Bluetooth Cheat Sheet v1.0

Definitions

A wireless technology with open specification for a **low-cost**, **low-power**, **short range** radio technology with 2 main priorities: **cheap**, and **low energy**.

- **Invented** by Ericsson in 1994, the Bluetooth SIG was founded in 1998 with 4 members: IBM, Intel, Nokia, Toshiba. Today has more than 2000 members.

- **Usage**: PAN, ad-hoc networks, data/voice comms, wireless telematics

- **Operates** at 2.4Ghz ISM Band, frequency hopping in spectrum **[2.402 – 2.480]** GHz

- **Profiles**: are additional protocols that defines what kind of data a Bluetooth module is transmitting. While Bluetooth specifications define how the technology works, profiles define how it's used.

- **Qualifications**: interoperability, no license, no LOS, auto ad-hoc network.

Versions

v2.1 2007	**+ Enhanced Data Rate**, basic pairing (button press, numerical pair), speed up to **3Mps**
v3.0 2009	**+ High Speed**, uses 802.11 (a,b,g,n) to send data, speed **up to 24 Mbps**.
v4.0 2010	**+ Low Energy**, speed more than **24 Mbps**.
v4.1 2013	**eliminates the interference of Bluetooth radio with 4G** automatically, improves data transfer and smarter connectivity.
v4.2 2014	introduces **Low Energy data packet length extensions**, Low Energy privacy upgrades and Low Energy secure connections.

Power

Power	Distance	Energy
Class 1	Long range, 100m	20 dBm / 100mW
Class 2	Mid range, 10m	4 dBm / 1-2.5mW
Class 3	Short range, 0.1-10m	0 dBm / 1mW

Ad-hoc Networking

Piconet	Decentral, 1 master – 7 slaves	Point to point or multipoint
Scatter-net	Overlapping of 2 piconets (max 10), different hopping freqs	Peer 2 peer

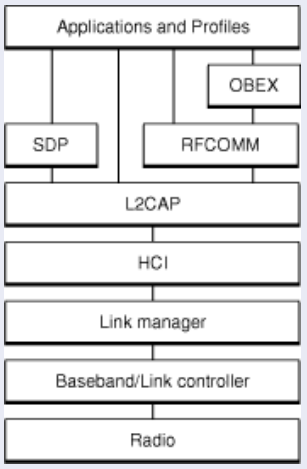
Connection Process

Inquiry: discover others by discovery process.

Paging: forming connection between 2 devices.

Connection: enters connection state. 3 modes: Active – in transmission or receiving of data, Sniff – power-saving mode, regular sleep-wake at every X ms, Hold – instructed to sleep by master and wake up after certain interval, Park – deepest sleep mode, until Master tells to wake up.

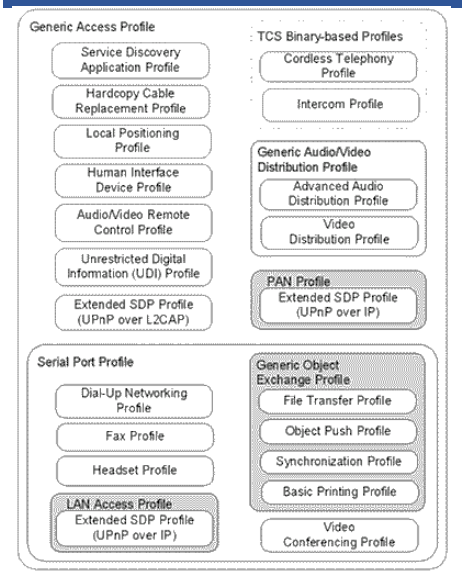
Protocol Stack



Layers & their roles

Bluetooth Radio	Lowest defined layer
Baseband	Physical layer, controls: error, flow, hopping (79 channels), security. Has 3 connection modes: - STANDBY, ACTIVE and Power Saving Mode
Audio	2 codecs: PCM & CVSD, both at 64kbps, using SCO links.
LMP (Link Manager Protocol)	authentication, link setup, acts as service provider, comm with LM PDUs
HCI (Host Controller Interface)	command interface, hardware status, can be via UART, RS232, or USB
L2CAP (Logical Link Control and Adaptation Protocol)	Establish connection (less/full) to upper layer. 2 links types are used: SCO (not supported by L2CAP), ACL (Async conn less)
RFCOMM (Radio Frequency Communication)	Emulation of serial ports, up to 60 simulaneous conns.
SDP (Service Discovery Protocol)	For discovery of services and their characteristics using request/response of one PDU each time.

Bluetooth Profile Structure



A2DP **Advanced Audio Distribution Profile** Defines the streaming of high quality audio signal between devices (one way).

AVRCP **Audio/Video Remote Control Profile** Defines the standard for remote control of TVs, Hifi, etc.

BIP **Basic Imaging Profile** For sending (pull/push) images between devices with ability to resize, convert images.

BPP **Basic Printing Profile**

DID **Device ID Profile** Identifies device, manufacturer, product version, etc.

DUN **Dialup Networking Profile**

FTP **File Transfer Profile**

GAVDP **General Audio/Video Distribution Profile** – basis for A2DP and VDP.

GEOP **Generic Object Exchange Profile** – based on OBEX

HFP **Hands-Free Profile** – uses SCO to carry mono voice audio data.

HID **Human Interface Device Profile** – input devices such as mouse, joysticks, keyboards, etc.

PAN Personal Area Networking Profile

SDAP Service Discovery Application profile

Bluetooth Classic v/s Bluetooth 4.0 LE

IEEE Standard	802.15.1	802.15.1
Frequency (GHz)	2.4	2.4
Maximum raw bit rate (Mbps)	1-3	1
Typical data throughput (Mbps)	0.7-2.1	0.27
Maximum Range (Meters)	10 (class 2), 100 (class 1)	50
Relative Power Consumption	Medium	Very low
Example Battery Life	Days	Months to years
Network Size	7	Undefined

NFC Cheat Sheet – Part 1 v1.0

Definitions

NFC or **Near Field Communication** is a short range high frequency wireless communication technology.

NFC is mainly aimed for **mobile** or **handheld devices**.

A **radio communication** is established by touching the two phones or keeping them in a proximity of a few centimeters (up to **10 cm**) .

It allows for simplified transactions, data exchange, and wireless connections between two devices.

Allows communication between
- Two powered **(active)** devices
- Powered and non self-powered **(passive)** devices

Features

NFC is an extension of **Radio frequency identification (RFID)** technology that combines the interface of a smartcard and a reader into a single device. This allow **two-way communication** between endpoints, where earlier systems were one-way only.

It operates within the globally available and unlicensed radio frequency band of **13.56 MHz**, with a bandwidth of **14 kHz**.

Working distance with compact standard antennas: up to **10 cm** .

