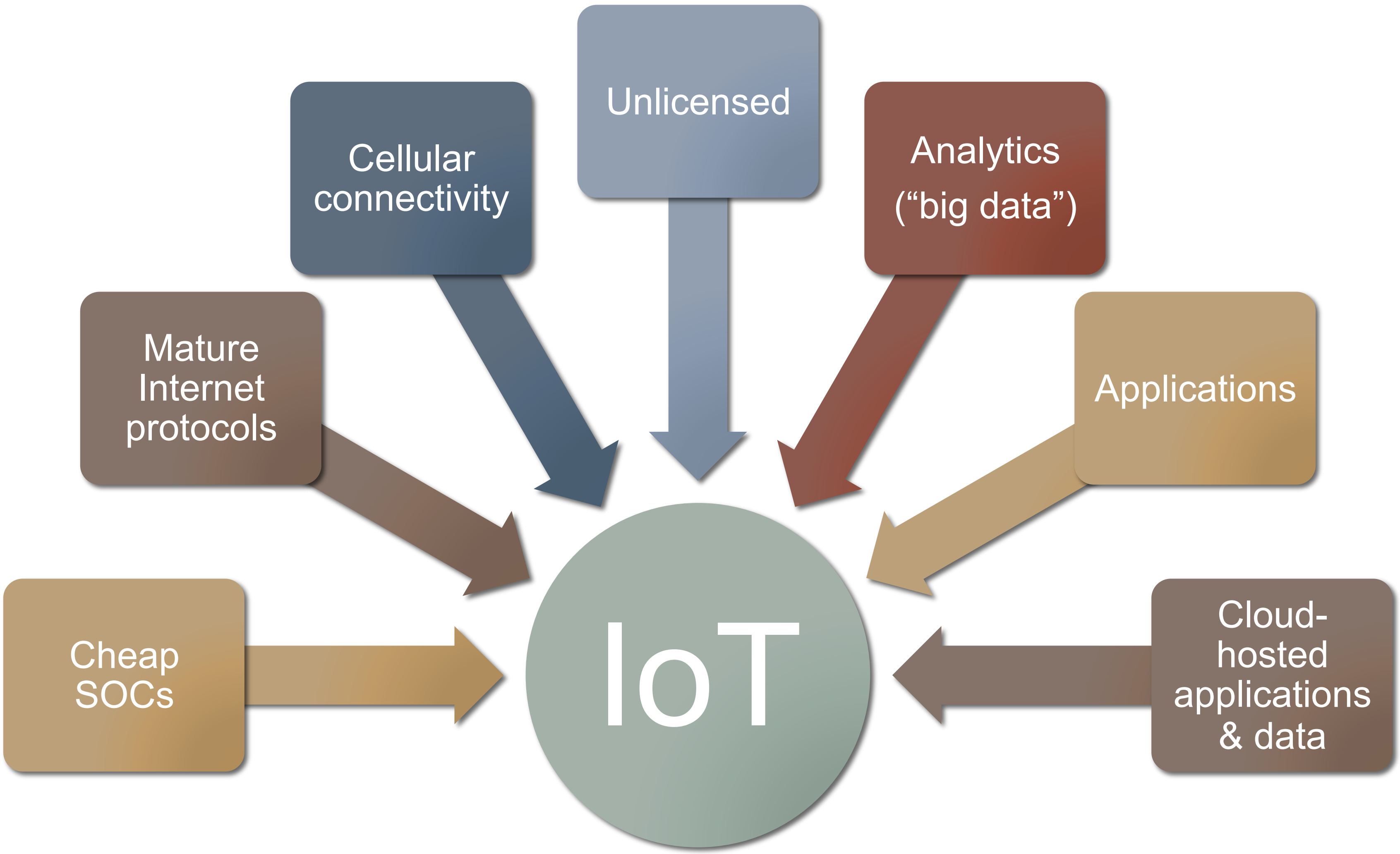
Picobrew


Cyber-Physical Systems - a Concept Map
See authors and contributors.
http://CyberPhysicalSystems.org
Cyber-Physical Systems
ARE
Feedback Systems
POSSIBLY WITH
Humans in the Loop
THAT ARE
Networked and/or Distributed
POSSIBLY WITH
Wireless Sensing and Actuation
Adaptive and Predictive
Intelligent
Real Time
REQUIRE
Cyber Security
Resilience
Privacy
Malicious Attacks
Intrusion Detection
Improved Design Tools
THAT ENABLE
Design Methodology
THAT SUPPORTS
Specification, Modeling, and Analysis
OF
Hybrid and Heterogeneous Models
Models of Computation
Continuous and Discrete
Networking
Interoperability
Time Synchronization
Scalability and Complexity Management
THROUGH
Modularity and Composability
Synthesis
Interfacing with Legacy Systems
Validation and Verification
Assurance
Certification
Simulation
Stochastic Models
HAVE APPLICATIONS IN
Communication
Consumer
Energy
Infrastructure
Health Care
Manufacturing
Military
Robotics
Transportation

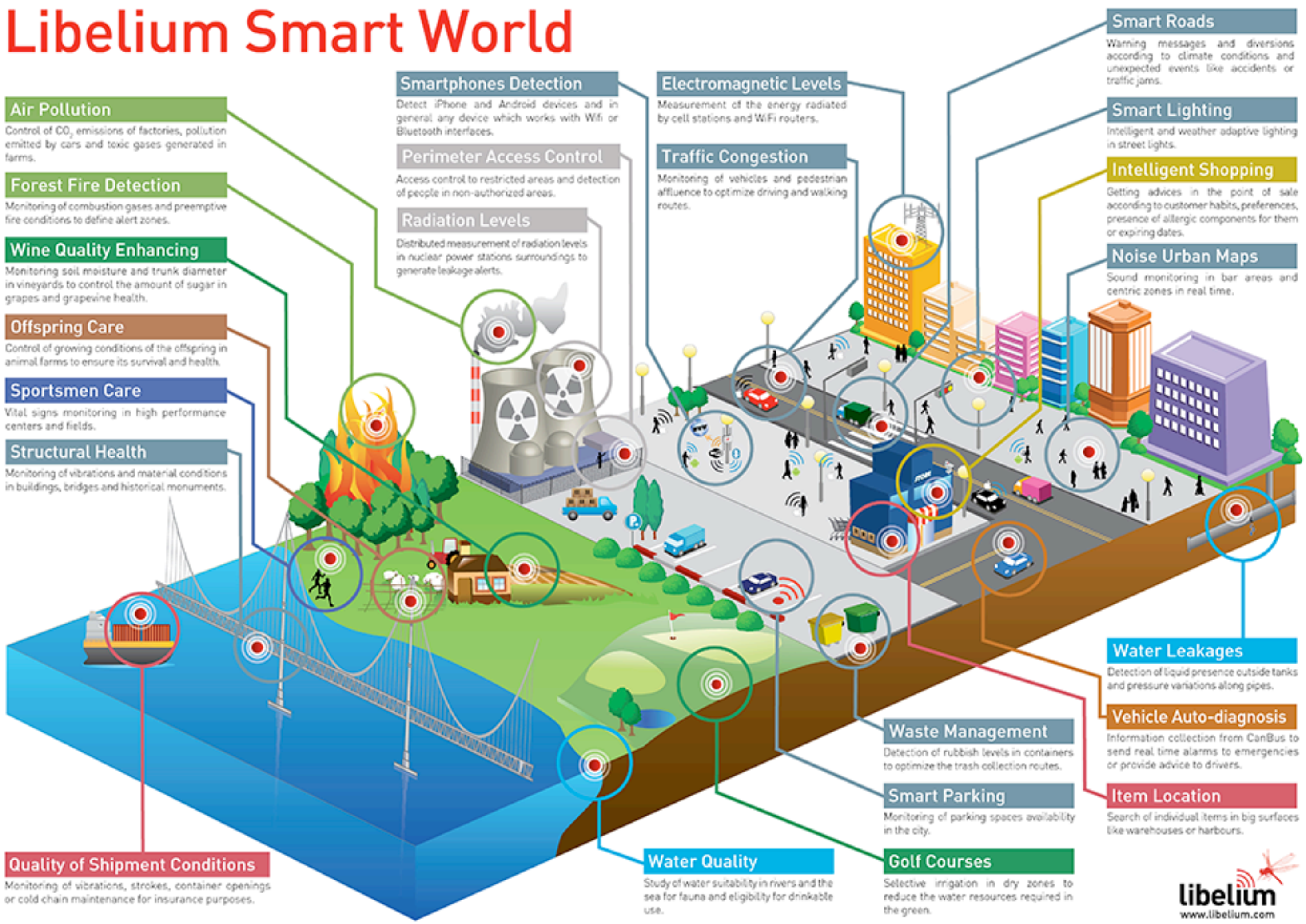
CPS, WSN & IoT challenges

	Node	Network	Program	Example
IoT	re-usability	interoperability	heterogeneous & loosely coupled	VoIP thermostat
CPS	real-time guarantees	predictability & redundancy	model verification	avionics industrial control
WSN	energy efficiency	minimize communication	homogeneous minimal OS	structural monitoring (bridge)

Key enablers



Environmental sensing

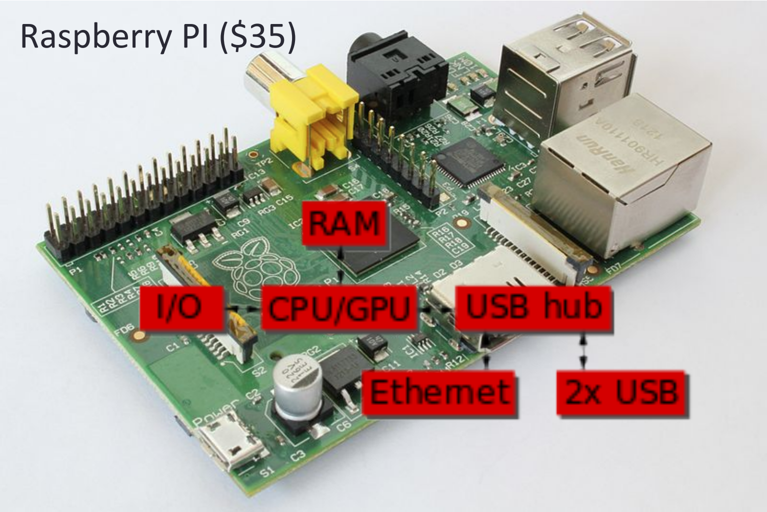


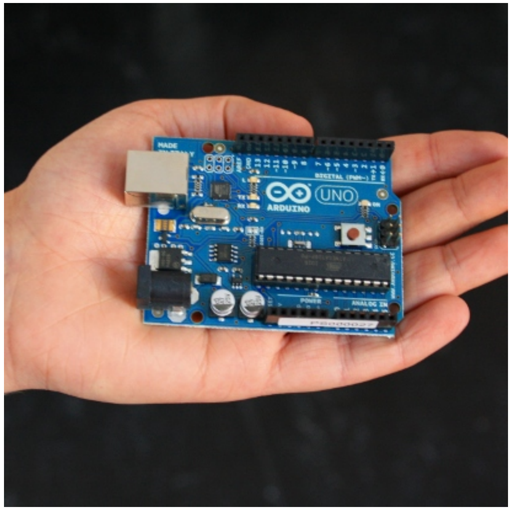
http://www.libelium.com/top_50_iot_sensor_applications_ranking/#show_infographic

The killer app

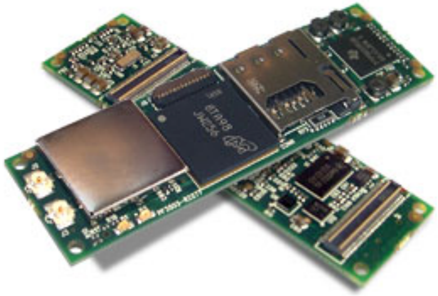
with energy-harvesting

IoT = cheap microcontrollers + network interfaces





Arduino Uno, €20



Gumstix (WiFi, BT): 58 mm, $199

Where does IoT make sense?

