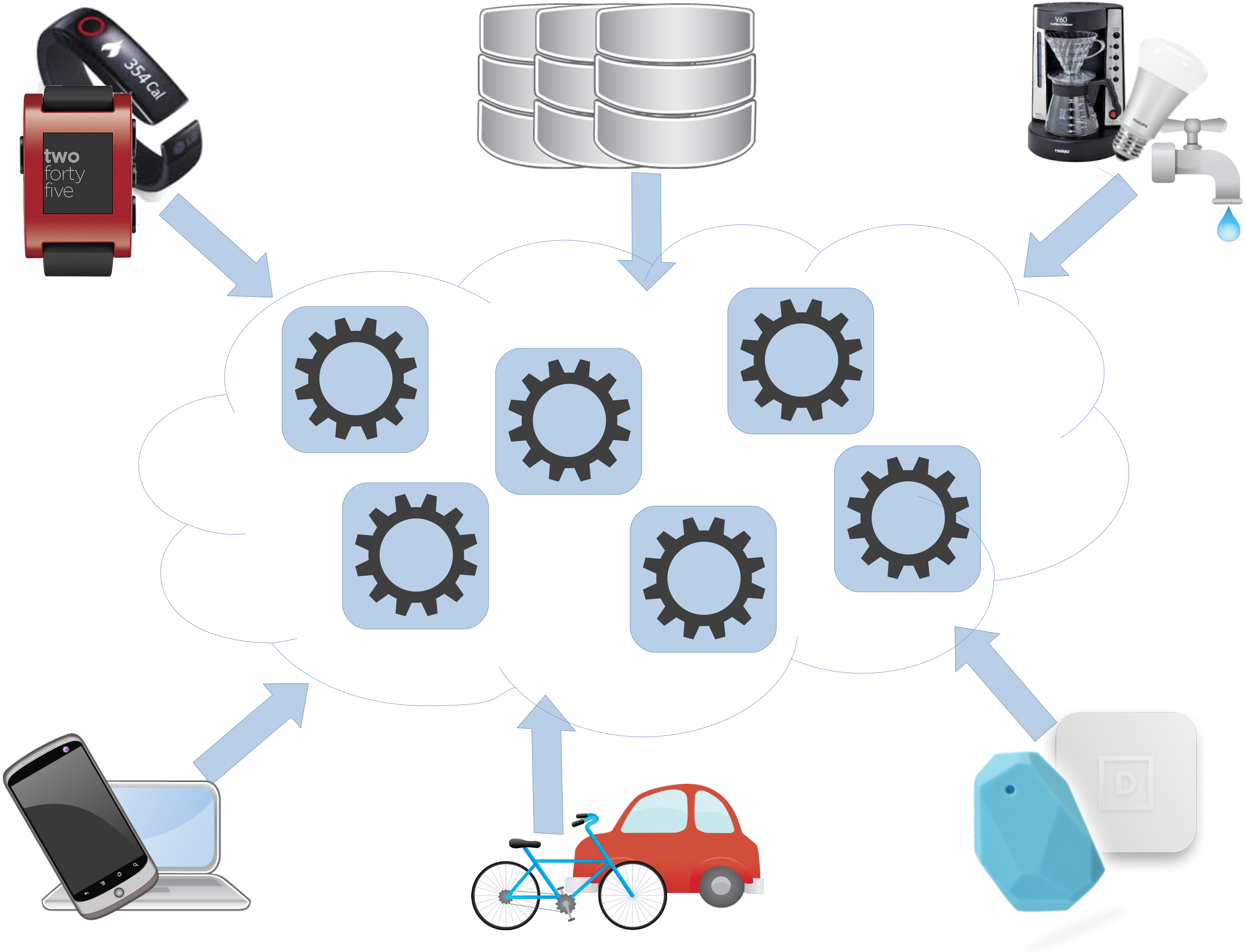
Beetle:

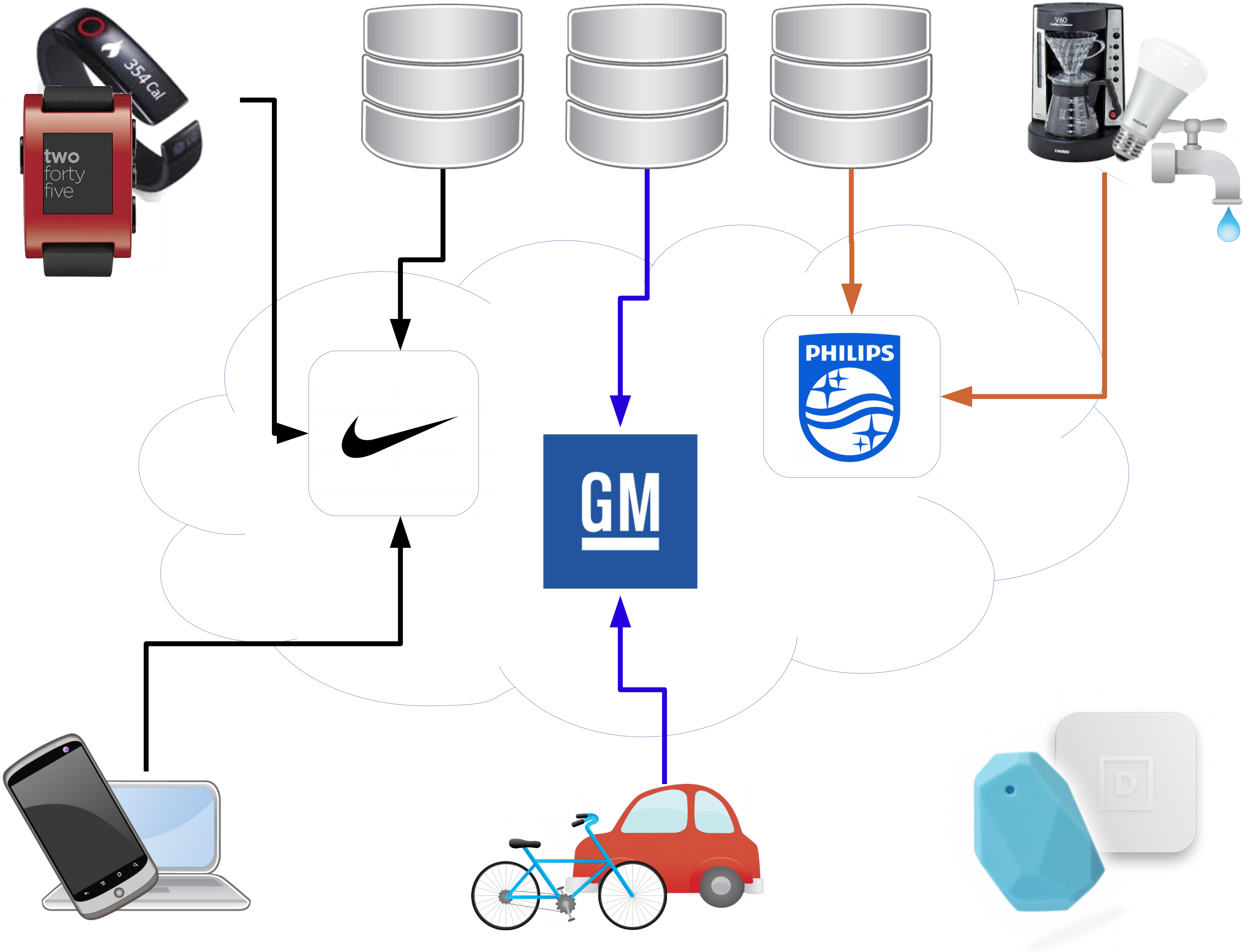
Many-to-many communication in Bluetooth LE