Supported data rates: **106, 212** and **424 Kbit/s**

For two devices to communicate using NFC, one device must have an NFC **reader/writer** and one must have an **NFC tag**

Specification	Purpose
NFC Data Exchange Format (NDEF)	Defines a common data format between NFC-compliant devices and tags
Record Type Definition (RTD)	Specifies rules for building standard record types Five specific RTDs (**Text, URI, Smart Poster, Generic Control, and Signature**) are used to build standard record types
Logical Link Control Protocol (LLCP)	Defines a protocol to support peer-to-peer communication between two NFC-enabled devices
Connection Handover	Defines how to establish a connection using other wireless communication technologies
Operations Specifications for Four Tag Types (1/2/3/4)	Enable core interoperability between tags and NFC devices

Reader/writer mode

the NFC device is capable of reading NFC Forum-mandated tag types, such as a tag embedded in an NFC smart poster

Peer-to-Peer mode

Two NFC devices can exchange data. For example, you can share Bluetooth or Wi-Fi link set-up parameters or you can exchange data such as virtual business cards or digital photos.

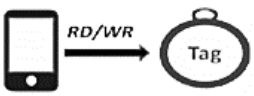
Card Emulation mode

The NFC device appears to an external reader much the same as a traditional contactless smart card. This enables contactless payments and ticketing by NFC devices without changing the existing infrastructure.

History

- **1983** · The first patent to be associated with the abbreviation RFID was granted to Charles Walton.
- **2004** · Nokia, Philips and Sony established the Near Field Communication (NFC) Forum.
- **2011** · First mobile phone(Nokia 6131) with NFC released by NOKIA.
- **2010** · Samsung Nexus S: First Android NFC phone.
- **2011** · NFC support becomes part of the Symbian mobile operating system and Blackberry OS.

NFC Two Modes of Communication



Passive Communication Mode: The Initiator device provides a carrier field and the target device answers by modulating existing field. In this mode, the Target device may draw its operating power from the Initiator-provided electromagnetic field.

Active Communication Mode: Both Initiator and Target device communicate by alternately generating their own field. A device deactivates its RF field while it is waiting for data. In this mode, both devices typically need to have a power

Aspect	NFC	Bluetooth	Bluetooth Low Energy
RFID compatible	ISO 18000-3	active	active
Standardisation body	ISO/IEC	Bluetooth SIG	Bluetooth SIG
Network Standard	ISO 13157 etc.	IEEE 802.15.1	IEEE 802.15.1
Network Type	Point-to-point	WPAN	WPAN
Cryptography	not with RFID	available	available
Range	< 0.2 m	~100 m (class 1)	~50 m
Frequency	13.56 MHz	2.4–2.5 GHz	2.4–2.5 GHz
Bit rate	424 kbit/s	2.1 Mbit/s	25 Mbit/s
Set-up time	< 0.1 s	< 6 s	< 0.006 s
Power consumption	< 15mA (read)	varies with class	< 15 mA (read and transmit)

NFC Cheat Sheet – Part 2 v1.0

Benefits of NFC

Versatile: NFC is ideally suited to the broadest range of industries, environments, and uses

Open and standards-based: The underlying layers of NFC technology follow universally implemented ISO, ECMA, and ETSI standards

Technology-enabling: NFC facilitates fast and simple setup of wireless technologies, (such as Bluetooth, Wi-Fi, etc.)

Inherently secure: NFC transmissions are secure due to short range communication

Interoperable: NFC works with existing Contactless card technologies

Security-ready: NFC has bui  bilities to support secure applications

Applications of NFC

Asset Management – Use NFC phones to read smart tags per product for inventory

Access – Ensure secure building area access for personnel with NFC device and contactless reader

Parking - Use NFC to authenticate parking entry and keep record .

Meal orders - Customers order their meals by touching NFC Smart Posters.

- **Remote worker reporting** - Remote workers confirm locations visited and tasks completed
- **Maps** - An interactive NFC Smart Poster map allows the user to download the map, get additional information on relevant services, and access coupons, etc.
- **Events calendar** - Users can download tickets or coupons or be linked to event websites

IoT Hardware Cheat Sheet v1.0

Arduino

Open-source hardware prototyping platform, powered by **Arduino programming language** (based on Wiring, o/s framework for microcontrollers) and **Arduino IDE** (based on Processing).

- Simple **microcontroller**, usually Atmel chip such as ATmega328, 8-6-32 Mhz.

- **Programmed** in C (i.e. sketches) in Arduino IDE, runs in loop (as embedded software)

- Operating at **3.3V or 5V**, input voltage **6-20V**

- Has **digital pins** (1/0), which can be used for both input and output. Some special pins such as **PWM** can vary the amount of time that it's ON/OFF really fast so it can simulate an analog output signal.

- Has **analog pins**, which are mostly used for input, with a built-in A/D converter.

- **Flash memory size** 32KB, 64KB. ~0.5KB used by the bootloader.

- **SRAM size** 2KB, 4KB, 8KB depending on the processor.

Raspberry Pi

A credit card–sized **single-board computers** developed in the UK RPi Foundation to promote teaching of basic computer science in schools and developing countries.

- **ARM-based processor**, 700 MHz single-core ARM1176JZF-S (model A, A+, B, B+, CM), 900 MHz quad-core ARM Cortex-A7 (RPi 2)

- **Linux/Unix Operating system**, e.g. Raspbian, RISC OS, FreeBSD, NetBSD, Plan 9, Inferno, AROS

- **256 MB RAM** (model A, A+, B rev 1), **512 MB** (model B rev 2, B+, CM), 1 GB RAM (RPi 2)

- **Storage** uses SDHC slot (model A and B), Micro SDHC slot (model A+ and B+)

- Powerful Broadcom Video Core IV **graphic card**

- **Power**:1.5 W (model A), 1.0W (model A+), 3.5W (model B) or 3.0W (model B+), 4.0W (RPi 2)

- **Networking**: built-in USB Ethernet adapter (except model A/A+)

- **Low-level I/O**: GPIO: UART, I²C bus, SPI bus with two chip selects, I²S audio +3.3 V, +5 V, ground.

Choosing IOT Hardware Device

Base on: **Form Factor**, **Shape**, **Function**, **Dev/Design**, **Cost**, and **Manufacturability**.

RFC 7228 defines 3 classes of device depending on **RAM** and **flash memory size**:

Class 0	<< 10 KiB	<< 100 KiB
Class 1	~ 10 KiB	~ 100 KiB
Class 2	~ 50 KiB	~ 250 KiB

Famous microcontroller manufacturers: MicroChip, Atmel, Intel, Analog devices, etc.