- Automate manual data extraction
  - health, car, electric/gas meter, …
- Remote maintenance
  - vending machines, appliances, cars & trucks, trains, pumps, …
- Incorporate additional information
  - thermostats, light switches, traffic lights, parking meters, …
- Software-Defined Mechanics
  - locks, light switches
- But where does it solve more than 1st world problems?
  - commercial maintenance savings?
  - in-home customizable assistive technology

Connectivity Framework

Three dominant classes of wireless IoT links (there are others)

1. **Thing to Thing** (vehicles, sensors/actuators, etc.)

   LAN/PAN range; use spectrum suited to short distances; extensive spatial reuse

2. **Thing to Proxy** (e.g., gateways, hubs, hubs within vehicles, etc.)

   LAN/PAN* range; use spectrum suited to short distances; extensive spatial reuse

   – IoT adds significant load to existing services, such as WiFi and BT
   – Traffic upstream from proxies shares allocations and adds significant load to existing services used to link WiFi, etc. to core Internet.

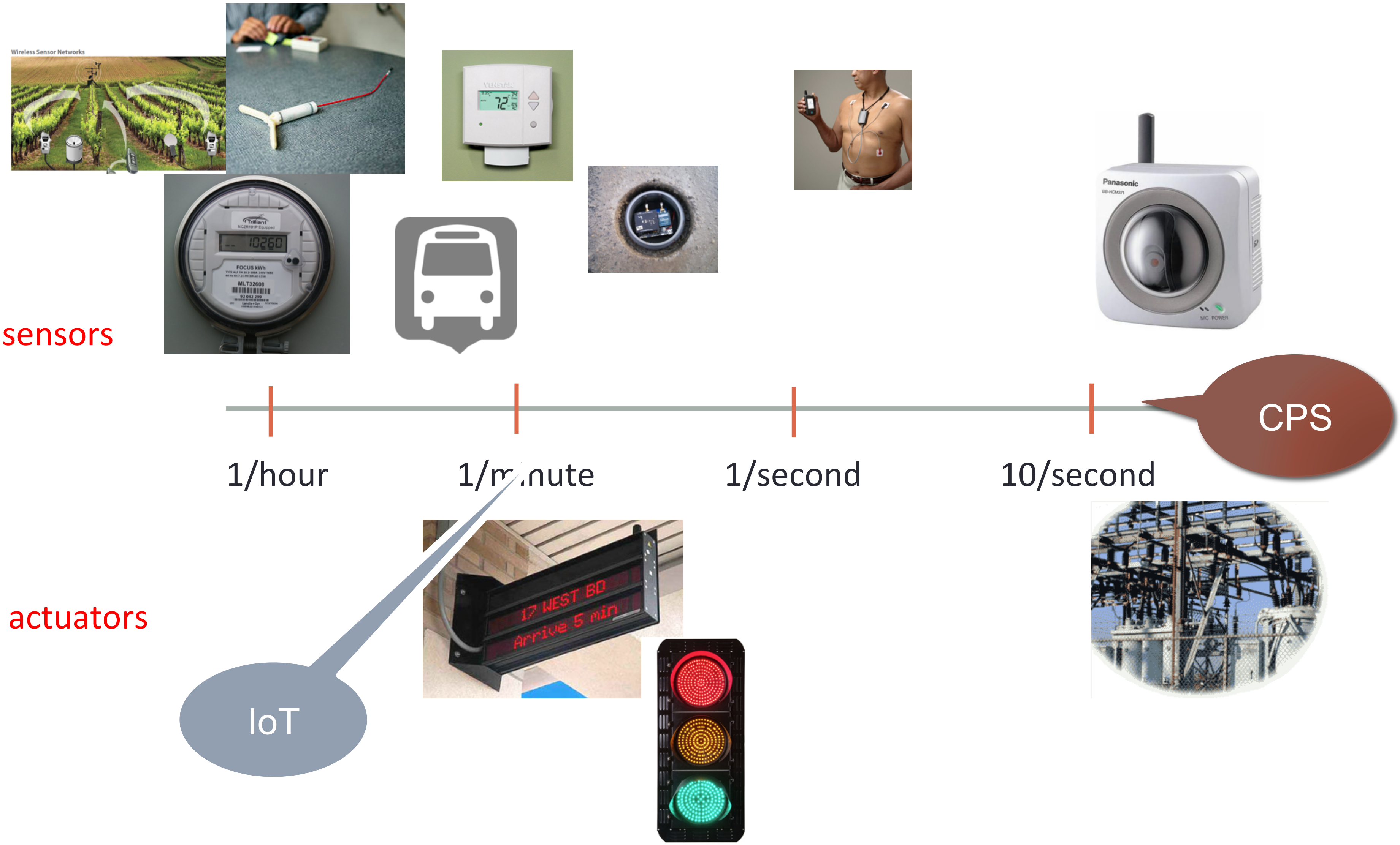
3. **Thing to Internet** (e.g., direct connection to 4G networks, WISPs, TVWS, etc.)

   last mile range; share spectrum with and/or use other wide area services

   – IoT adds load to 4G/TVWS services and poses challenges wrt long-lived things

* Personal Area Network -- typically operates within a range < 10M

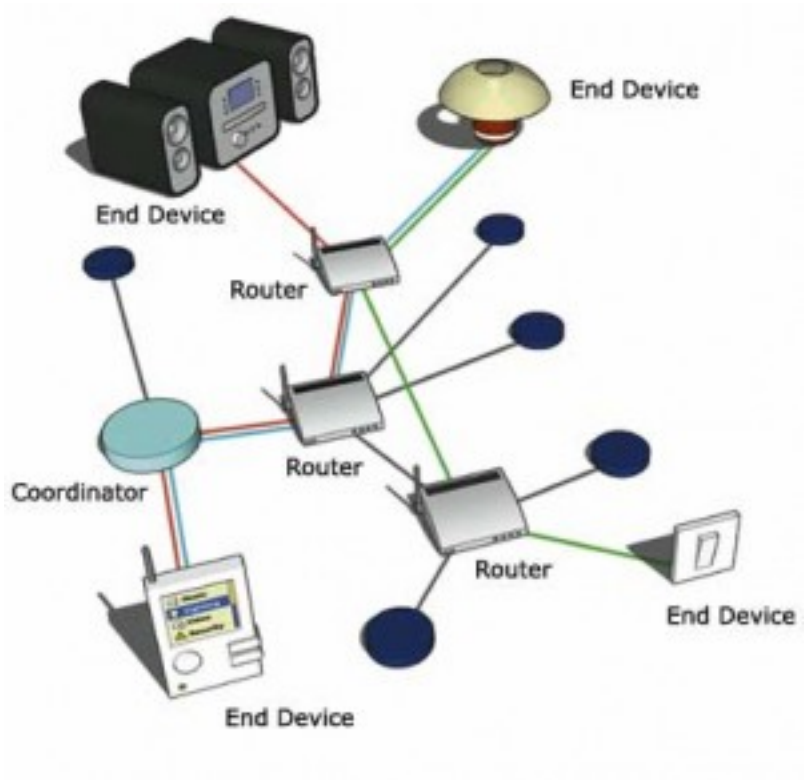
IoT varies in communication needs



4G/5G and IoT scenarios

	LPWANs	(2G) 4G/5G licensed	5G Wi-Fi	WLAN, WPAN
indoor, campus	?	3rd party systems (utility metering)		home automation; machine monitoring
vehicle	tracking	V2I	?	in-vehicle, DSRC
outdoors	agriculture, traffic control, infrastructure monitoring; eHealth	surveillance video? eHealth?		BT-LE connection to smartphone

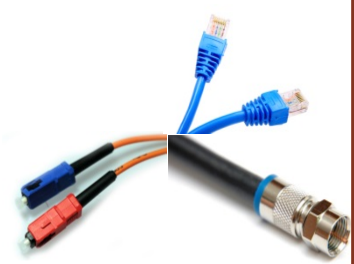
Not just cellular *or* unlicensed



IoT Connectivity Technologies

Wired

- Ethernet, Coax, Fiber, etc. considered as a single category



WPAN

- ANT+
- Bluetooth – Classic & Smart Ready
- Bluetooth Smart

- ZigBee PRO
- ZigBee RF4CE
- ZigBee Multi-Protocol
- EnOcean
- ISA100.11a
- WirelessHART
- Z-Wave
- Other 802.15.4

WLAN

- 802.11a/b/g
- 802.11n
- 802.11ac
- 802.11ad
- Other 802.11
- DECT ULE
- Other 2.4GHz
- Other Sub-GHz

WWAN

- 2G Cellular
- 3G Cellular
- 4G Cellular

Bill Morelli, IHS Technologies

Niche networks

short range

ZigBee®

low energy; mesh

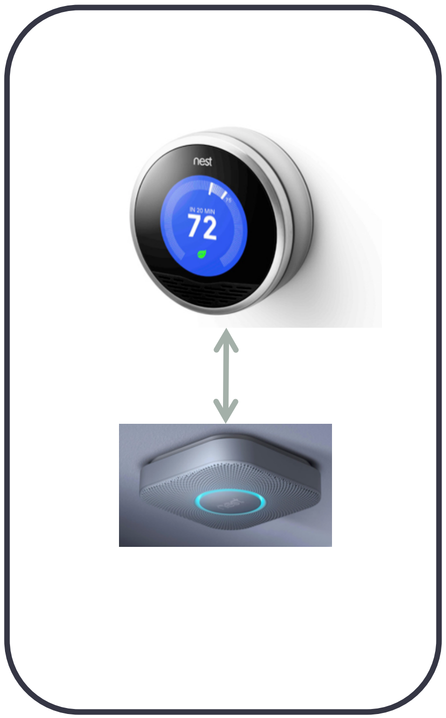
Bluetooth®

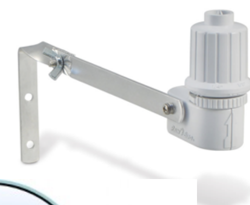
ubiquity; low cost

Wi Fi™

speed; public APs

IoT islands vs. IoT eco system





IoT: more than programmable light bulbs

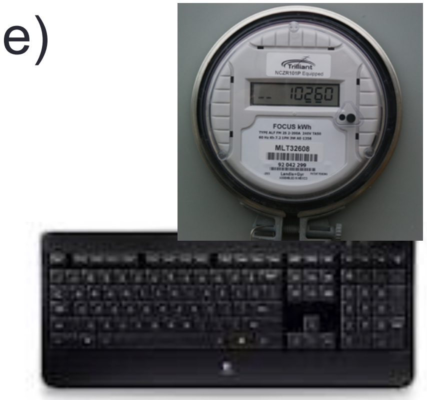
public sensors & actuators

semi-private

private

Network challenges

- Unlicensed
  - How do I attach and authenticate a device to a (home) network?
  - Credentials?
- Licensed
  - Reliability → multiple *simultaneous* providers
  - Mobility → different providers in different regions
  - Charging → often low, intermittent usage, sometimes deferrable (“Whispernet”)
    - From \$50/device/month → < \$1/month?
- Authentication
  - Which devices can be used by whom and how?
    - “Any employee can monitor the room temperature in any public space, but only Facilities staff can change it”

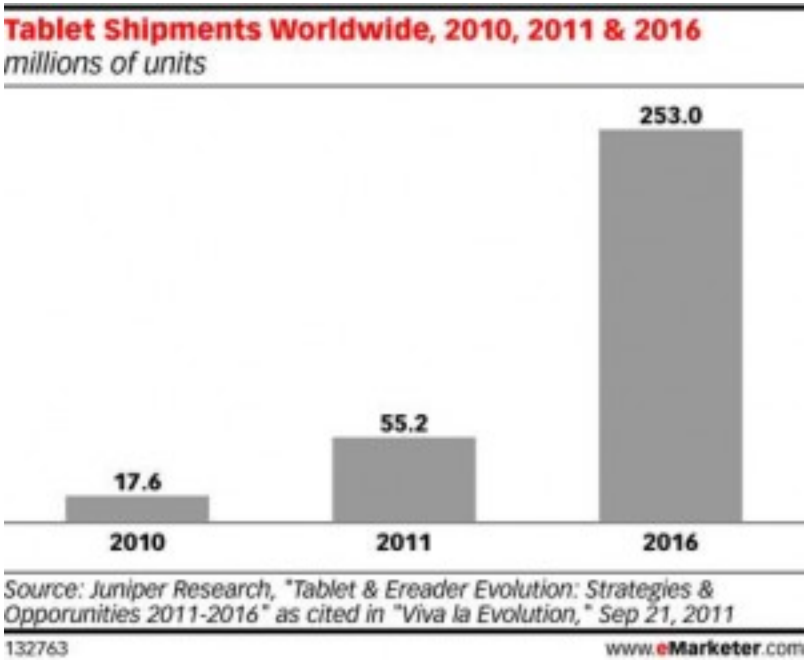


Phone numbers for machines?