Amit Levy, Laurynas Riliskis,
Philip Levis, David Mazières, and Keith Winstein

The *ideal* Internet of Things


354 Cal
two
forty
five
V60
D

The Internet of Things *today*


354 Cal
two
forty
five
V60
HARIO
PHILIPS
PHILIPS
GM
D

It's Not An Internet

> “...connectivity is its own reward, and is more valuable than any individual application such as mail or the World-Wide Web.”
>
> - RFC1958, *“Architectural Principles of the Internet”*

- Vertical integration of peripherals, gateways, and cloud software
- Connectivity is poor and constrained
  - BLE edge devices cannot communicate with each other
  - A BLE edge device can communicate with only single mobile phone
- Simple, desirable use cases are impossible
  - Your smart watch displaying data from your heart monitor
- The *things* – BLE edge devices – are dumb and powerless
  - Architecturally prevented from anything except interacting with a mobile application

Outline

- Introduction
- Bluetooth LE architecture
- Beetle
  - Network architecture
  - Mechanisms:
    - HAT
    - Virtual devices
    - Service export control

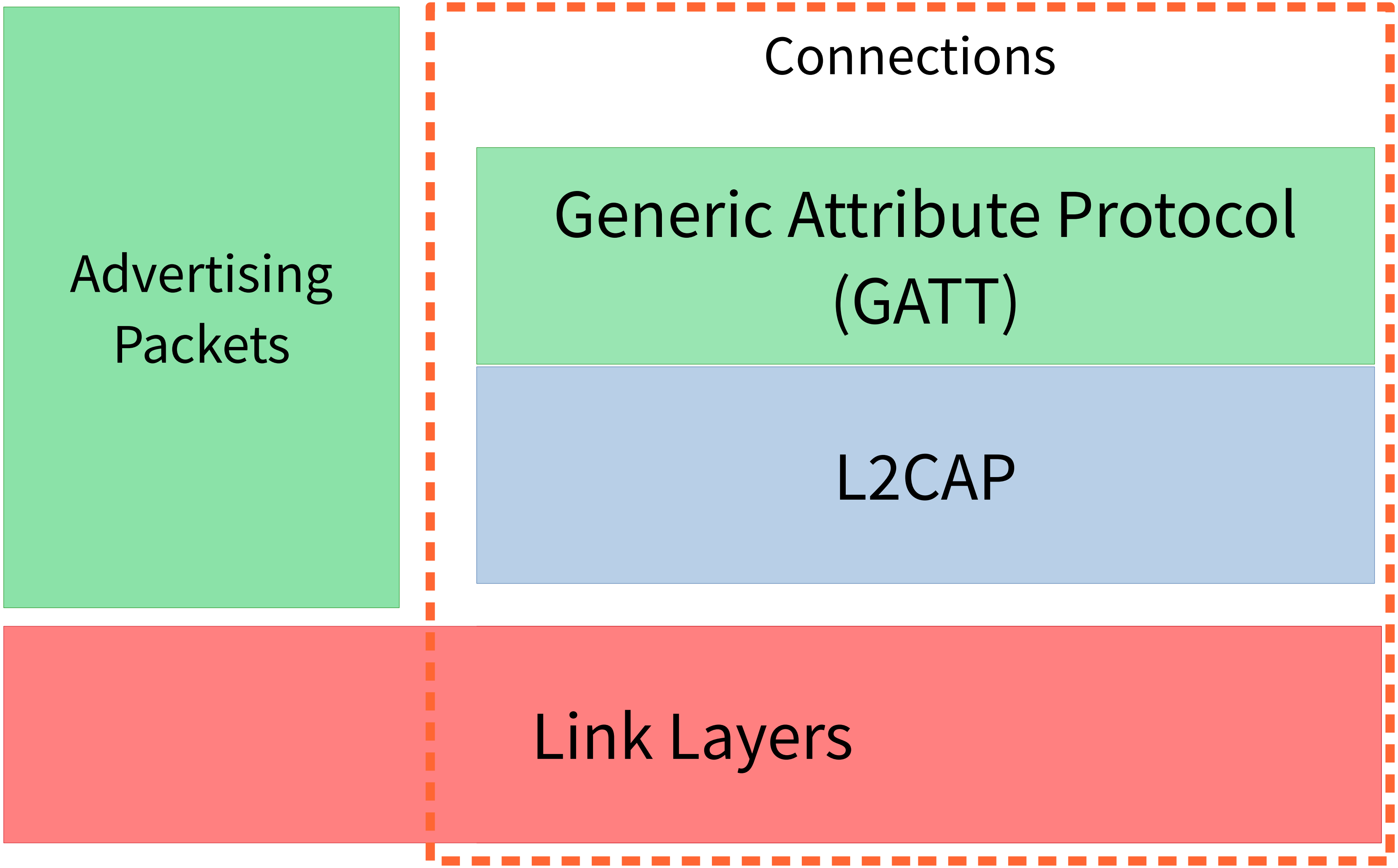
Outline

- Introduction
- Bluetooth LE architecture
- Beetle
  - Network architecture
  - Mechanisms:
    - HAT
    - Virtual devices
    - Service export control