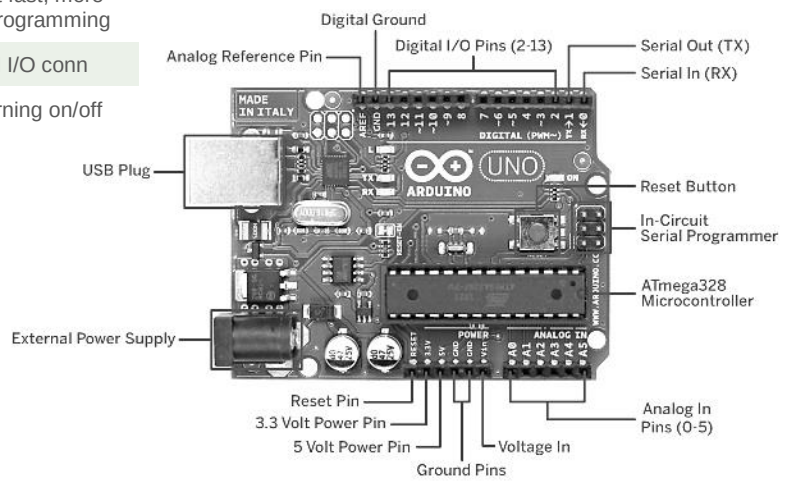
IoT General HW Bus/Port Types

I2C	Easiest and most expandable bus, easy for programming.
SPI	**Serial Peripheral Interface** Limited number of devices, but fast, more difficult to programming
UART	RS-232, old I/O conn
GPIO	Good for turning on/off IO ports.

Arduino UNO vs Raspberry Pi vs BeagleBone

Name	Arduino Uno	Raspberry Pi	BeagleBone
Model Tested	R3	Model B	Rev A5
Price	$29.95	$35	$89
Size	2.95"x2.10"	3.37"x2.125"	3.4"x2.1"
Processor	ATMega 328	ARM11	ARM Cortex-A8
Clock Speed	16MHz	700MHz	700MHz
RAM	2KB	256MB	256MB
Flash	32KB	(SD Card)	4GB(microSD)
EEPROM	1KB		
Input Voltage	7-12v	5v	5v
Min Power	42mA (.3W)	700mA (3.5W)	170mA (.85W)
Digital GPIO	14	8	66
Analog Input	6 10-bit	N/A	7 12-bit
PWM	6		8
TWI/I2C	2	1	2
SPI	1	1	1
UART	1	1	5
Dev IDE	Arduino Tool	IDLE, Scratch, Squeak/Linux	Python, Scratch, Squeak, Cloud9/Linux
Ethernet	N/A	10/100	10/100
USB Master	N/A	2 USB 2.0	1 USB 2.0
Video Out	N/A	HDMI, Composite	N/A
Audio Output	N/A	HDMI, Analog	Analog

Arduino UNO Board Layout



IoT Platforms Cheat Sheet – Part 1 v1.0

Xively

Xively is a **PaaS** for building connected IoT products, through Xively messaging platform, one can monitor the device signals, controls their status, integrates the input data with third party systems.

Xively Setup

1. **Setup** Xively account.

2. **Add New Device** in Xively Developer Center. Xively Workbench will provide 2 important data:
- **API Key**: key used for CRUD resources on Xively, usually configured onto the Local Gateway or the device with direct Internet access.
- **Feed ID**: ID or URL used for client website/app to read device data, status, etc.

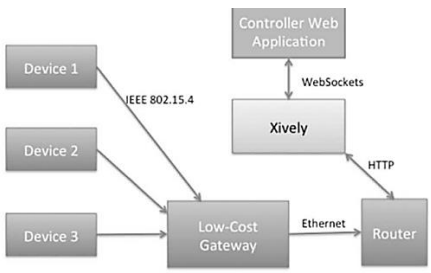
3. **Create channels** for the device.
A channel represents an attribute or a characteristic of the device, for example the value of light sensitivity from a photo cell. Channels come with additional data such as:
- **Location**: allows adding location data to the channels manually (fixed location) or dynamically (devices can also send the location data itself)
- **Metadata**: specifies the creator of the feed, name, website, group, etc.
- **Triggers**: a kind of IFTTT for self triggering of own service or third-party services (base on the comparison result of the channel data).
- **Request Log**: for real-time debugging and monitoring of device.

3. **Deploy** – Add Product Batch and Product
This action creates a batch of virtual products that will map to the physical devices. The mapping is done via the Serial Number of each device. The serial number can be added by uploading a CSV file or entering manually.

Wislab

Czech wireless laboratory, develops a visualization framework for sensor devices plugged in to Xively cloud.

SensMap Visualization Framework
- Visualize **state** of sensor devices
- Visualize **location** of sensor devices outdoor and in-building
- Manage devices' location via **interactive map or floor plan**
- Report sensor values with **graphs**, and **comparison reports**
- Visualize wireless sensor **topology** and **signal quality** between nodes



Xively Terminology

Activation Code	The code submitted to Xively by a device with Internet connectivity when the device powers on. A device creates its activation code by generating a HMAC-SHA1 ID, based on the serial number of the device and the product secret stored in the device's firmware. The Activate Device API stores the activation code and activation date as attributes of the device in the database and returns the Feed_id and api_key to the device.
API Key	API Keys are used to control access to the resources via the API. An API key is a hierarchy of three objects: key, permission and resource. A key object contains one or more permissions objects, and a permissions object can optionally contain resource objects.
Channel	Every DataStream in the API is a two-way communications channel on the Xively platform. Channels are limited to 32 characters per Data point value.
Datapoint	A single value of a DataStream at a specific point in time. It is simply a key value pair of a timestamp and the value at that time.
Datastream	A bi-directional communications channel that is used for exchanging of data between Xively and authorized devices, applications, and services.
Device	Devices are individual, physical instances of a product. Each device in a batch of products is provisioned with a unique serial number which it uses to activate on the Xively service.
Environment / Feed	The collection of channels (data streams), defined for any given device. A Feed's metadata can optionally specify location, tags, whether it is physical or virtual, fixed or mobile, indoor or outdoor, etc. Every device has exactly one Feed.
Frozen Feed	A Feed which was last updated **more than 15 minutes** ago.
Live Feed	The Feed was updated with data **within the last 15 minutes**.
Gateway	A gateway is a conceptual element of connected objects.
Historical Query	A query that returns a list of historical Data points within the specified range for one or more DataStream's.
Master Key	A master key is a non-resource restricted, private key, which has permissions to perform all HTTP methods.
Pre-registering	To pre-register a group of devices with Xively, use Xively Workbench to upload the serial numbers made available to you by the manufacturer. For each serial number uploaded, Xively uses the Create Device API to create the device in the database, setting the serial and the created_at attributes. So when a pre-registered device transmits its activation code to Xively, the API knows that these devices are authorized to be used on Xively.
Product Secret	A random hex string automatically generated when a product is created in the provisioning system. The product secret is used in the activation process.
Provisioning	The process whereby a physical device sharing a serial number with an inactive device defined in a product batch in Xively comes online, submits an activation code to Xively and receives its configuration information back so that the device can interact with and through Xively.
Resources	Xively exposes 7 resources via URLs (Uniform Resource Locators): Feeds, DataStream, Data points, Keys, Triggers, Products, and Devices. To access a Xively resource, you submit an HTTP request to one of the published URLs using the four HTTP verbs (DELETE, GET, POST, PUT) and indicating the format representation (JSON, XML, CSV).
Serial Number	a unique ID assigned to each device, accessible to the activation code running on the device. This ID can be a value generated and programmed into the device during the manufacturing process, or it can be a device specific identity such as a MAC address or processor serial number.
Location Waypoints	The collection of locations registered for a device whose disposition is mobile. You cannot delete or change waypoints.