< 2010

212 555 1212 →

254 mio.



500 123 4567
(and geographic numbers)

OnStar by GM

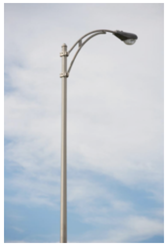
500 123 4567
533, 544 →

12% of adults

5 mio.

311,000

64 mio.



44.9 mio.

now: one 5XX code a year…
(8M numbers)

10 billion +1 #'s available

see Tom McGarry, Neustar

Communication identifiers

Property	URL owned	URL provider	E.164	Service-specific
Example	alice@smith.name sip:alice@smith.name	alice@gmail.com sip:alice@ilec.com	+1 202 555 1010	www.facebook.com/alice.example
Protocol-independent	no	no	yes	yes
Multimedia	yes	yes	maybe (VRS)	maybe
Portable	yes	no	somewhat	no
Groups	yes	yes	bridge number	not generally
Trademark issues	yes	unlikely	unlikely	possible
Privacy	Depends on name chosen (pseudonym)	Depends on naming scheme	mostly	Depends on provider "real name" policy

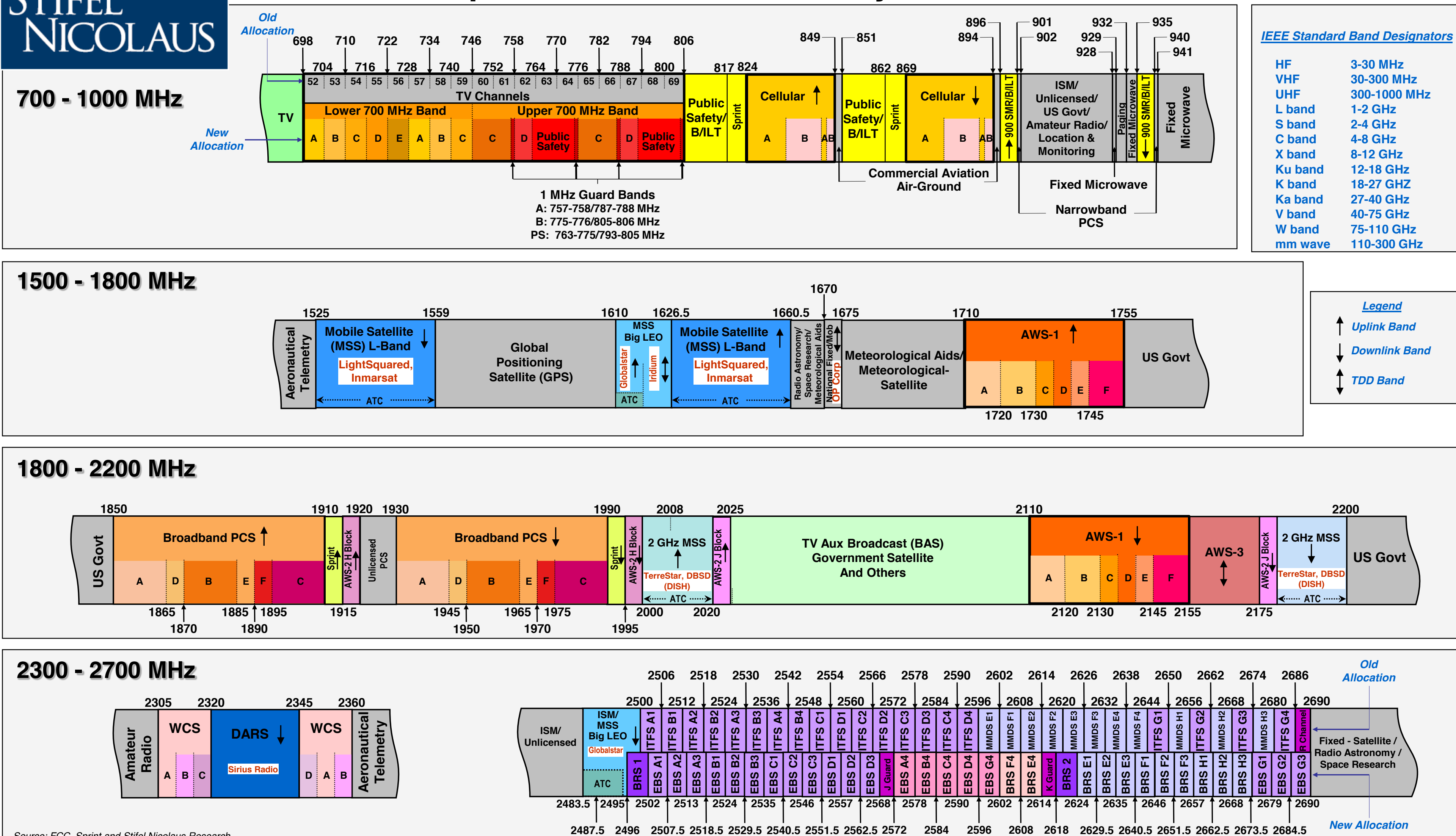
→ IoT will likely be assigned local IP address space and owner-based names (meter17.pseg.com) [if any]

The age of application-specific {sensors, spectrum, OS, protocol …} is over

- *Computing system*: dedicated function → OS
  - → abstract into generic components
  - e.g., USB human interface device (HID)
- What are the equivalent sensor and actuator classes?
- *Networks*: generic app protocols
  - request/response → HTTP
  - event notification → SMTP, SIP, XMPP
- *Spectrum*: from new application = new spectrum to generic data transport

U.S. Spectrum Allocation of Key Bands

July 14, 2011



Source: FCC, Sprint and Stifel Nicolaus Research

Post-spectrum world

Old (pre-2000)

- Mostly single-use, application-specific allocations (“radar”, “LMR”, “paging”)
- Mostly federal OR non-federal use
- Each band its own world
- Static usage
- Limited spectral efficiency concerns
- Go west (up), young application!

Now

- No more unallocated bands (below 30+ GHz) → multi-use, generic transport
- Shared federal & non-federal use
- Neighbor “issues” (GPS, TV)
- Usage may change (satellite → mobile)
- Spectral efficiency – but how measured? (bits/s/Hz/km^2?)
- Limited ability to go to higher frequencies

Current unlicensed spectrum

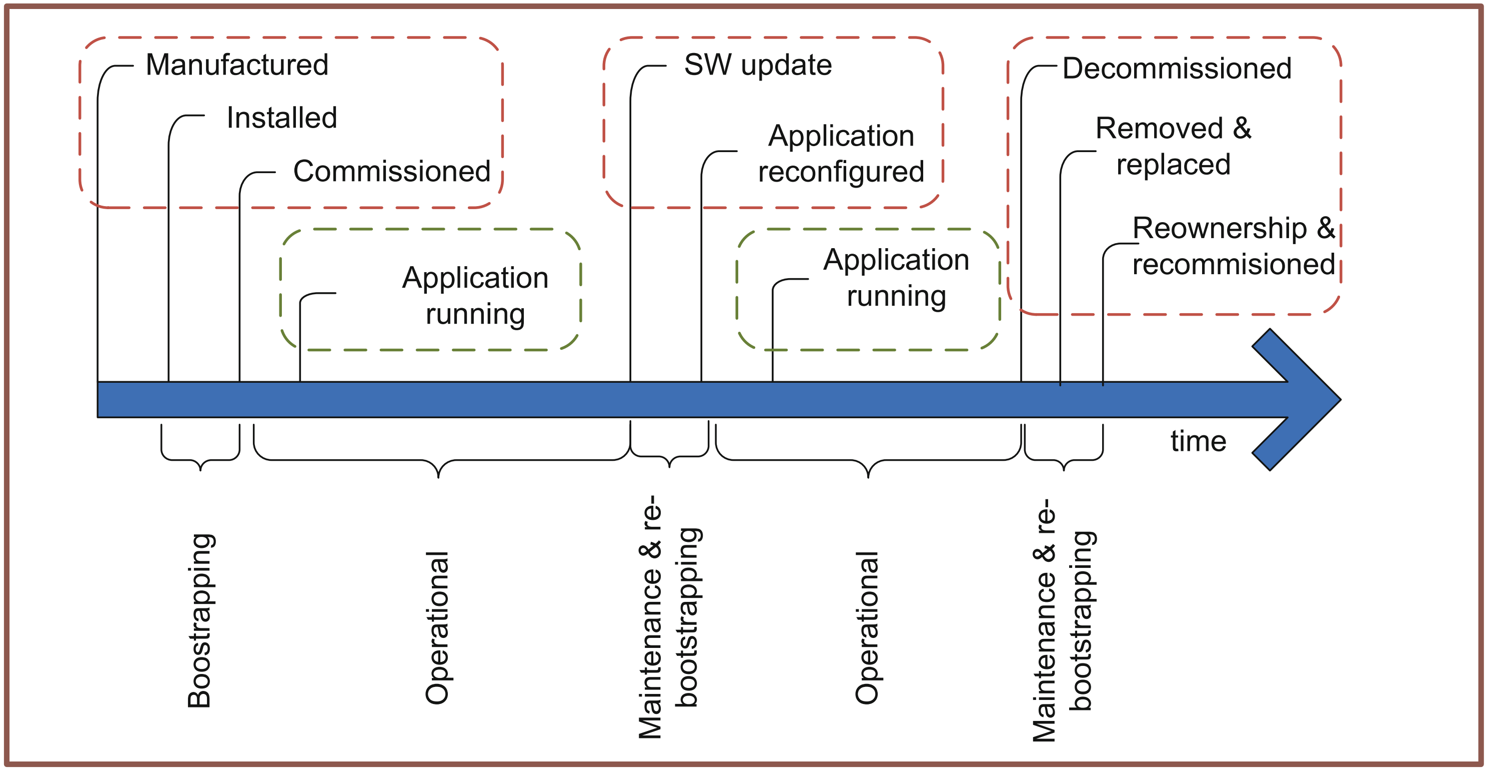
Frequency range		Bandwidth	Center frequency	Availability
6.765 MHz	6.795 MHz	30 KHz	6.780 MHz	Subject to local acceptance
13.553 MHz	13.567 MHz	14 KHz	13.560 MHz	
26.957 MHz	27.283 MHz	326 KHz	27.120 MHz	
40.660 MHz	40.700 MHz	40 KHz	40.680 MHz	
433.050 MHz	434.790 MHz	1.84 MHz	433.920 MHz	Region 1 only and subject to local acceptance
902.000 MHz	928.000 MHz	26 MHz	915.000 MHz	Region 2 only
2.400 GHz	2.500 GHz	100 MHz	2.450 GHz	
5.725 GHz	5.875 GHz	150 MHz	5.800 GHz	
24.000 GHz	24.250 GHz	250 MHz	24.125 GHz	
61.000 GHz	61.500 GHz	500 MHz	61.250 GHz	Subject to local acceptance
122.000 GHz	123.000 GHz	1 GHz	122.500 GHz	Subject to local acceptance
244.000 GHz	246.000 GHz	2 GHz	245.000 GHz	Subject to local acceptance

+ TV white spaces (in 476-692 MHz range) – availability varies

Challenge: enabling discovery & access control

- Devices should be discoverable & re-usable
  - e.g., provide audio interface to bus display
  - environmental probes (temperature, noise, rain, …)
  - location (iBeacon) → 911
- Layers of functionality
  - anybody in vicinity can read
  - anyone in *family* can change
  - parents can re-program
- Allow delegation
  - grant temporary access to somebody or something else
  - by message or physical proximity
- Currently, all one-off solutions
  - OAuth? NFC?