Bluetooth Low Energy

- Single-hop protocol
- Physical, Link and Application layers
- Optimized for small exchanges and low energy:
  - ~24 byte exchanges; infrequently
  - µA power consumtpion
  - Can run for years on coin battery

Bluetooth Low Energy

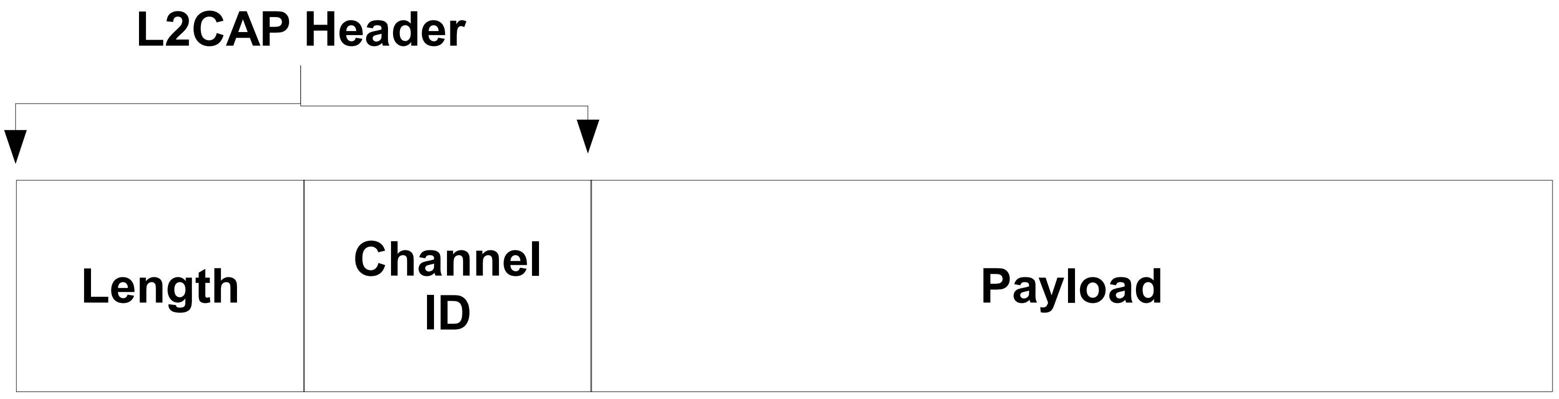


Link Layer

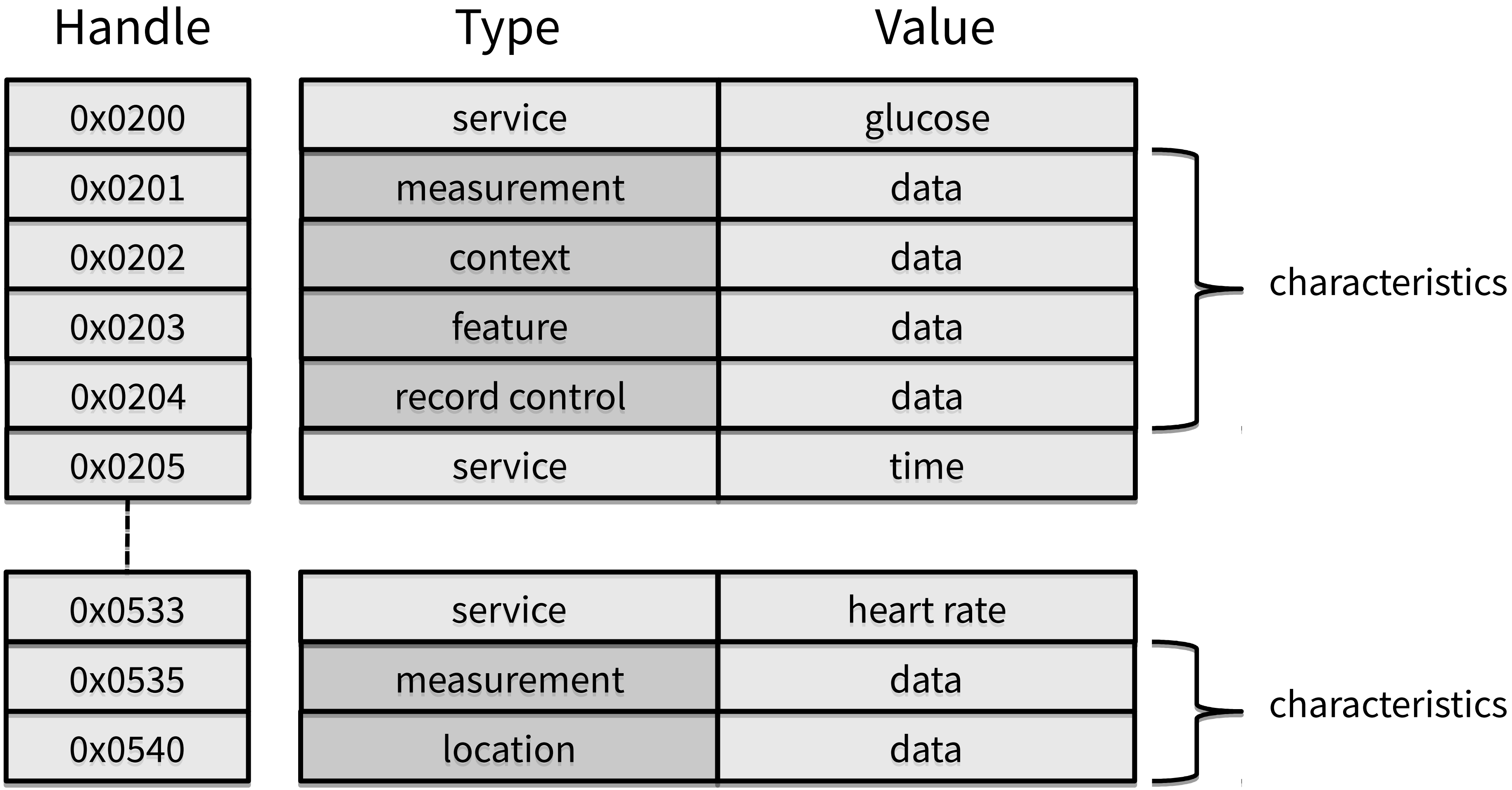
- “Piconet” topology
- Two roles:
  - Peripheral (fitness band, watch, dead-bolt, etc)
  - Central (smart phone, laptop, gateway, etc)
- Centrals manage connections with multiple peripherals
- Peripherals can connect to a *single* central only