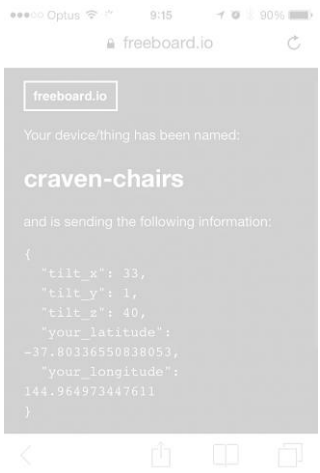
Xively Featured Hardware Partners

eConais	Wireless modules manufacturer
mbed	Platform and OS for IoT devices based on 32-bit ARM Cortex-M
TST	System integrator providing customized wireless solutions
Arduino	Open-hardware prototyping platform
electric imp	Wi-Fi enabled development platform powered by Cortex M3

IoT Platforms Cheat Sheet – Part 2 v1.0

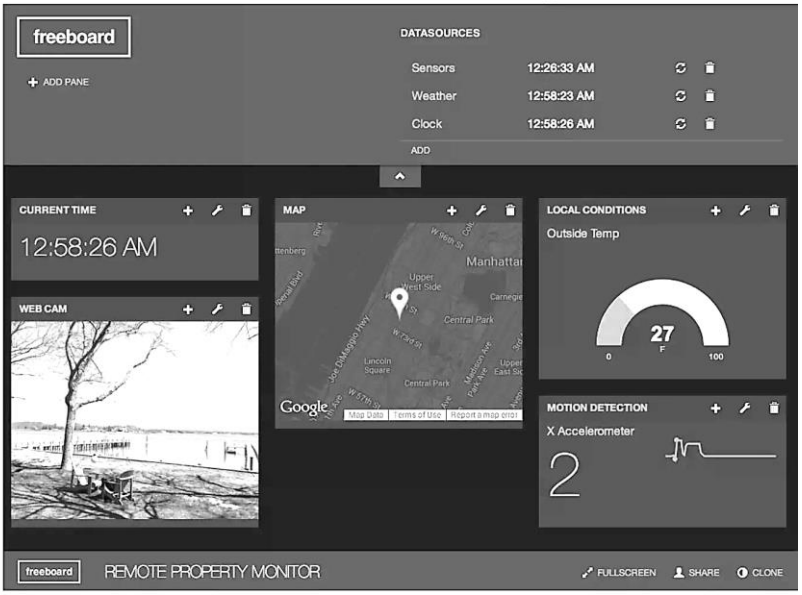
FreeBoard.io

Freeboard gather live “streaming” data coming from IoT devices such as from "dweets" and present them as meaningful graphs, text and gauges. It might be tracking the temperature in the home alongside its energy usage, or having a dashboard that monitors pollution around the city using GPS and air quality sensors attached to shared-bicycles.



Drag & drop widgets

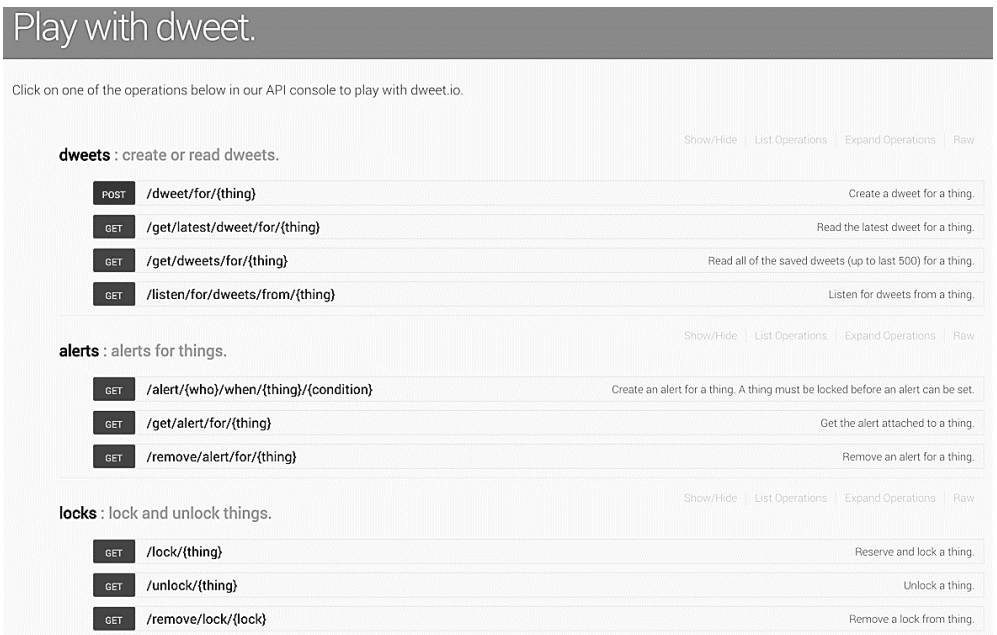
Add devices and datasources



Share it instantly

Dweet.io

Dweet.io allows users to share data from mobile, tablets, and pcs, and them to other devices and accounts across social media platforms. Dweet.io provides an API to access the different functionality of the Dweet.io service. Users can make REST calls to read and create dweets, lock and unlock things, and perform other calls. The API returns JSON and JSONP.