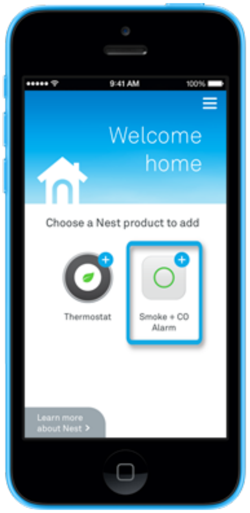
Device lifecycle



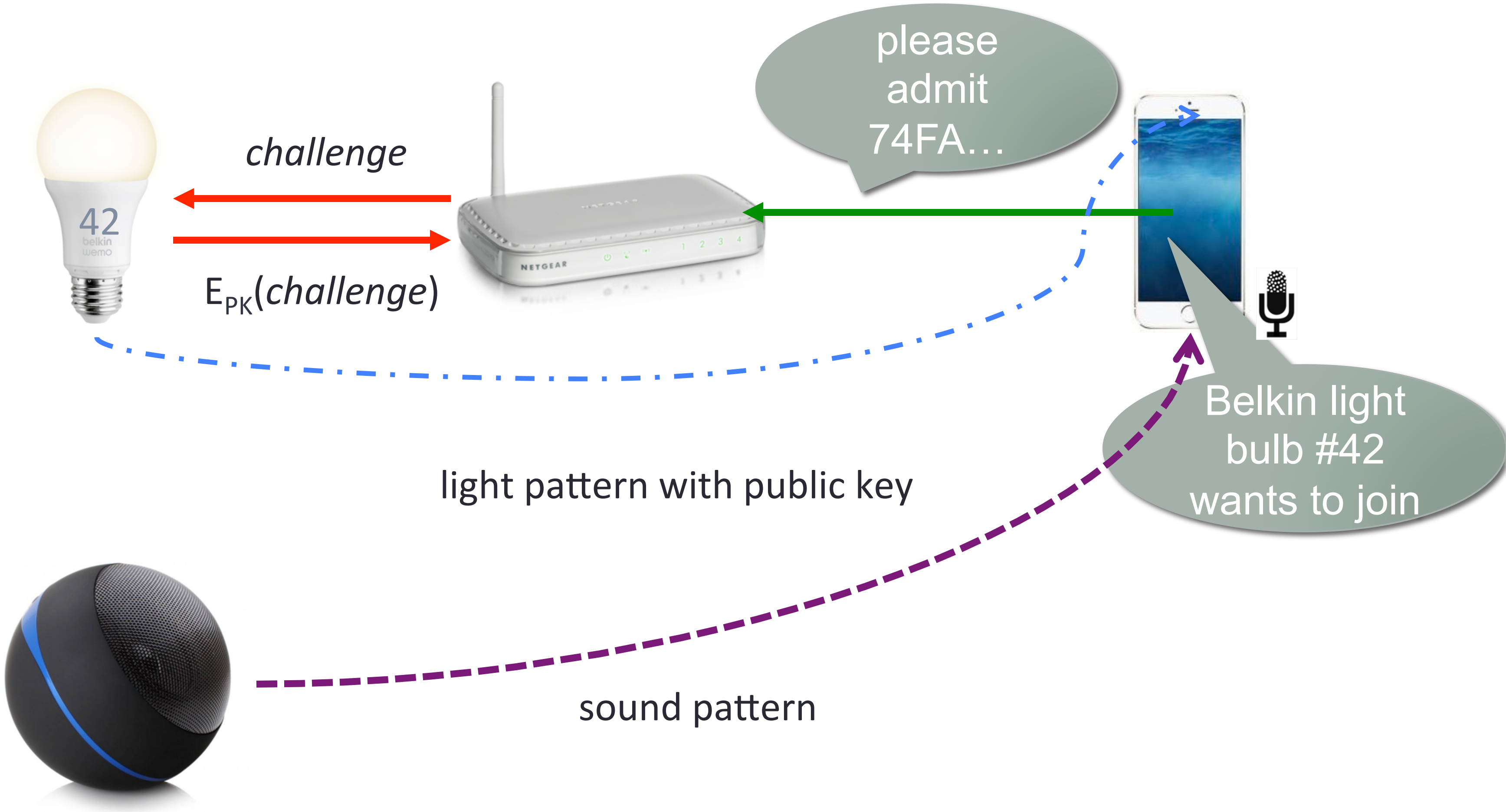
T. Heer et al., "Security Challenges in the IP-based Internet of Things", 2011.

Challenge: enrollment

- Commercial buildings → enroll 1,000s of devices at once
- Home → enroll one device at a time
  - current model: one app per device (class)
  - re-do if Wi-Fi password changes
  - common options:
    - QR code
    - P2P Wi-Fi (Wi-Fi Direct)
  - possibilities
    - "hi, I'm a Philips light bulb – add me!" (PKI)



Other introduction models



Avoid single password → allow device transfer

IoT good-citizen rules

FCC TAC recommendations +

- Implement current best practices
  - no plain-text data or commands
    - low-power CPUs are no excuse – long-payback or infrequent crypto operations
  - no default passwords
- Do not assume that your (cellular) network is around in > 8 years
  - short-range unlicensed bands more likely a safe harbor
- Update yourself securely
- Don't trust random APs → PassPoint, 802.1x?
  - matters mainly for DNS and denial-of-service
- Go into fail-safe mode if no updates
- Be nice to cellular network (signaling, white spaces, …)
  - and maybe "kill switch" if misbehaving (or stolen!)
- Don't ask for special spectrum
  - except maybe if you're a health-and-safety device (but share nicely)
  - or maybe low-bandwidth narrowband spectrum

Example: SIGFOX (902 MHz, 100 bps) is a connectivity solution that focuses on low throughput devices. On SIGFOX you can send between 0 and 140 messages per day and each message can be up to 12 bytes of actual payload data.

ShellShock for light switches

```
bash
$ env x='() { :;}; echo vulnerable'
```

- IoT risks: privacy, DDOS, extortion, …
- *Securely* field updateable or no connection to Internet
  - still vulnerable if malware on home network
- Lifetime of devices > lifetime of company
- Insurance model:
  - source code escrow + maintenance for N years
- UL listing

Theses: Internet lessons

- The Internet is about more than the Internet protocol
- Reliability multiplies, costs add
- Quality is no substitute for quantity
- Data links layers come & go, IP stays
- The age of application-specific {sensors, spectrum, OS, protocol …} is over
- Protocols matter, but programmability matters more

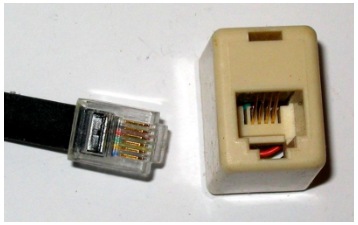
Technology changes, interfaces are forever

110/220V

1904

1901

1993



1878

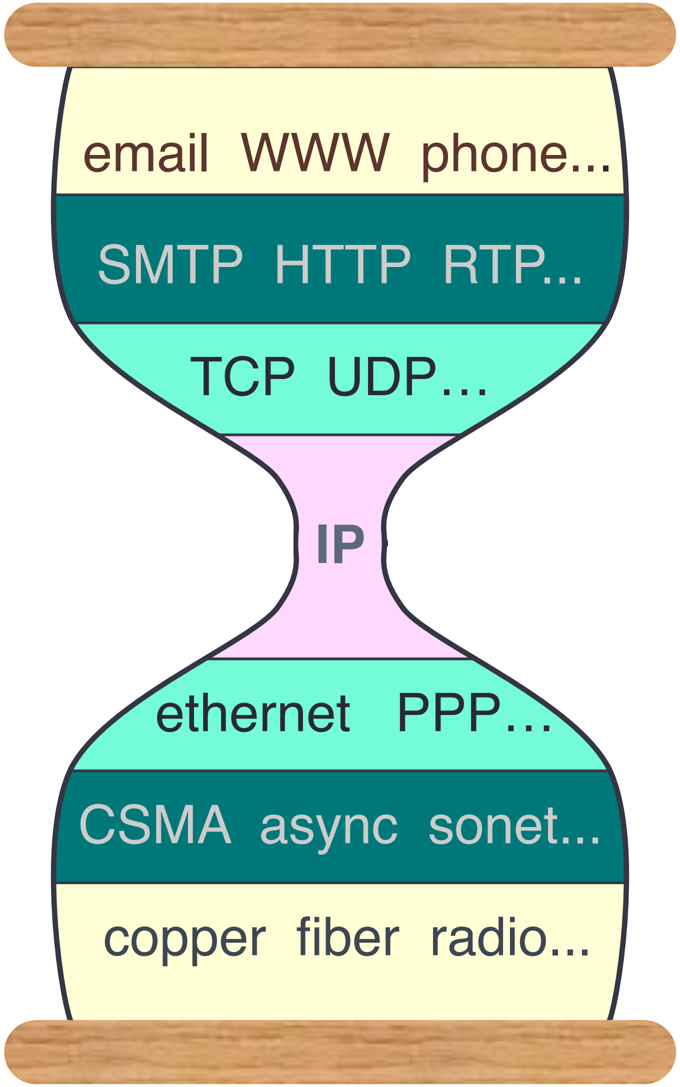
1992

1982

1830

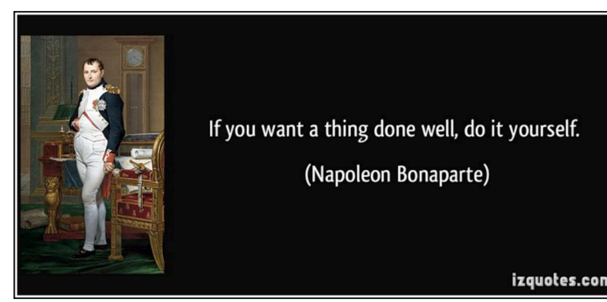
The Internet is about more than the Internet protocol

- Protocol hour glass
- But ignores important (late) lessons:
  - naming matters: no consumer Internet without DNS
  - support auto-configuration: DHCP, DNS, netconf
  - network management & diagnostics: ICMP, traceroute, SNMP
- Common data coding layers:
  - initially ASN.1
  - then XML
  - now JSON



Reliability multiplies, cost adds

- Pre-Internet (e.g., phone system):
  - Single switch, 5-nines reliability, $3M dollars
  - mostly hardware
  - specialty components
- Internet:
  - assume that components fail → compensate by redundancy
    - cost for high-reliability components increases >> linear
    - e.g., two paths with 99% reliability each
    - → generic transport (e.g., multiple 4G, fiber, DSL, cable, satellite)
  - build control loops and protocols that expect failure
    - retry, graceful degradation, adaptation, …
    - end-to-end argument
  - as much software as possible → *software-defined networks & infrastructure, IoT*
    - late bindings in functionality