L2CAP Channels

- Logical channels over single link
- Reliable
- Some channels reserved (e.g. GATT, signaling)



Generic Attribute Protocol (GATT)



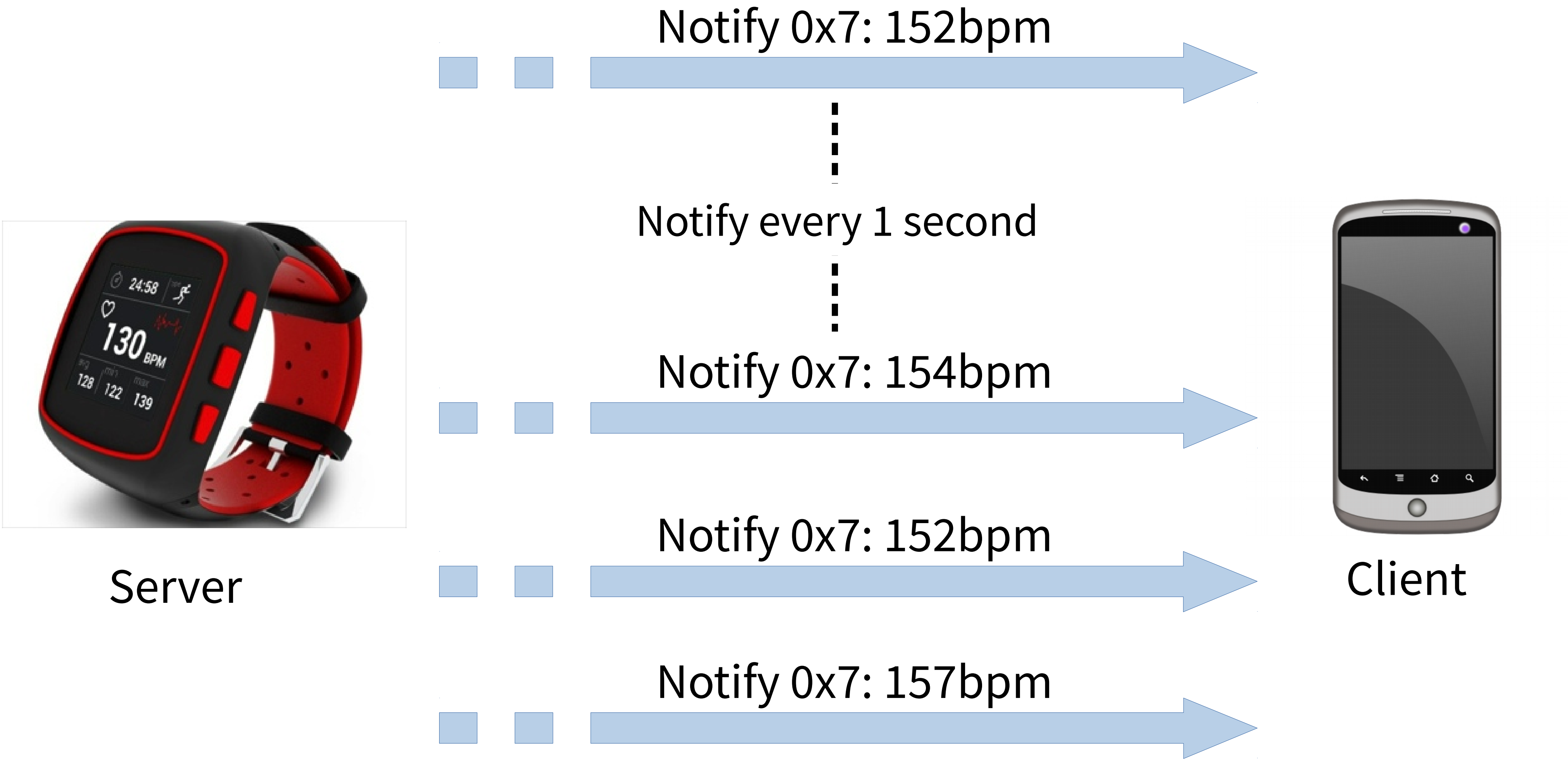
Handle	Type	Value	
0x0200	service	glucose	
0x0201	measurement	data	characteristics
0x0202	context	data	
0x0203	feature	data	
0x0204	record control	data	
0x0205	service	time	
0x0533	service	heart rate	
0x0535	measurement	data	characteristics
0x0540	location	data	

GATT

- Two roles:
  - Server has the attributes
  - Peripherals and Centrals can be both clients and servers simultaneously
- Key/Type/Value store:
  - Read/Write
  - Notify/Indicate
  - Find by type