IoT Architecture Cheat Sheet v1.0

IoT Architecture Types

- **Network/Connectivity Architecture**
  - Routing/Switching devices
  - Network Management
  - Topology
  - Optimization...
- **Protocol Architecture**
  - Protocol Stack Layering
  - Stack Link
  - Protocol Modification/Profiles...
- **Software Architecture**
  - Abstraction of software system
  - Design Pattern / System/Software Engineering
  - Process Modeling / Enterprise Modeling /UML
  - User Cases / Functional Blocks
- **Service Architecture**
  - TMN
  - NGOSS
  - ITIL
- **Grand Architecture**
  - All of the above, and Addressing/Identification, Marketing/Service requirements etc...
- **Application Specific Architecture**
  - A focus view of a specific application that utilize all necessary architectural components and perspective

IoT-A Channel Model

Types of Wireless Sensor Networks

1. Unstructured WSN

- Dense collection of nodes
- Ad-hoc deployment
- Difficulty in network maintenance

2. Structured WSN

- Few and scarcely distributed nodes
- Pre-planned deployment
- Lower network maintenance

Transport Layer's Main Concern

Congestion	When buffer reaches the **lower threshold**, the congestion bit is set with a certain probability. When buffer reaches the **higher threshold**, the congestion bit is set for all packets.
Reliability	Direction, reliability measure, packet recovery
Energy Conservation	

Network Layer's Main Concern

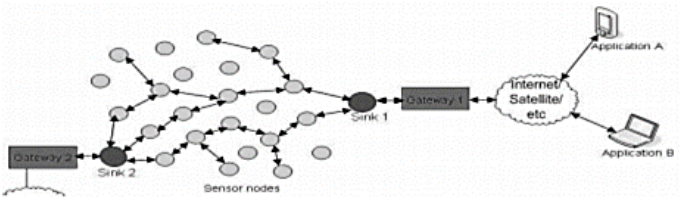
Traffic Flow	Routing type, data aggregation, computation overhead
Energy Efficiency	Extra energy requirement

Architecture Model Contents

- Physical Entity view
  - "any physical object that is relevant from a user or application perspective"
- Deployment view
- Operational view
- IoT Context view
  - focuses mainly on what lies outside the system and how the system interfaces to the outside world
- IoT Domain Model
  - Defines what the Physical Entities in the IoT Context View are and how these entities relate to each other and what is to be done with them (sensing of properties,..)
- Functional view
- Information view
  - what physical quantities are monitored by the sensors; how are the quantities related to each other, etc.? In the recurring example the quantity that is handled by the system is the air temperature in the cargo area of the truck.

Wireless Sensor Networks

- Large number of heterogeneous Sensor node devices spread over a large field.
- Wireless sensing + Data Networking.



Smart Sensor Nodes:

- Low power devices
- Consists of- one or more sensors, a processor, memory, a power supply, a radio and an actuator.

A WSN can be defined as

- a network of devices, denoted as **nodes**,
- which can sense the environment and communicate the information gathered from the monitored field (e.g., an area or volume) through **wireless links**.
- The data is forwarded, possibly via multiple hops, to a **sink** (sometimes denoted as **controller** or **monitor**) that can use it locally or is connected to other networks (e.g., the Internet) through a gateway.
- The nodes can be stationary or moving.
- They can be aware of their locations or not.
- They can be homogeneous or not.
- Monitoring and communication are performed cooperatively by the nodes

Fog Computing

Definition	Architecture facilitating **end-user clients** or **near-user edge devices** (without going to the Cloud), for following tasks: storage, communication, control, configuration, measurement, management. Support IoX (Internet of Anything)
v/s Cloud Computing	Computing services **at the edge of the network** v/s servers in a DC, emphasizing: proximity to end-users, dense geographical distribution, local resource pooling, latency reduction for QoS, edge analytics mining, better UX, redundancy for cloud failure.

IoT Sensor Technologies Cheat Sheet v1.0

What is Sensor?

- Device that receives and responds to a signal or stimulus.
- It converts stimulus / energy into electrical signals (e.g. Voltage). **Note**: Tranducer can also convert energy to energy (speaker) but is **not** a sensor.

Categories
- Contact v/s Contactless
- Motion, Environmental, Position

Product Types
- Camera, Microphone
- Radar, Accelerometer, Compass, Motion Detector
- Gas Detector

Sensing Attributes

Camera	Microphone	Velocity
Acceleration	Force	Strain
Light	Acoustic	Motion
Temperature	Flow	Pressure
Humidity	Chemicals	

Actuator: converts electrical signal back to energy form. E.g. motor, vibrator.

Each **sensor type** offers different levels of: **Accuracy, Sensitivity, Specificity, Ability, Cost consideration**.

Mechanical Sensors	MEMS, Accelerometers, Gyroscopes
Optical Sensors	Photodetectors, IR, Fiber Optic, Interferometers
Semiconductor Sensors	Gas Sensors, Temperature Sensors, Magnetic Sensors, Optical Sensors, Ion-Sensitive Field-Effect Transistors
Electrochemical Sensors	Potentiometric Sensors, Amperometric Sensors, Coulometric, Conductometric Sensors
Biosensors	Transducers for Biosensors, Key Characteristics of Biosensors

Sensor Characteristics Detail

Sensitivity	Is the change in input required to generate a unit change in output.
Hyteresis	Output of a sensor may be different for a given input, depending on **whether the input is increasing or decreasing**.
Resolution	Discrimination, or the smallest increment of measurand that causes a detectable change in output.

Sensor History

first thermostat in **1883**
the concept of biosensing was first proposed by Clarke and Lyons in **1962**.
The concept of the glucose biosensor was brought to commercial reality in **1975** by the Yellow Springs Instrument Company
In **1959**, CalTech Richard Feynman ecture at the "There is Plenty of Room at the Bottom." – outlined the basic concepts and techniques for MEMS devices.
1998 with MEMS-based gyroscopes from Bosch for commercial applications in the automotive sector
1995 comprise several sensors that measure physiological signals of interest and make that data available wirelessly to a computing device.
2010+ IoT, SoC Sensors

Temperature Sensors

A temperature sensor is a device, typically, a thermocouple or RTD, which is provided for temperature measurement through an electrical signal.

A thermocouple (T/C) is made from two dissimilar metals that generate electrical voltage in direct proportion to changes in temperature.

Temperature sensing can be done either through direct contact with the heating source material or remotely, with indirect touch with the source using radiated energy instead.

There is a wide range of temperature sensors available in the market and are listed below.

- Thermocouple
- The RTD
- Thermistors
- Semiconductor sensors
- Digital Temperature Sensors

Thermocouple

It is a type of temperature sensor, made by joining two dissimilar metals at one end.

The joined end is referred to as the HOT JUNCTION.

The other end of these different metals is called the COLD END or COLD JUNCTION.

The cold junction is actually formed at the last point of thermocouple material.

If there is a difference in temperature between the hot junction and cold junction, a small voltage is created.

Sensor Characteristics Detail P2

Accuracy	Sensor's ability to provide an output close to the true value of the measurand.
Precision	The reproducibility of a measure, quantified as % of std deviation from the mean.
Errors	Systematic Errors, Random Errors (Noise), Error Bands
Repeatability	Ability to produce the same output when the same input is given.
Tolerance	The variations in the reported output among a batch of similar elements due to random manufacturing variations.