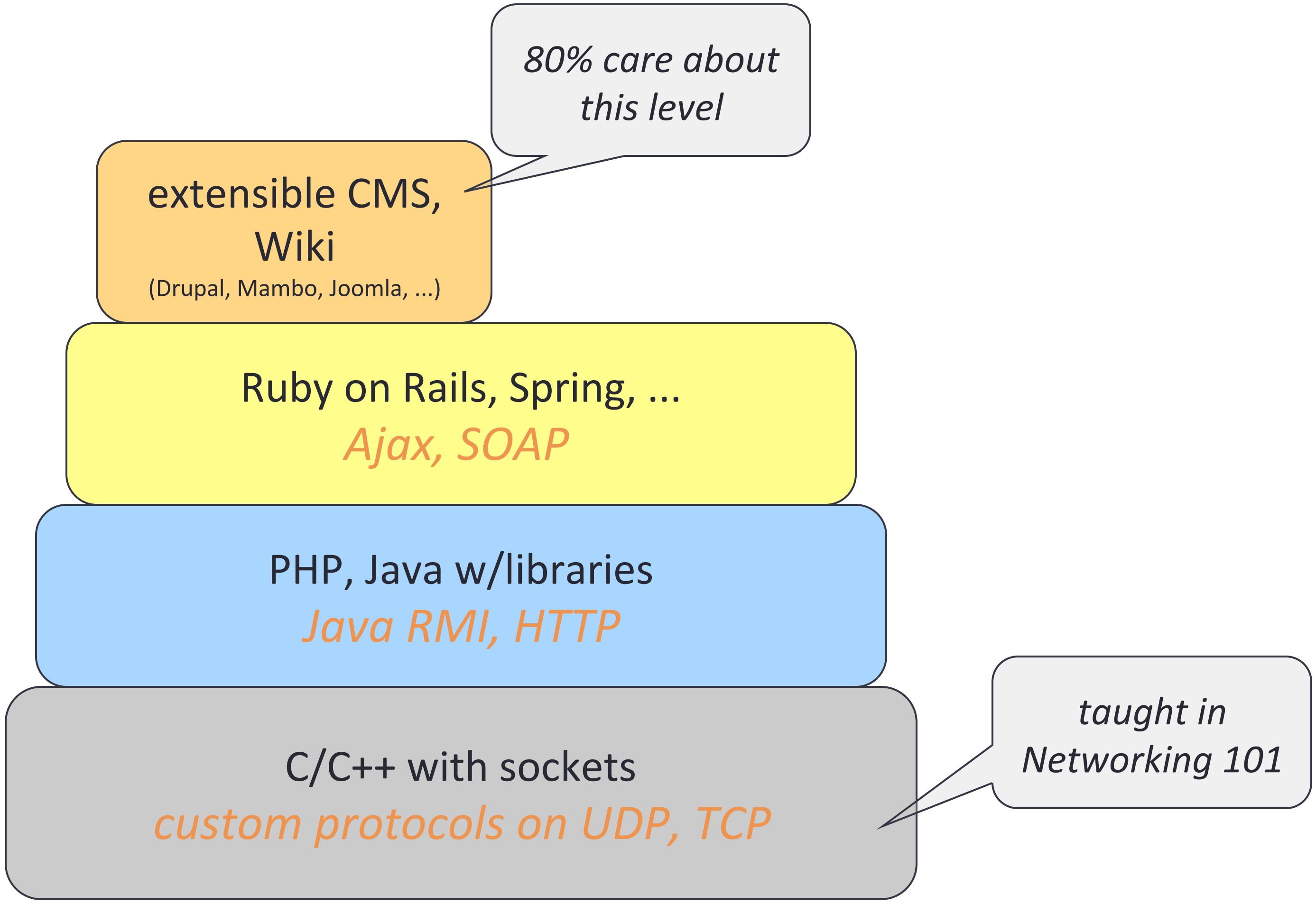
Quality is no substitute for quantity

- Hypothesis: every new networking area breeds QoS research
- Outcome so far:
  - QoS helps only when the network is *modestly* overloaded
    - but then YouTube doesn't work, either
  - downed trees don't respect reserved resources
  - adds complexity and security vulnerabilities
  - simple priority (e.g., for latency-sensitive applications)
  - → avoid *buffer bloat*
  - shared networks are better → faster links → low queuing delays

Protocols matter, but programmability matters more

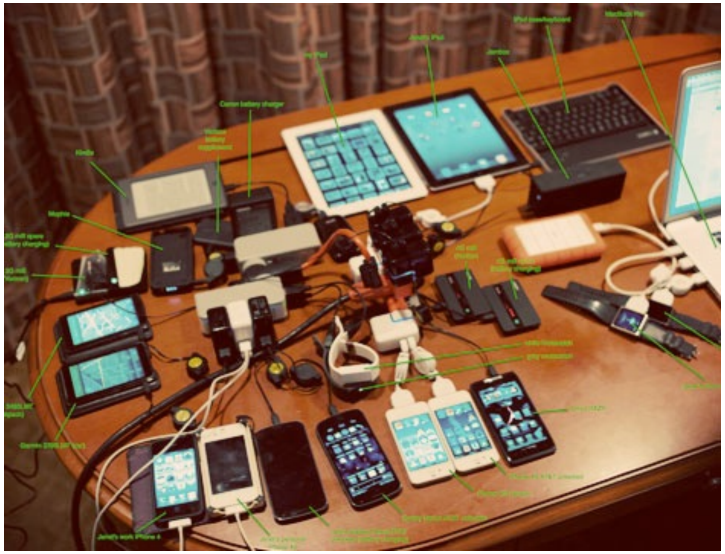
- Nobody wants to program raw protocols
- Most significant network application creation advances:
  - 1983: socket API → abstract data stream or datagram
  - 1998: Java network API → mostly names, HTTP, threads
  - 1998: PHP → network input as script variables
  - 2005: Ruby on Rails → simplify common patterns
- Many fine protocols and frameworks failed the programmer hate test
  - e.g., JAIN for VoIP, SOAP for RPC

Building Internet applications



The IoT integration problem

- Too many online services to use
- Too many devices to interact with
- Online and physical worlds disconnected
- M2M communication with a human in the middle
- → SECE (Sense Everything, Control Everything)



SECE

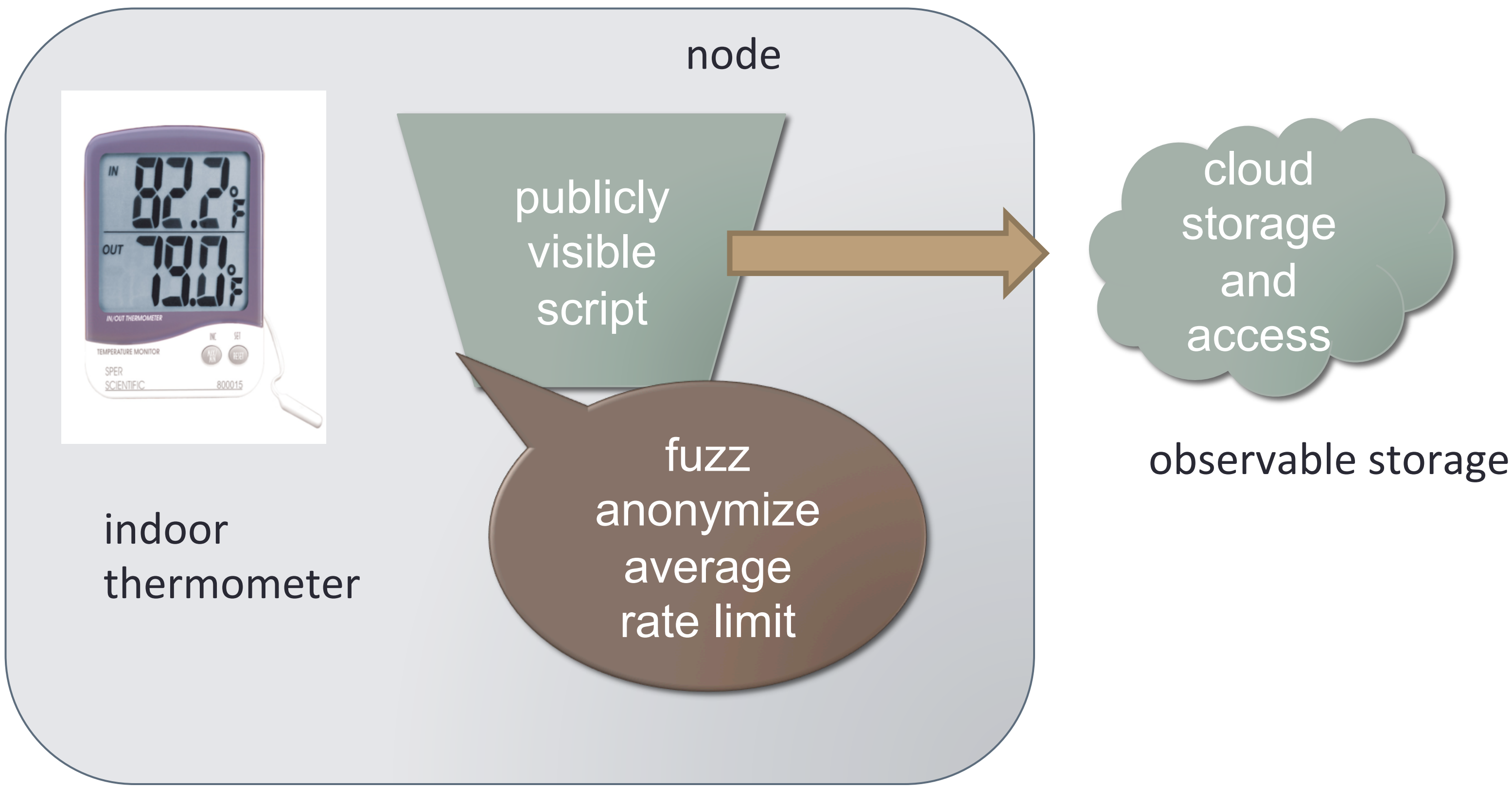
Sense Everything, Control Everything

Goal: An operating system for IoT

- OS = **abstract devices & networks**
- virtual devices act like physical devices
- program stays the same even as physical properties change
- abstract naming: by location, property, …
- manage shared access: permissions, logging, updates, …
- not a node OS