Opcode	**Handle**	**Opcode parameters (type, value ...)**

GATT: Simple Example



Notify 0x7: 152bpm

Notify every 1 second

Notify 0x7: 154bpm

Notify 0x7: 152bpm

Notify 0x7: 157bpm

Server

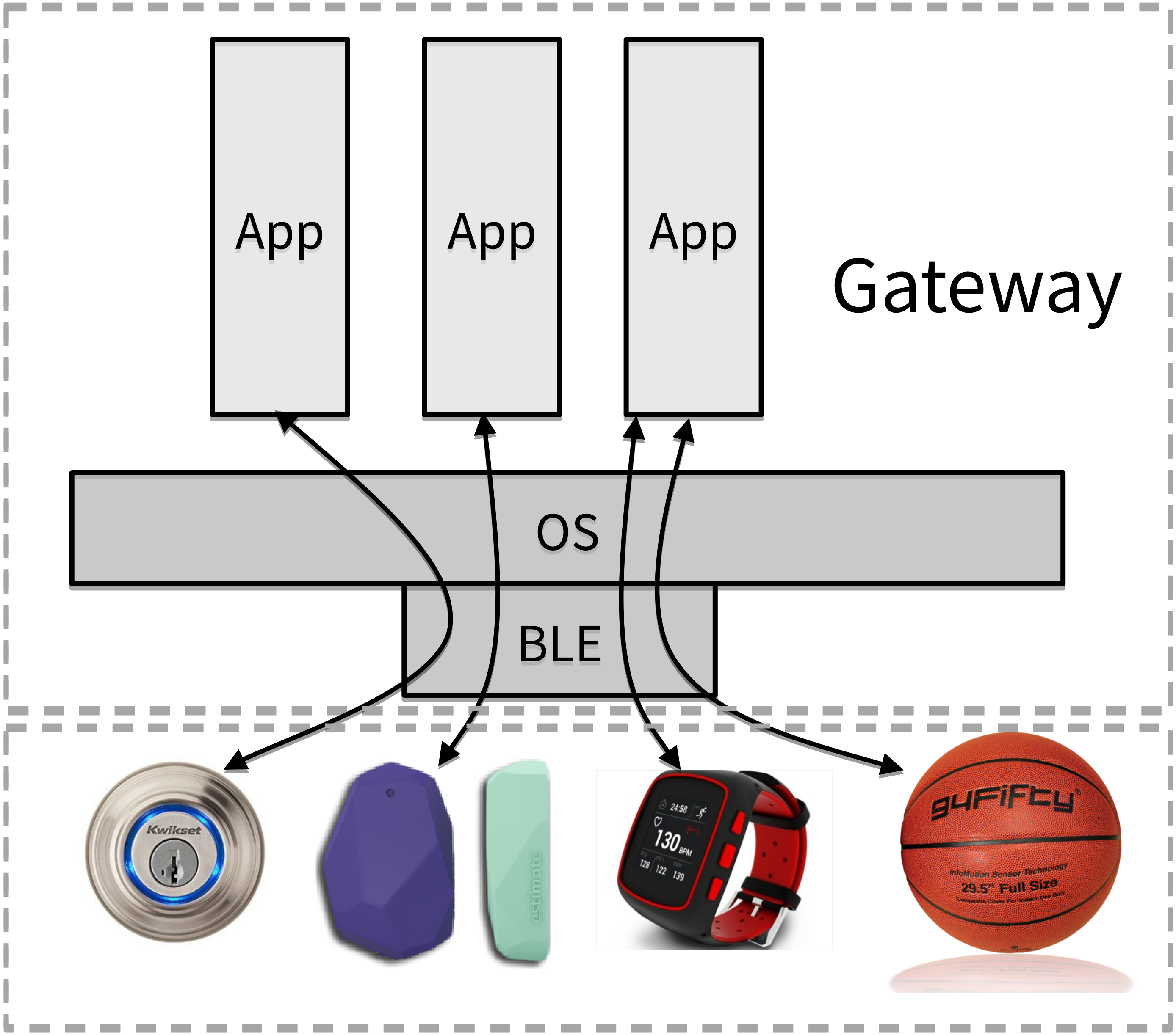
Client

GATT

- Interoperable:
  - Standardized service/characteristic types
  - Incorporates service discovery
- Transactional
  - Only onle outstanding command per connections in each direction
- High level
  - Many chips expose *only* GATT to embedded programmers

A peripheral can only maintain one open connection![*]