Drivers for Sensor Applications

Health and Fitness
Aging Demographics
Personalized Healthcare
Public Health
Technology Nexus
- sensors have evolved beyond being just "dumb" sensing devices to become smart sensors or sensor systems through the integration of ICT technologies

National Security
- potential attacks from chemical, biological, radiological, and nuclear (CBRN) sources

The Internet of Things
- temperature, CO_2, light, noise, moisture)

Water and Food
Environmental Challenges

Sensor Characteristics

- Range
- Transfer Function
- Linearity and Nonlinearity
- Sensitivity
- Environmental Effects
- Modifying Inputs
- Interfering Inputs
- Hysteresis
- Resolution
- Accuracy
- Precision
- Error
- Statistical Characteristics
- Repeatability
- Tolerance
- Dynamic Characteristics

Digital Temperature Sensors

Digital output sensor usually contains a temperature sensor, analog-to-digital converter (ADC), a two-wire digital interface and registers for controlling the IC's operation.
Temperature is continuously measured and can be read at any time. If desired, the host processor can instruct the sensor to monitor temperature and take an output pin high (or low) if the temperature exceeds a programmed limit. Lower threshold temperature can also be programmed, and the host can be notified when the temperature has dropped below this threshold.
Thus, digital output sensor can be used for reliable temperature monitoring in microprocessor-based systems.

Semiconductor Sensors

They are classified into different types like Voltage Output, Current Output, Digital Output, Resistance Output Silicon and Diode Temperature Sensors.

Modern semiconductor temperature sensors offer high accuracy and high linearity over an operating range of about 55°C to +150°C.

Internal amplifiers can scale the output to convenient values, such as 10mV/°C.

They are also useful in cold-junction compensation circuits for wide temperature range thermocouples.

Sensor Characteristics Detail P3

Response Time	Time taken for the sensor to change its output from its previous state

IoT Framework Cheat Sheet – Part 1 v1.0

Major IoT Standards

AllSeen	**Dec-2013** Microsoft, Qualcomm, Cisco, LG, Panasonic, Sharp, Haier, Electrolux
Thread	**Jul-2014** ARM, Big Ass Fans, Freescale, Nest, Samsung, Silicon Labs, Yale
Open Interconnect Consortium (OIC)	**2014** Atmel, Broadcom, Dell, Samsung, Intel, Wind River
Industrial Internet Consortium (IIC)	**2014** AT&T, Cisco, GE, IBM, Intel, Wind River

Core Framework

Advertisement and Discovery

About Announcement	Recommended mechanism for advertising, provide set of metadata about the application.
Well-known Name	A more primitive mechanism for announce and discover of applications, used by About Announcement.

Session and Port
After discovery process happens, connection is created using Session and Port.
A session can be either point to point or multi-point. It is created on specific port.

Bus Attachment
Mediate all connection to AllJoyn bus.

Bus Object
Is attached to a specific bus path (same object can be on different bus paths), implements a set of interfaces, allows a remote entity to call a method on local application.

ProxyBusObject
A symmetrical object to Bus Object, created by remote application to gain access to the BusObject.

Sessionless Signal
A mechanism to receive signals w/o having to manually create a session.

Instropection
To discover a remote AllJoyn application about its objects and object paths, full interface: methods, params, properties, and signals.

Events and Actions
To describe its signals and methods accordingly

Security
Occurs at the application level, as there is no trust at the device level. Using PIN code, PSK, or ECDSA. Message are encrypted with AES-128 CMM

AllJoyn Framework

AllJoyn is a collaborative **open-source software framework** that makes it easy for developers to write applications that can **discover nearby devices**, and **communicate with each other** directly regardless of brands, categories, transports, and OSes **without the need of the cloud** (the framework runs on the local network and does not require the cloud to function).

Transports	Wi-Fi, Ethernet, Serial, PLC, via a **Gateway Agent** (AllJoyn service)
Bindings	C, C++, Obj-C, Java
Platforms	RTOS, Arduino, Linux, Android, iOS, Windows, Mac, OpenWRT, Unity plug-in
Security	peer-to-peer encryption (AES128) and authentication (PSK, ECDSA)
Configurations	In XML format

The AllJoyn Session

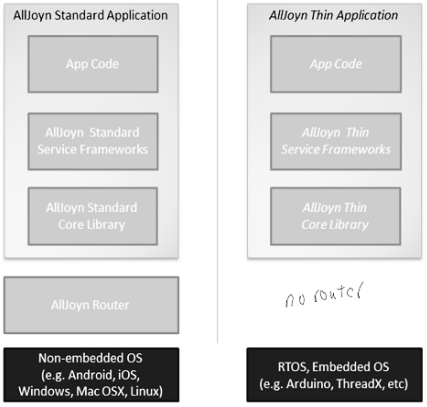
The AllJoyn requires "from" and "to" information to form a session.

TO Correspond to the targeted **service**	{session options, bus name, session port} - **session options**: how the data is exchanged, i.e. TCP or UDP. - **bus name**: the well-known name of corresponding bus attachment - **session port**: point of delivery "inside" the bus attachment. Eg., {reliable IP messages, org.alljoyn.samples.chat.a, 42}
FROM Correspond to the location of the **client** component	{session options, unique name, session ID} - **unique name**: the client's unique name - **session ID**: the client is assigned an ID when the connection is established. Eg., {reliable IP messages, org.alljoyn.samples.chat.a, :2.1, 1025}

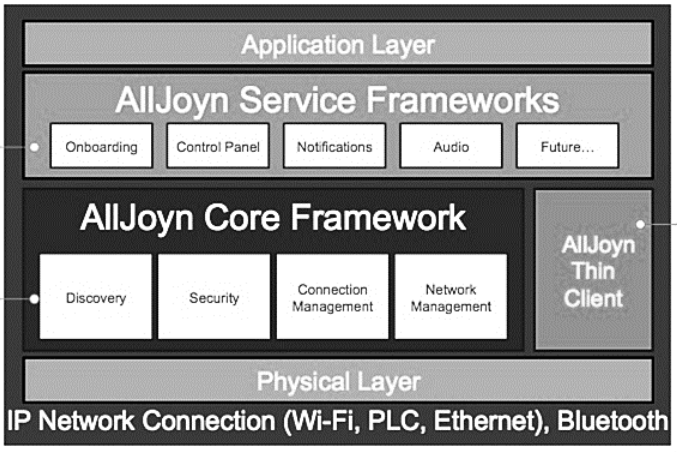
AllJoyn Architecture

AllJoyn framework comprises:
- **AllJoyn Apps**, and **AllJoyn Routers**.
- App talks to Router and vice versa, **App can't talk to App**.

Bundled Router	App uses its own Router, usually Android, iOS, Windows, or OSX apps.
Standalone Router	Multiple apps on the same device share one Router.
"Remote" Router	App uses Router on a different device, usually apply for embedded devices with little or no resource.