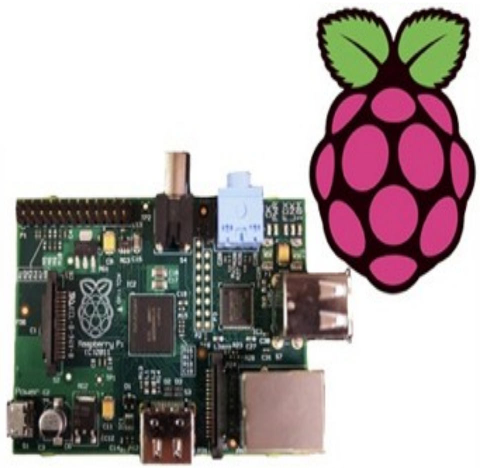
= one logical abstractio

Local processing for privacy



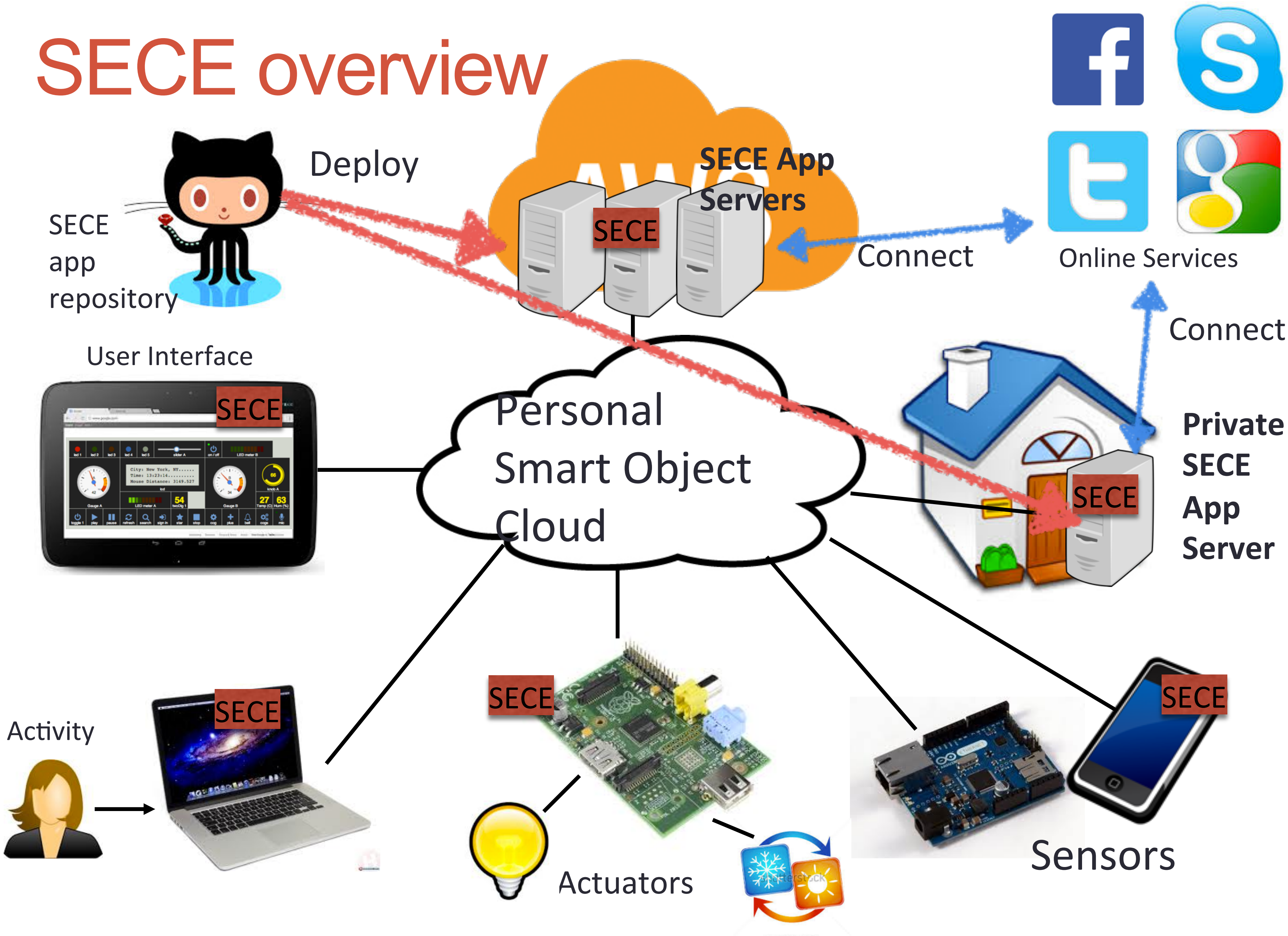
fog computing model

Challenge: integrate embedded, mobile & virtual

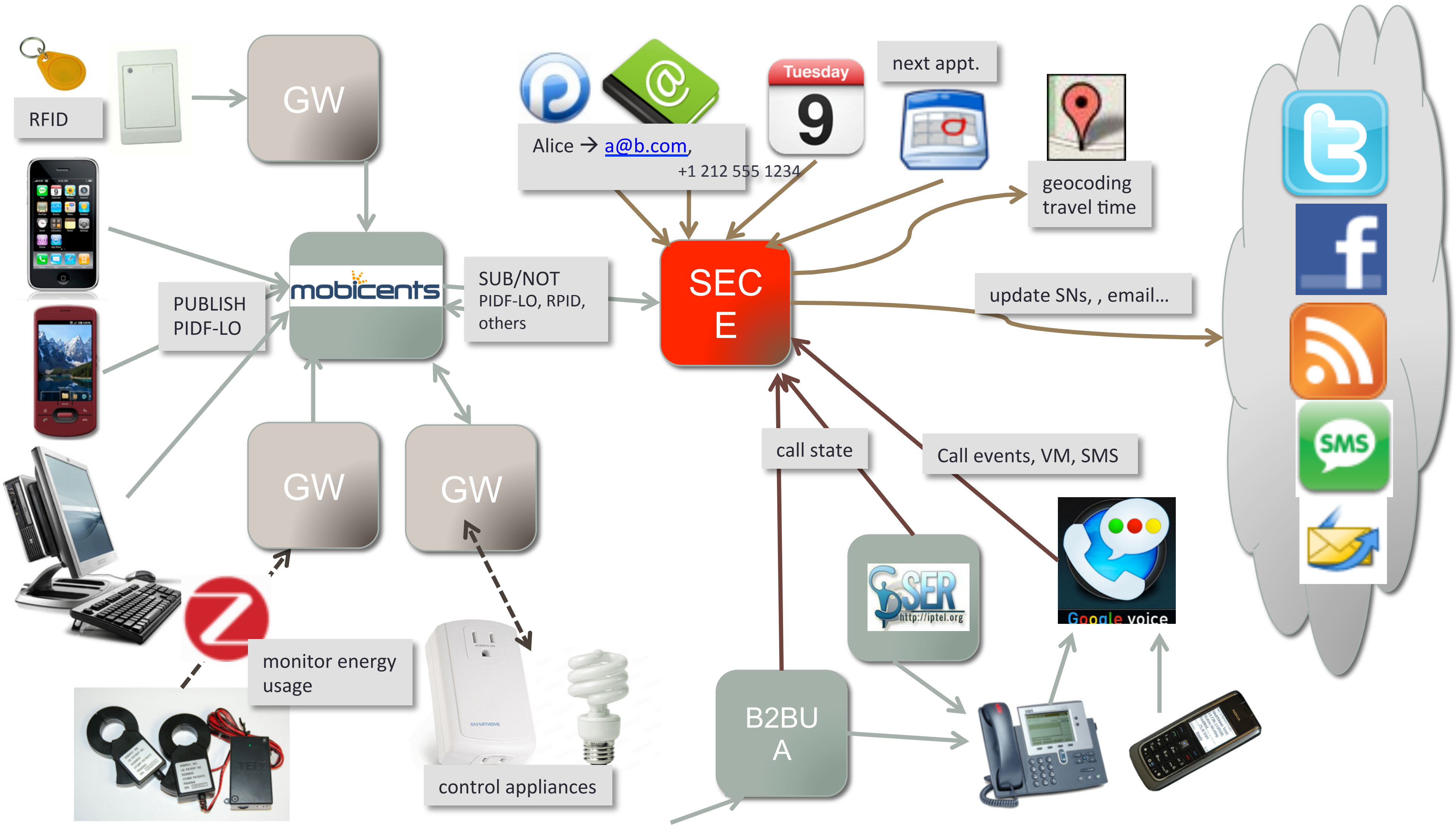


magnetometer
accelerometer
location
gyroscope

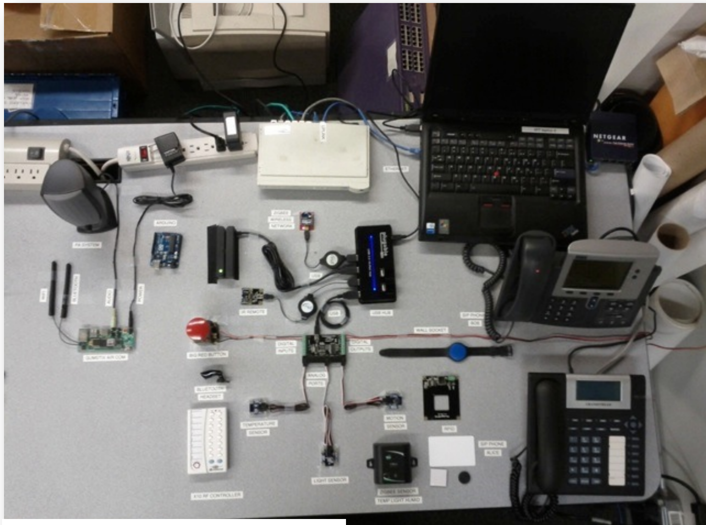
SECE overview

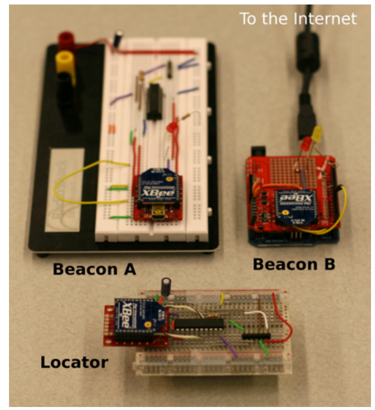


SECE: The glue for Internet applications

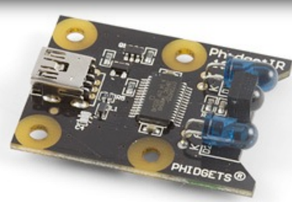


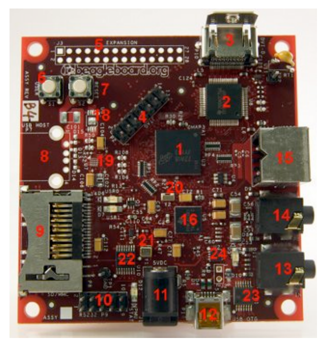
SECE hardware





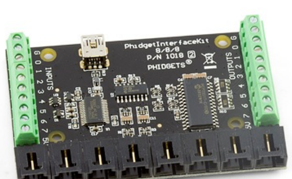
BeagleBoard

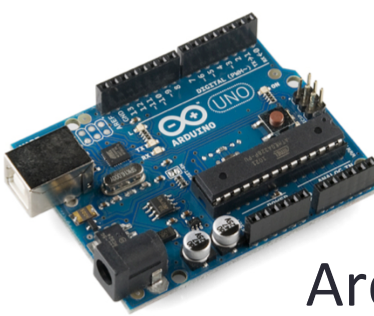




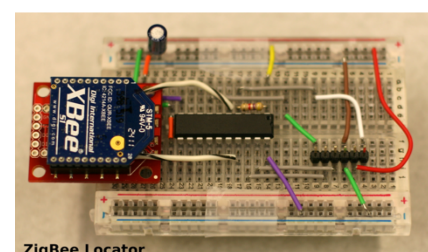
Phidgets

ZigBee



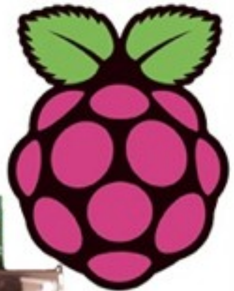


Arduino



Available Platforms

BeagleBoard



Raspberry PI

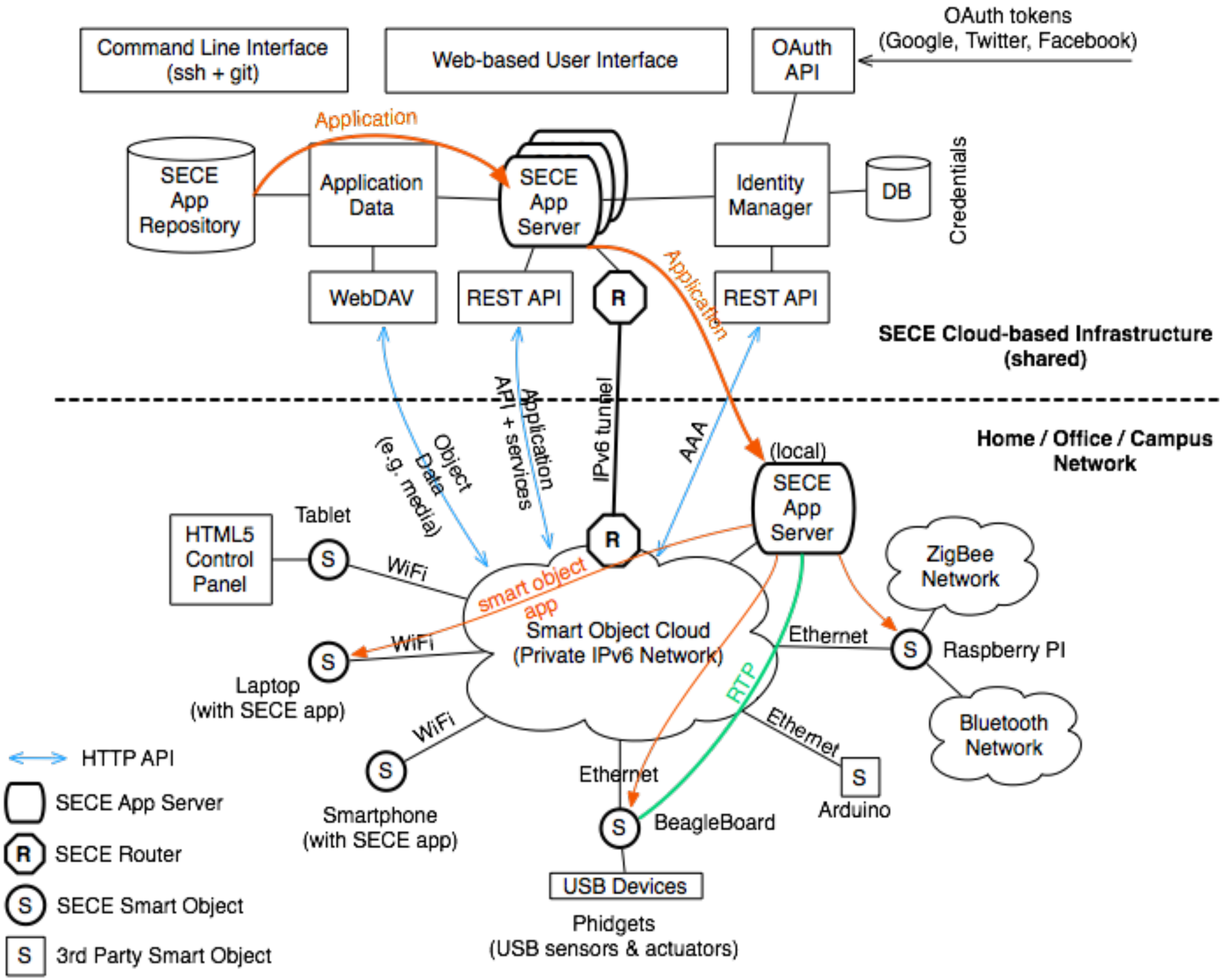
OveroAir

Arduino

Android Devices

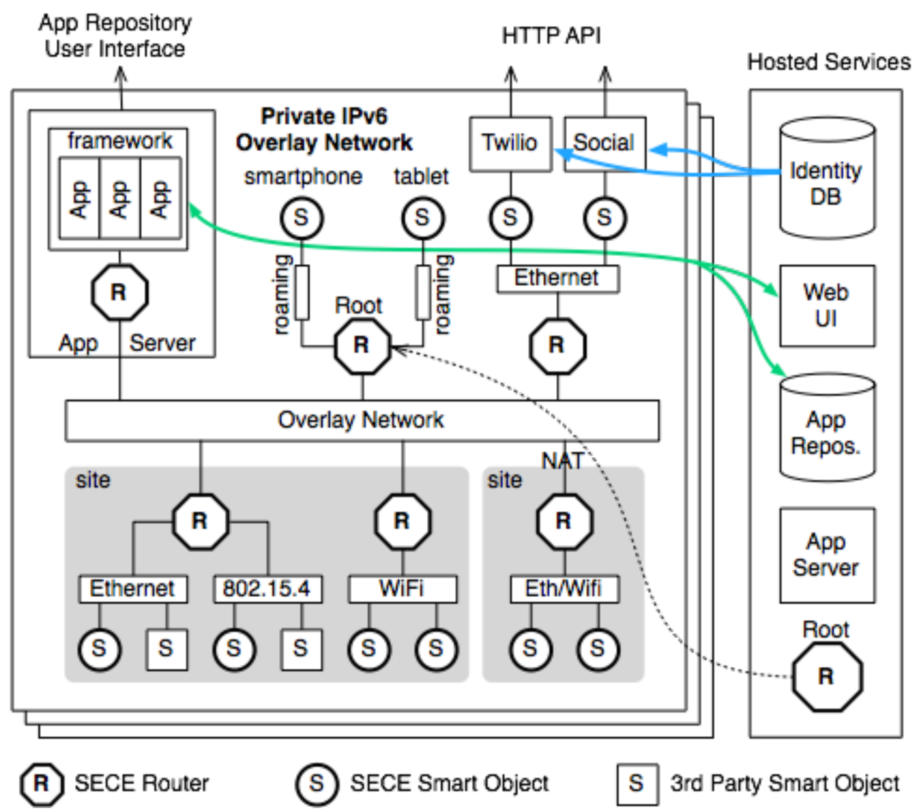
IoT & robotics

SECE system architecture



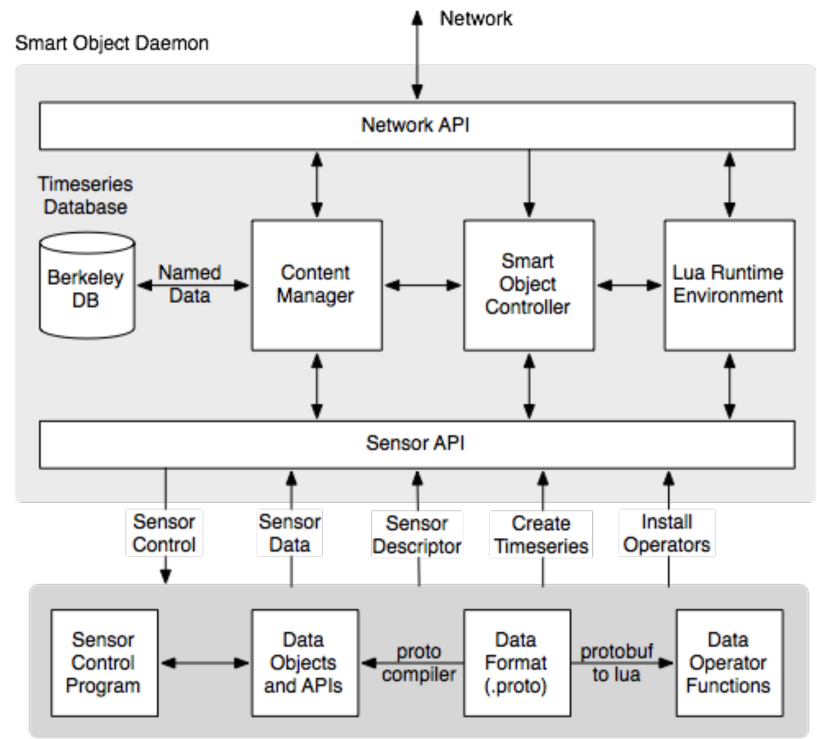
SECE system architecture

- *Application server*
  - Python + coroutines
  - Custom object model
  - App management via **git**
- *Smart object daemon*
  - time-series data management
  - discovery + configuration
  - network APIs
  - platform drivers (Android, OSX, ZigBee, Phidgets
- *HTML5 user interface*
  - virtual devices
  - visual programming