One-to-One Communication

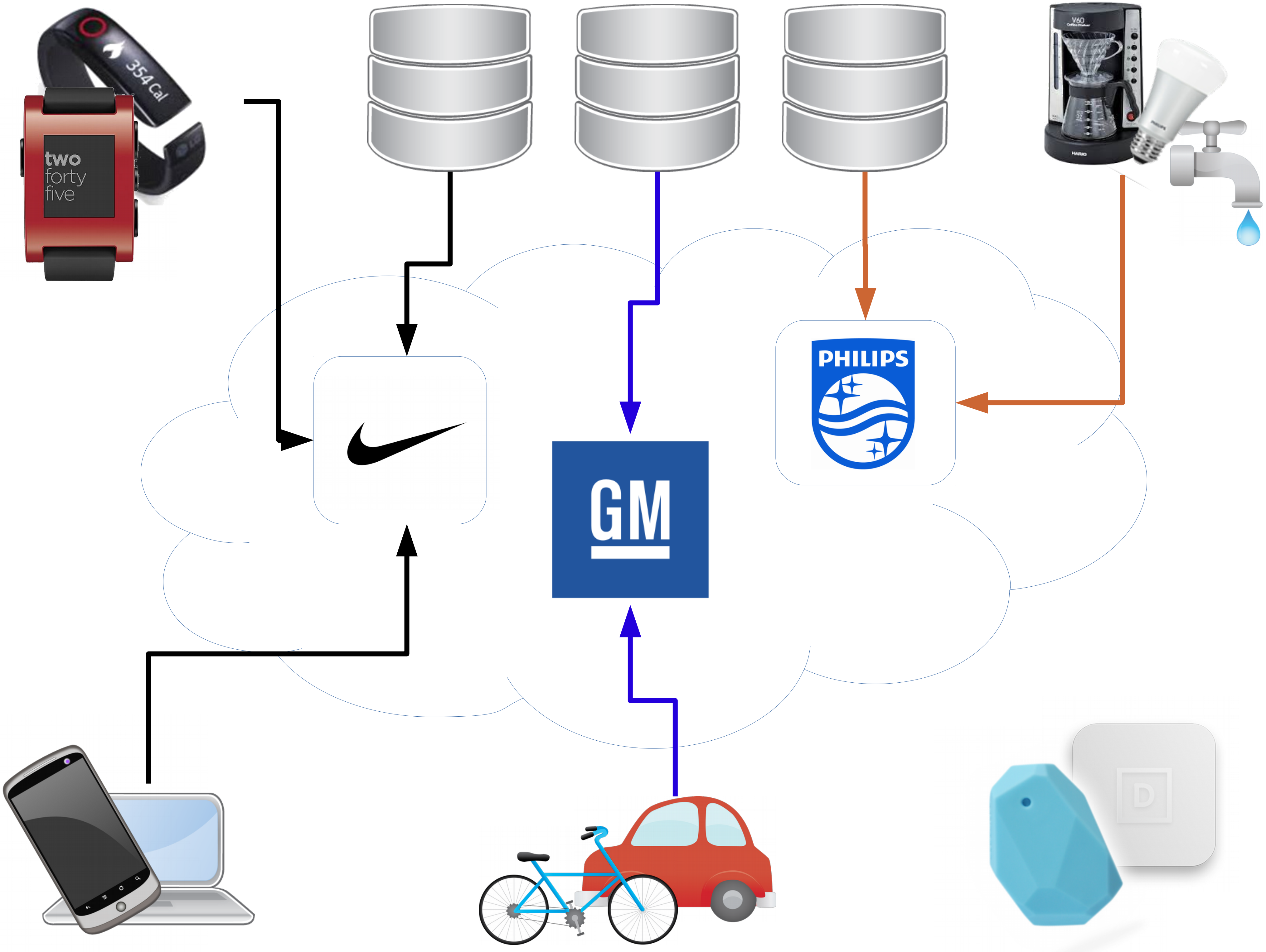


Today: Gateway Interposes on Data

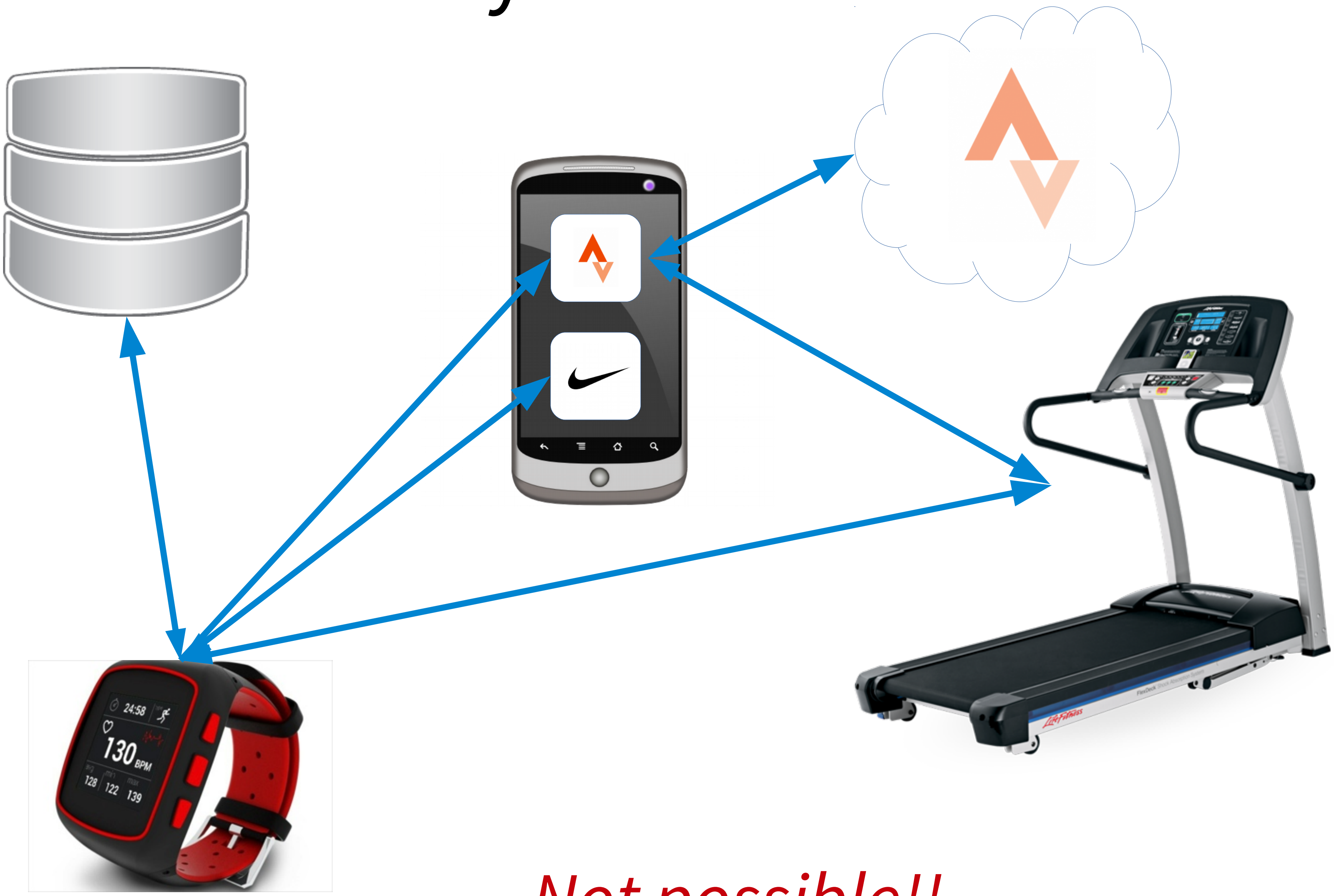
- Each peripheral connects to a single app on the gateway
  - Can only communicate directly with that app
- App consumes GATT data. Mediates only supported interactions:
  - Issue GATT commands to other connected peripherals
  - Proprietary protocol to servers (e.g. over app-specific HTTP)
  - (Limited) Intent-based interface to other apps
- The app doesn't support an interaction you want?
  - Tough luck...

Bluetooth LE Limitations

- BLE is a link *not* a network
- Not currently possible:
  - Peripheral-to-peripheral
  - Multiple applications & one peipheral
  - Peripheral-to-cloud
- Result is walled gardens


354 Cal
two
forty
five
V60
PHILIPS
GM

Why is this bad?



Not possible!!

Outline

- Introduction
- Bluetooth LE architecture and applications
- Beetle
  - Network architecture
  - Mechanisms:
    - HAT
    - Virtual devices
    - Service export control

Beetle

- Builds a *network* out of BLE
  - Peripherals can communicate with one another
  - Multiple applications can (safely) use a peripheral
  - Peripherals can interact with broader Internet
- A software layer that runs on your gateway (phone), adding three mechanisms
  - Handle address translation (HAT) for multi-link networking
  - Virtual devices for software and IP networking
  - Service export control for securely managing this greater connectivity
- Completely backwards compatible with existing BLE devices

Beetle: Design Overview

- Gateway bluetooth daemon
  - Manages all BLE links to the gateway
- Provides networking to BLE devices as OS service on the gateway (i.e. smart phone)
- Gateway routes between peripherals, apps and cloud
  - Gateway *does not* interpose on data
- Leverage richer user-interface on gateway to manage routing and security policies