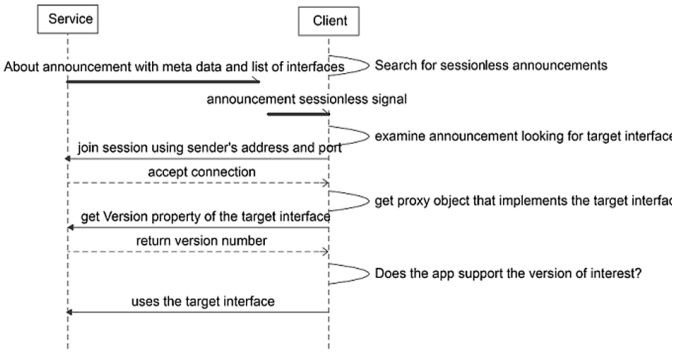


AllJoyn Core Library	Lowest level APIs to interact with AllJoyn network, such as **Security**, **Network**, **Session**, **Messaging / Routing**
AllJoyn Service Framework	Implements a set of common services, like **onboarding**, **notification**, **configuration**, or **control panel**.
AllJoyn App Code	Application logic, can access both Service Framework or Core APIs.

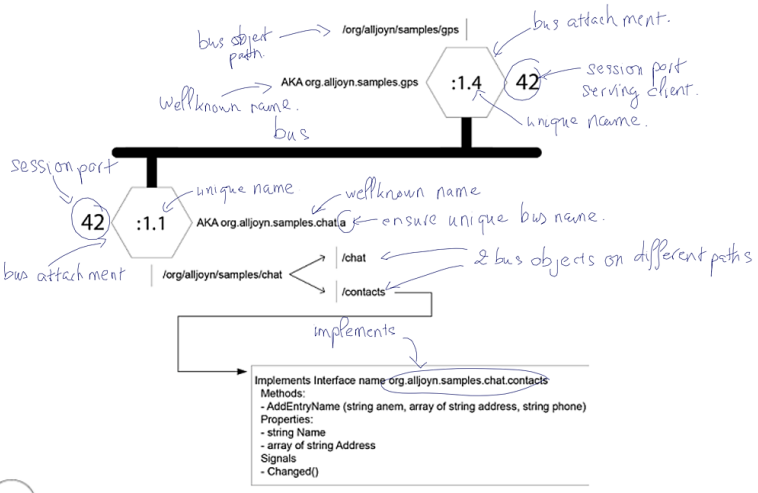


IoT Framework Cheat Sheet – Part 2 v1.0

Discovery Call Flows



Bus and Bus Attachment



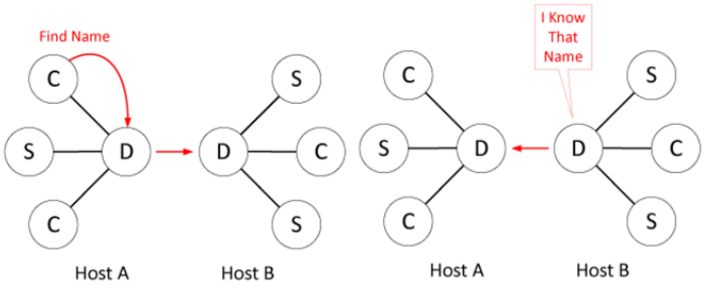
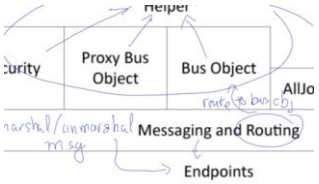
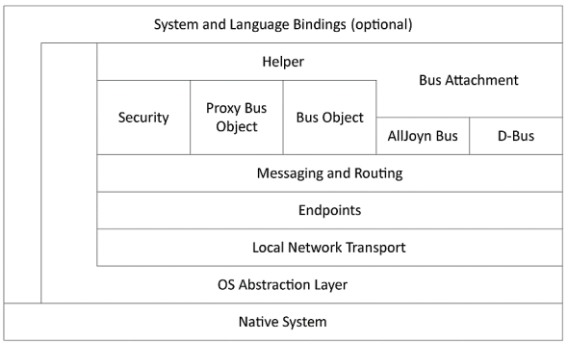