- Overlay router
  - link setup and management
  - addressing + discovery
- *Semantic naming*
  - knowledge base + query language
- *Service gateways*
  - Facebook, Twilio, Google, Twitter



Smart Object Model

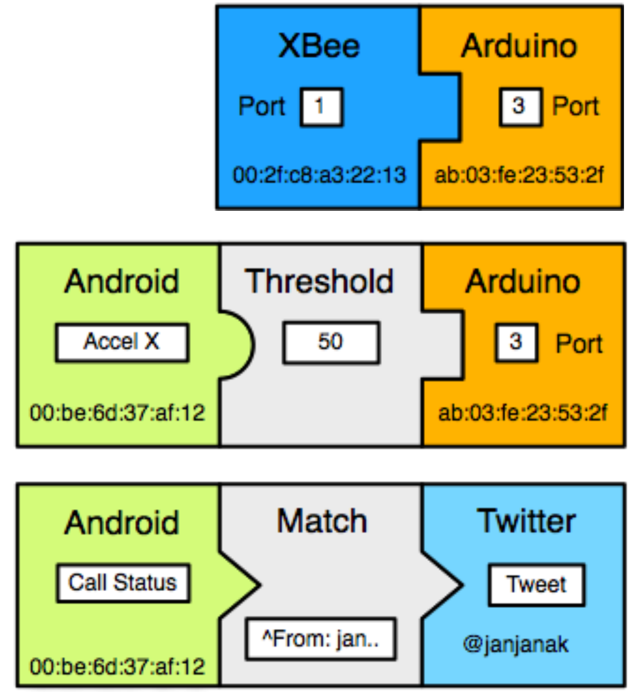
- *Network interface*
  - connects the object to an IP-based overlay network
  - communication protocols and interfaces
  - node discovery, capability negotiation, configuration
- *Data storage*
  - time-series (history) of sensor data
  - collection of named structured documents
  - contents accessible over the network
- *Runtime environment*
  - data processing, programmatic filtering, feedback loops
  - high-level scripting language (Lua, JavaScript)
  - Smart Object Model (SOM) (~ HTML DOM)
- *Sensing and actuation points*
  - addressable components connected via a system bus
  - internal: I2C, SPI, UART (smartphones)
  - external: field bus, BACnet, X-10, XBee, Z-Wave

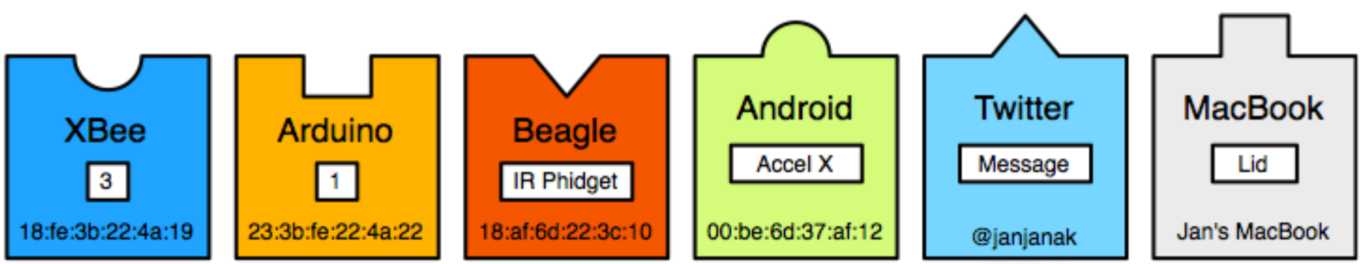


in progress

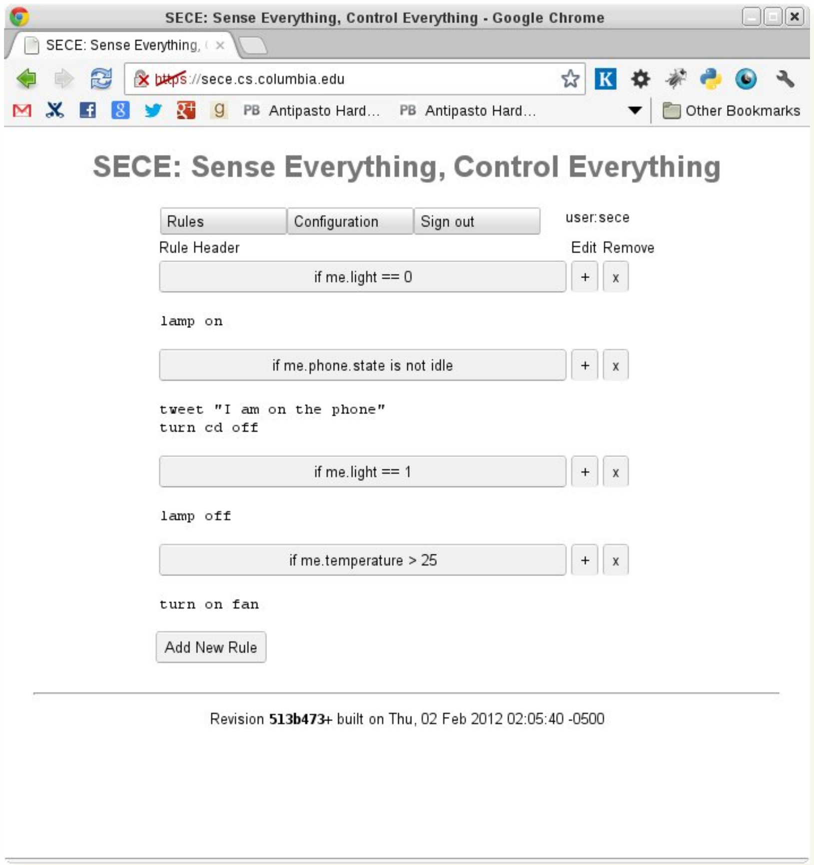
SECE UI: Visual programming

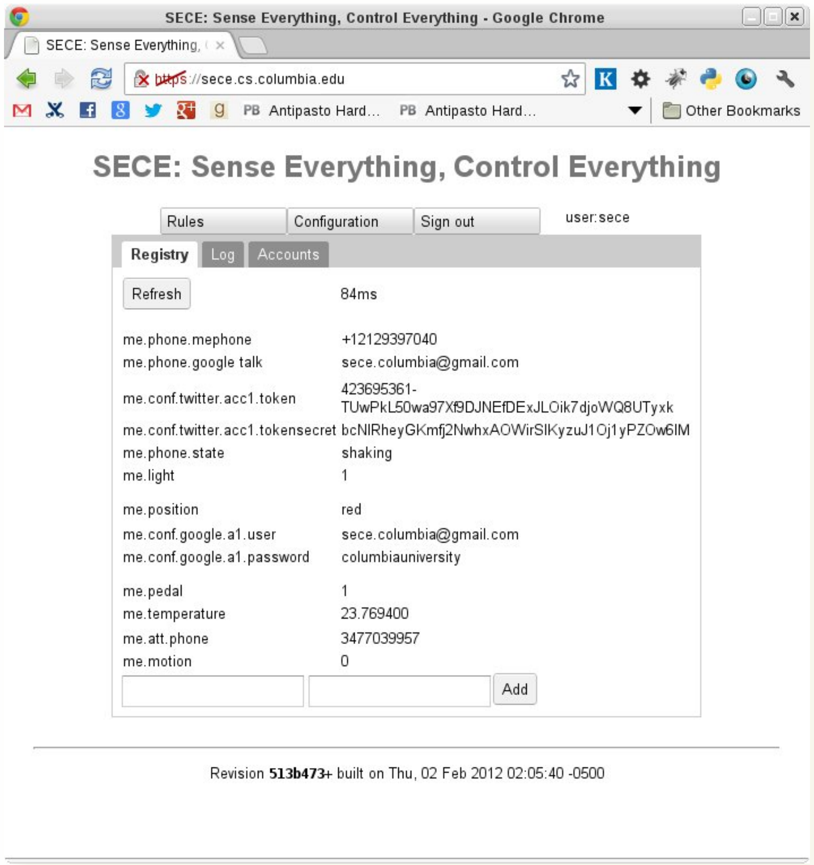
- User-friendly UI for programming smart objects
- Inspired by MIT Scratch
- Blocks represent devices or programs on an application server
- Different latch-socket shapes represent different types of data
- Drag&drop HTML UI
- Suitable for simple add-hoc applications