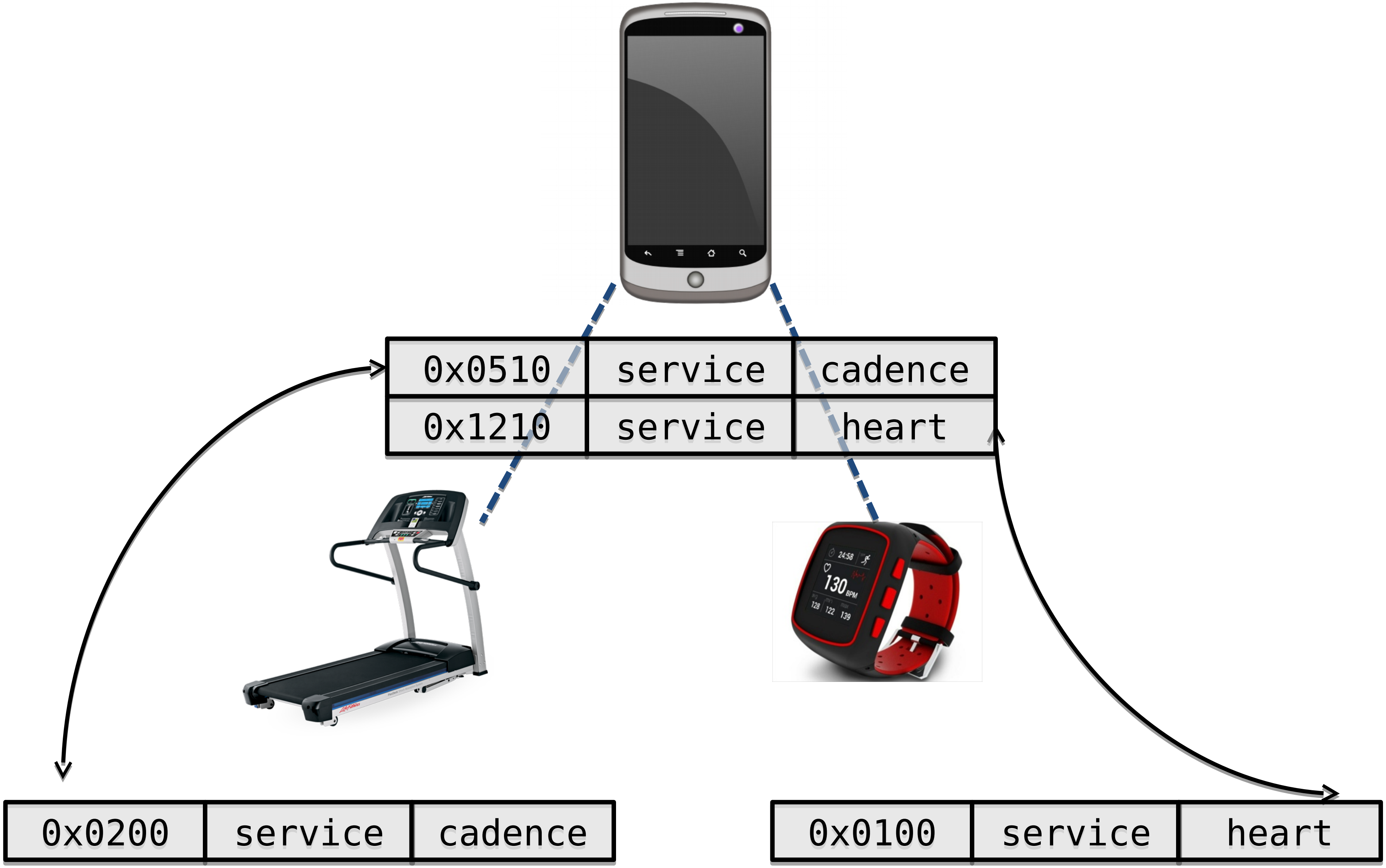
Beetle: Gateway Mechanisms

- Handle address translation (HAT)
  - Multi-link networking
- Virtual devices
  - Software connectivity
  - Interface with other protocols (e.g. HTTP, Intents)
- Service export control
  - Manage security policies in the face of greater connectivity

Handle Address Translation (HAT)

- Re-export peripheral services as gateway services
- Proxied attributes on the gateway
  - Associated with a remote attribute on a peripheral
  - Beetle routes messages to proxied attributes to the appropriate peripherals
- Translate peripherals handles into gateway address space
  - Similar role to NAT in TCP/IP world

Handle Address Translation (HAT)



0x0510	service	cadence
0x1210	service	heart

0x0200	service	cadence

0x0100	service	heart

HAT: Handle Allocation

- Ensure that grouped attributes appear together in the gateway address space
- Global handle address space
  - Attributes appear as same handle to all peripherals
  - Would allow exchange of handles between peripherals
  - Unlikely, but possible, address space exhaustion
  - Leaks some information
- Separate handle address space for each BLE connection
  - Allocation can be more efficient; can deal with reallocation better
  - More scalable if high degree of connectivity is common
  - Peripherals cannot exchange handles in data packets

HAT: Discovery

- Typical BLE connection has fixed set of services
- In Beetle, new services appear as more peripherals connect or policy is changed
- Take advantage of “Service Changed” characteristic
  - Notifies client when new set of services changes
  - Provides a range of affected handles
- Keep track of which peripherals might notice the service has changed to minimize noise
  - If a peripheral never asks for a service, it shouldn't matter