Figure: Client requests to Find Name

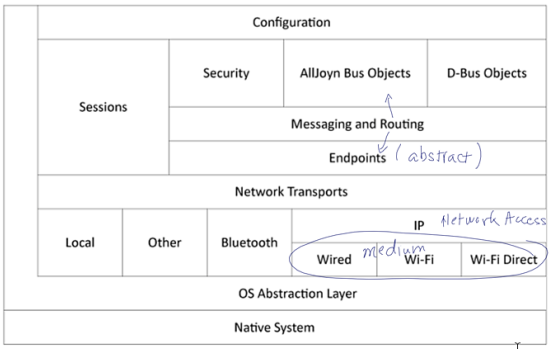
Figure: Router reports Found Name



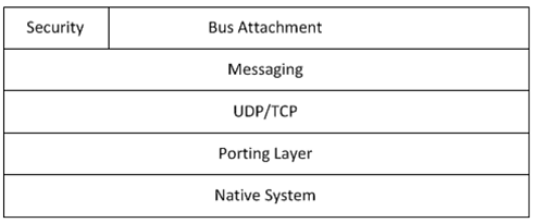
AllJoyn Standard Core Library Layering



AllJoyn Standard Core Library Router Layering



AllJoyn Thin Core Library Layering



IoT Framework Cheat Sheet – Part 3 v1.0

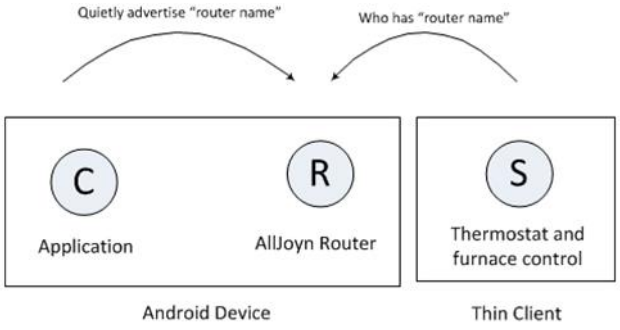


Figure: Thin Core Library router discovery

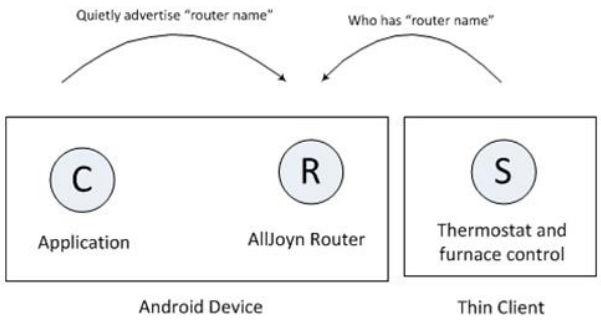


Figure: Thin Core Library router discovery

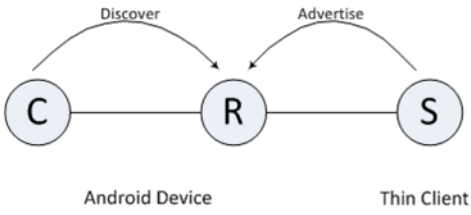


Figure:: Service discovery with the Thin Core Library

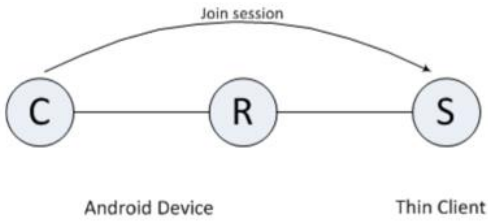


Figure: Android device joins session with service on the Thin Core Library

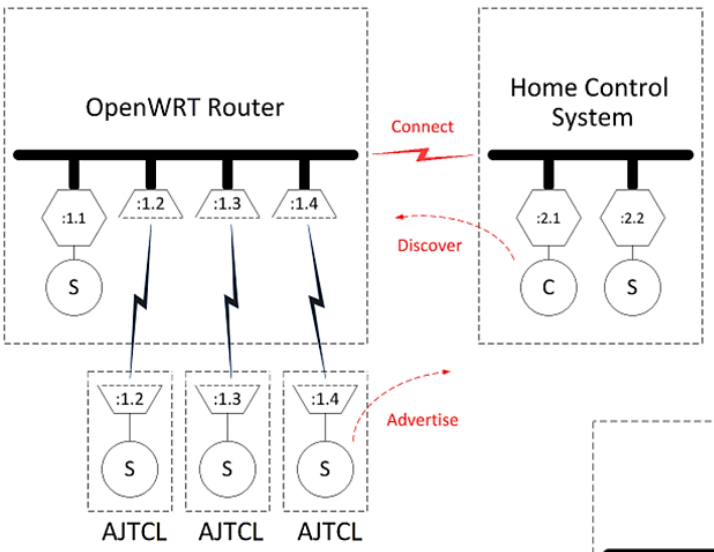


Figure: OpenWRT router, Thin Core Libraries, and home control system

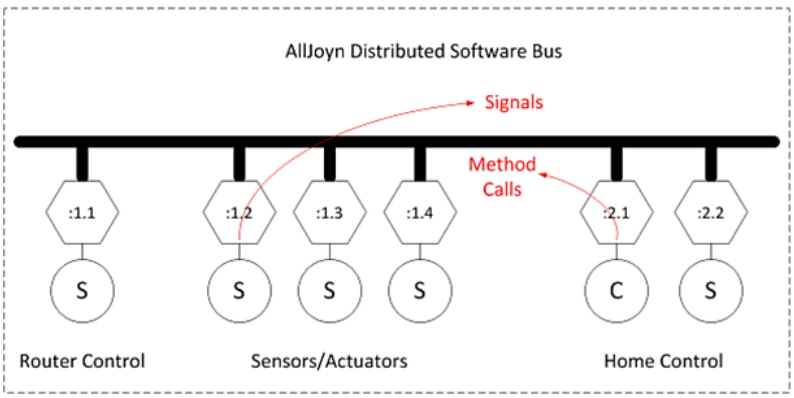


Figure: AllJoyn distributed software bus

IoT Framework Cheat Sheet – Part 4 v1.0

AllSeen vs OIC

- AllSeen
  - Focuses on device-to-device and starts at the home extending outward.
  - Additional gateway agent, a bridging technology
  - completely open and will never require a specific vendor to implement
  - focuses on product-to-product communications
  - doesn't require a cloud in the middle.
- OIC was born out of business applicability of IoT extending into consumer.

Table 3: Selected Internet of Things Protocols

PROTOCOL	MQTT	CoAP	XMPP	HTTP/RESTful
Transport	TCP/IP	UDP	TCP/IP	TCP/IP
Messaging	Publish/ Subscribe; Request/ Response	Request/ Response	Publish/ Subscribe; Request Response	Request/ Response
Cellular Suitability (1000s nodes)	Excellent	Excellent	Excellent	Excellent
Low Power and Lossy Network (LLN)	Fair	Excellent	Fair	Fair
Primary Orientation	Message	Web service/ Document	Message	Web service/ Document
Energy/ power needs	Low	Low	High	High
Key scenarios	Lightweight and embedded devices; unreli-able connections	Field; state-transfer; platform/network	Mass scale; persistent connections	Home and office

Source: Arlen Nipper, Graham Churchill, Electronic Design[18], Cisco Blogs[19]

IoT Misc – Part 1 v1.0

IoT and Bluetooth related RFCs	
7668	IPv6 over BLUETOOTH(R) Low Energy
4944	Transmission of IPv6 Packets over IEEE 802.15.4 Networks
6282	Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks
6775	Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)
6574	Report from the Smart Object Workshop
7228	Terminology for Constrained-Node Networks
7390	Group Communication for the Constrained Application Protocol (CoAP)
7400	6LoWPAN-GHC: Generic Header Compression for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)
7554	Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement
7641	Observing Resources in the Constrained Application Protocol (CoAP)

802	Overview	Description
802.3	Ethernet	"Grandaddy" of the 802 specifications.
802.15	Wireless Personal Area Networks	Communications specification that was approved in early 2002 by the IEEE for wireless personal area networks (WPANs).
802.15.1	Bluetooth	Short range (10m) wireless technology for cordless mouse, keyboard, and hands-free headset at 2.4 GHz.

	ATtiny2313	ATtiny84	ATtiny85	ATmega328	ATmega2560
Cost	$3.13	$3.53	$2.65	$3.70	$16.75
Pin Count	20	14	8	32	100
Max IO pins	18	12	6	23	86
Flash Memory	2 Kb	8 Kb	8 Kb	32 Kb	256 Kb
EEPROM	128 bytes	512 bytes	512 bytes	1 Kb	4 Kb
ADC channels	0	8	4	6 (8 on SMD)	16
PWM channels	4	4	6	6	15
Timers	1 8bit 1 16bit	1 8bit 1 16bit	2 8bit	2 8bit 1 16bit	2 8bit 4 16bit
SRAM	128 bytes	512 bytes	512 bytes	2 Kb	8 Kb
Hardware Serial	No	No	No	Yes – 1	Yes – 4
Hardware I2C	No	No	No	Yes	Yes
External Interrupts	2 (8 PCINT)	1 (12PCINT)	1 (6 PCINT)	2 (23 PCINT)	8 (32 PCINT)