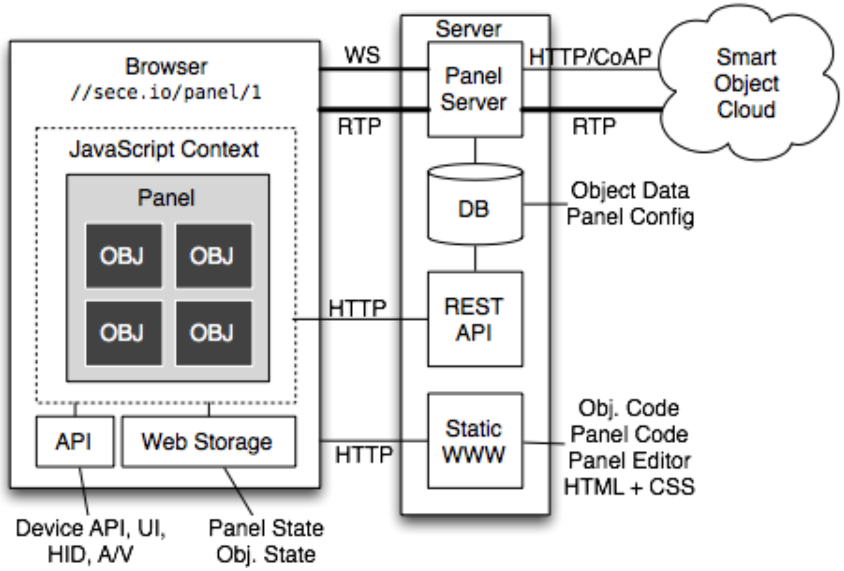
SECE UI: web


SECE: Sense Everything, Control Everything - Google Chrome
https://sece.cs.columbia.edu
SECE: Sense Everything, Control Everything
Rules
Configuration
Sign out
user:sece
Rule Header
Edit Remove
if me.light == 0
lamp on
if me.phone.state is not idle
tweet "I am on the phone"
turn cd off
if me.light == 1
lamp off
if me.temperature > 25
turn on fan
Add New Rule
Revision 513b473+ built on Thu, 02 Feb 2012 02:05:40 -0500


SECE: Sense Everything, Control Everything - Google Chrome
https://sece.cs.columbia.edu
SECE: Sense Everything, Control Everything
Rules
Configuration
Sign out
user:sece
Registry
Log
Accounts
Refresh
84ms
me.phone.mephone +12129397040
me.phone.google talk sece.columbia@gmail.com
me.conf.twitter.acc1.token 423695361-TUwPkL50wa97Xf9DJNEfDExJLOik7djoWQ8UTyxk
me.conf.twitter.acc1.tokensecret bcNIRheyGKmfj2NwhxAOWirSIKyzuJ1Oj1yPZOw6lM
me.phone.state shaking
me.light 1
me.position red
me.conf.google.a1.user sece.columbia@gmail.com
me.conf.google.a1.password columbiauniversity
me.pedal 1
me.temperature 23.769400
me.att.phone 3477039957
me.motion 0
Add
Revision 513b473+ built on Thu, 02 Feb 2012 02:05:40 -0500

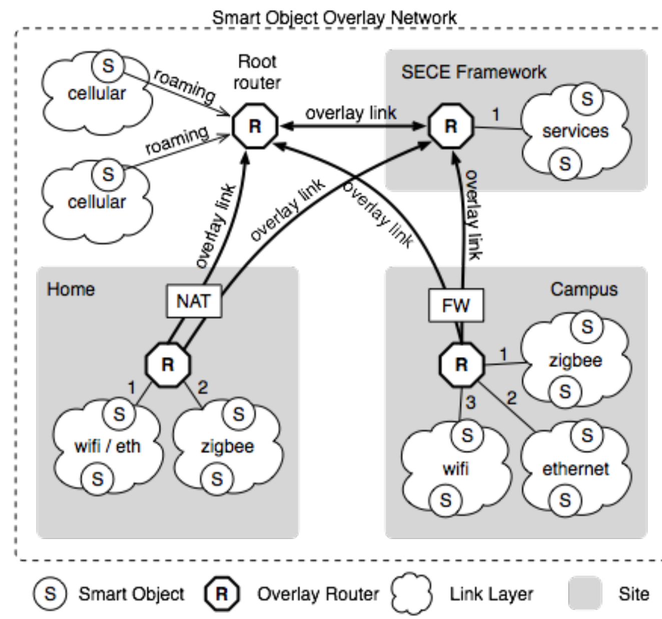
SECE UI: Programmable control panel

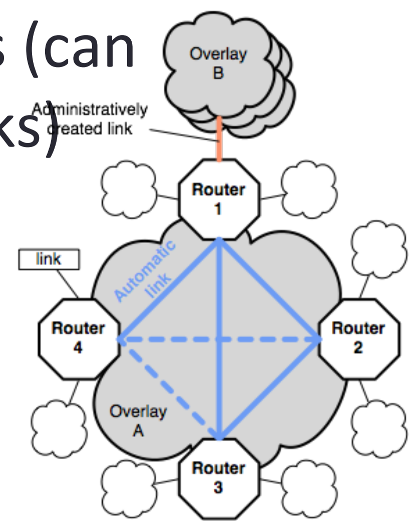
- Virtual JavaScript smart objects
- Emulate physical devices in browser (buttons, LEDs, switches, gauges)
- Create via drag&drop
- Customizable IoT control panels
- Programmable in JavaScript

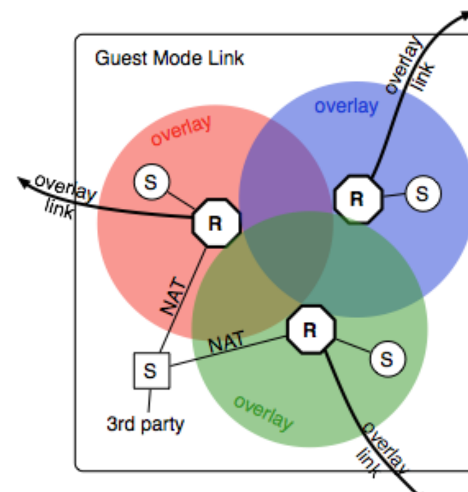


SECE smart object cloud

- IPv6 overlay network architecture
- Automatic tunnel links setup by router nodes
- Spans multiple networks and sites of a SECE user (home office)
- Supports shared local links (can coexist with other networks)
- Similar to an IPv6 VPN





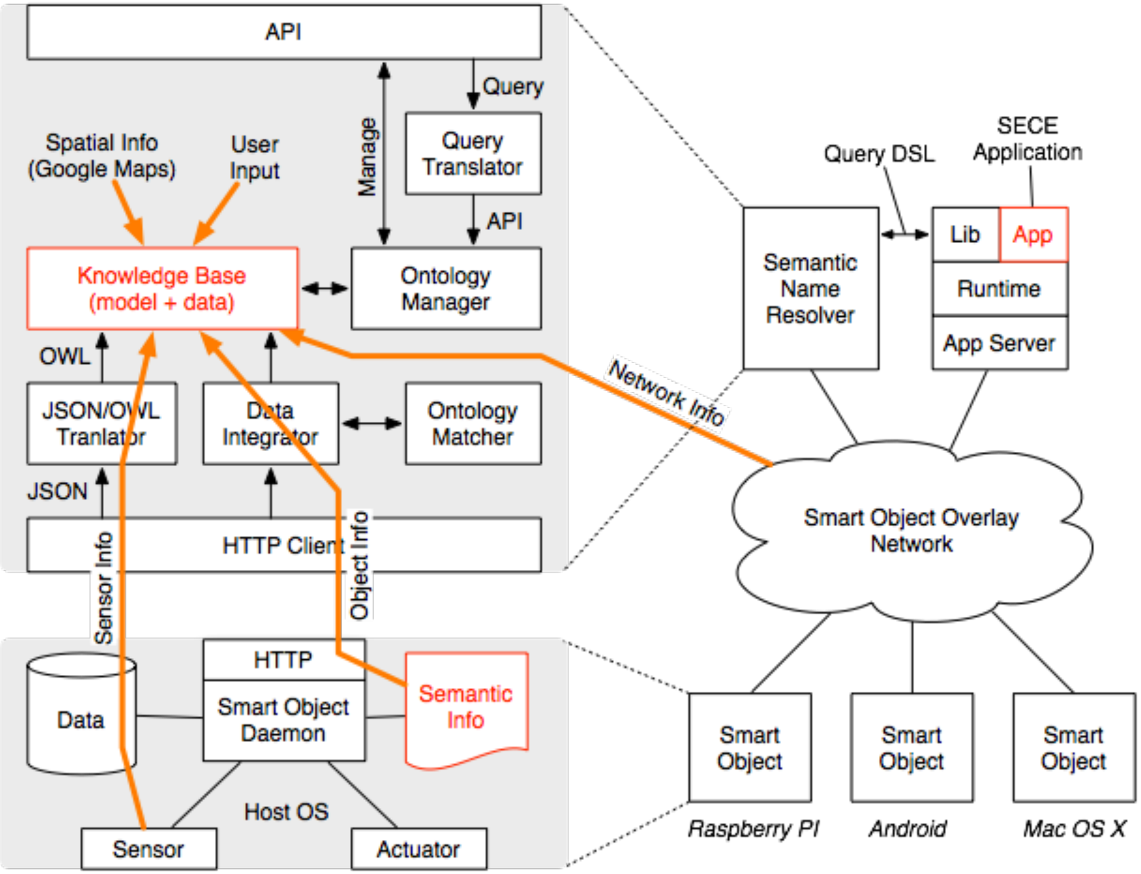


SECE naming (in progress)

Identified	Changes	Example
Specific hardware	when replaced	MAC address or UUID
Specific device	new owner or location	sece://nest42.pseg.com/
Logical device	new location	sece:// co.kitchen.schulzrinne.org
Function	constant (may not exist or may have multiple instances)	sece://cn=us;pr=nj;type=smoke

Semantic naming

- Knowledge base
  - abstract representations of objects
    - sensors, actuators, appliances
  - through human readable query language
- Objects can be referenced
  - through unique IDs
  - properties or context (e.g., location, point of network attachment)
- Unlike SQL, semantic name resolver can infer new information
  - queries return results incorporating the inferred information



Conclusion

- IoT of things as natural progression
  - but limited mainstream in-home applications so far
  - boring commercial applications!
- Network is (relatively) easy, security and software are much harder
- Reaching beyond connectivity and software islands
  - plug-and-program