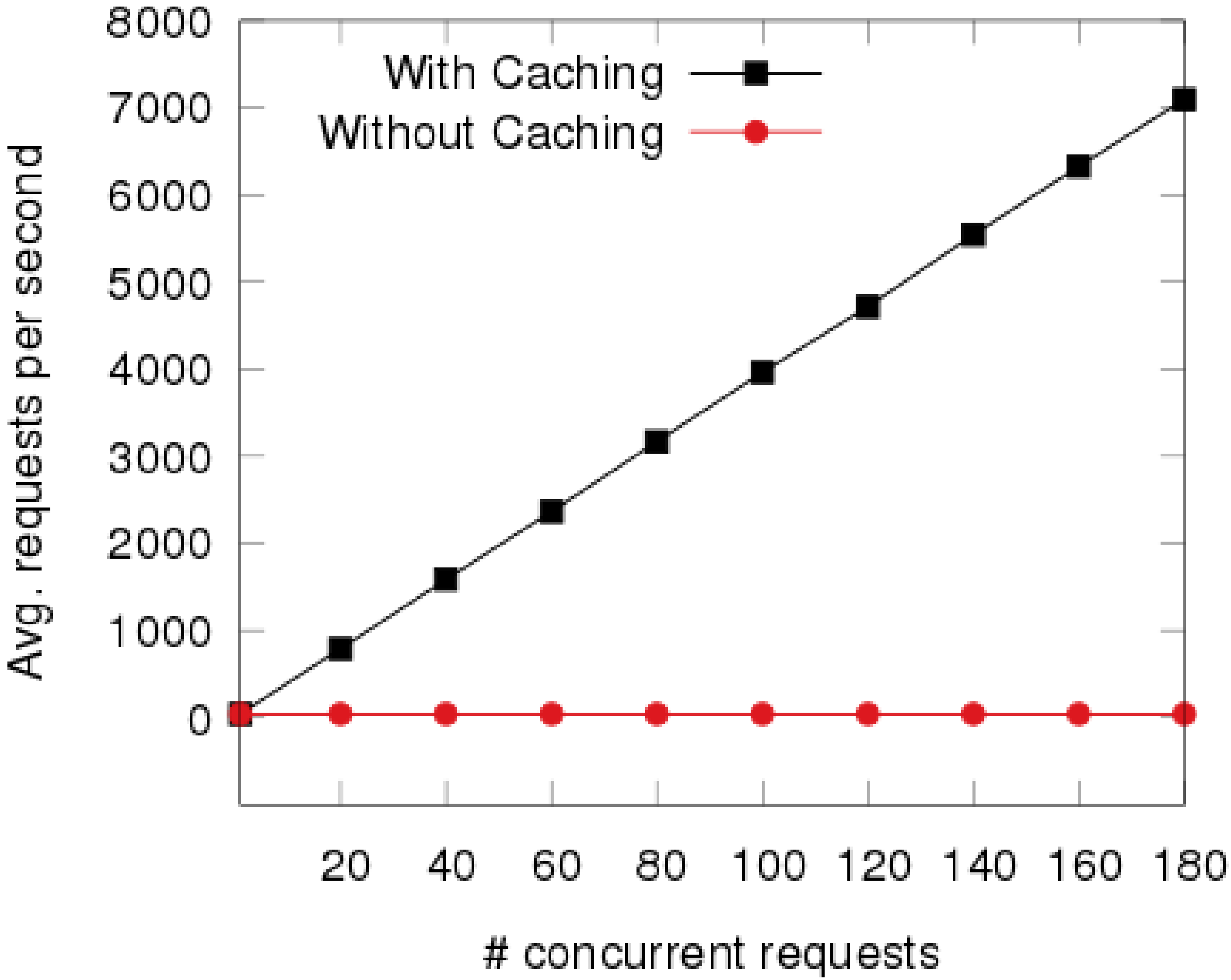
HAT: Notifications

- GATT notifications are a two-step process:
  - *Subscribe/unsubscribe* to notification by writing 1 or 0 to an attribute
  - Server begins notifying when value changes
- Cannot re-expose subscription attribute directly
- Instead:
  - Maintain a subscription set for every server notification source
  - Intercept *subscribe* and *unsubscribe* messages
  - Only forward first *subscribe* or last *unsubscribe* to server

HAT: Characteristic Caching

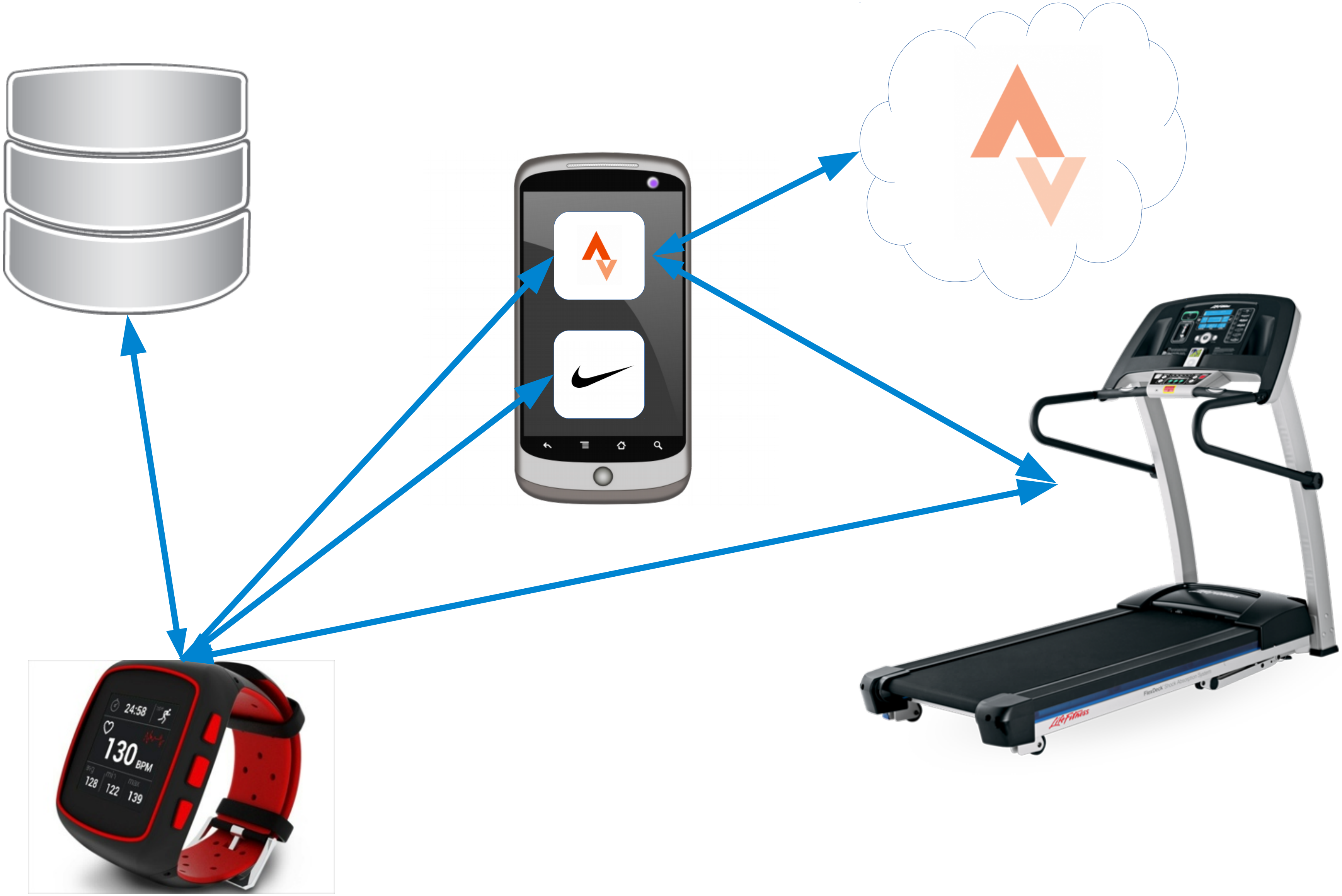
- Recall: GATT is transactional
  - Cannot issue two commands concurrently over same connection
  - How do we scale to many clients?
- Cache read values on gateway for one connection interval
- Optional “characteristic descriptor” allowing server to control cache
- Each client gets same performance if it were the only client

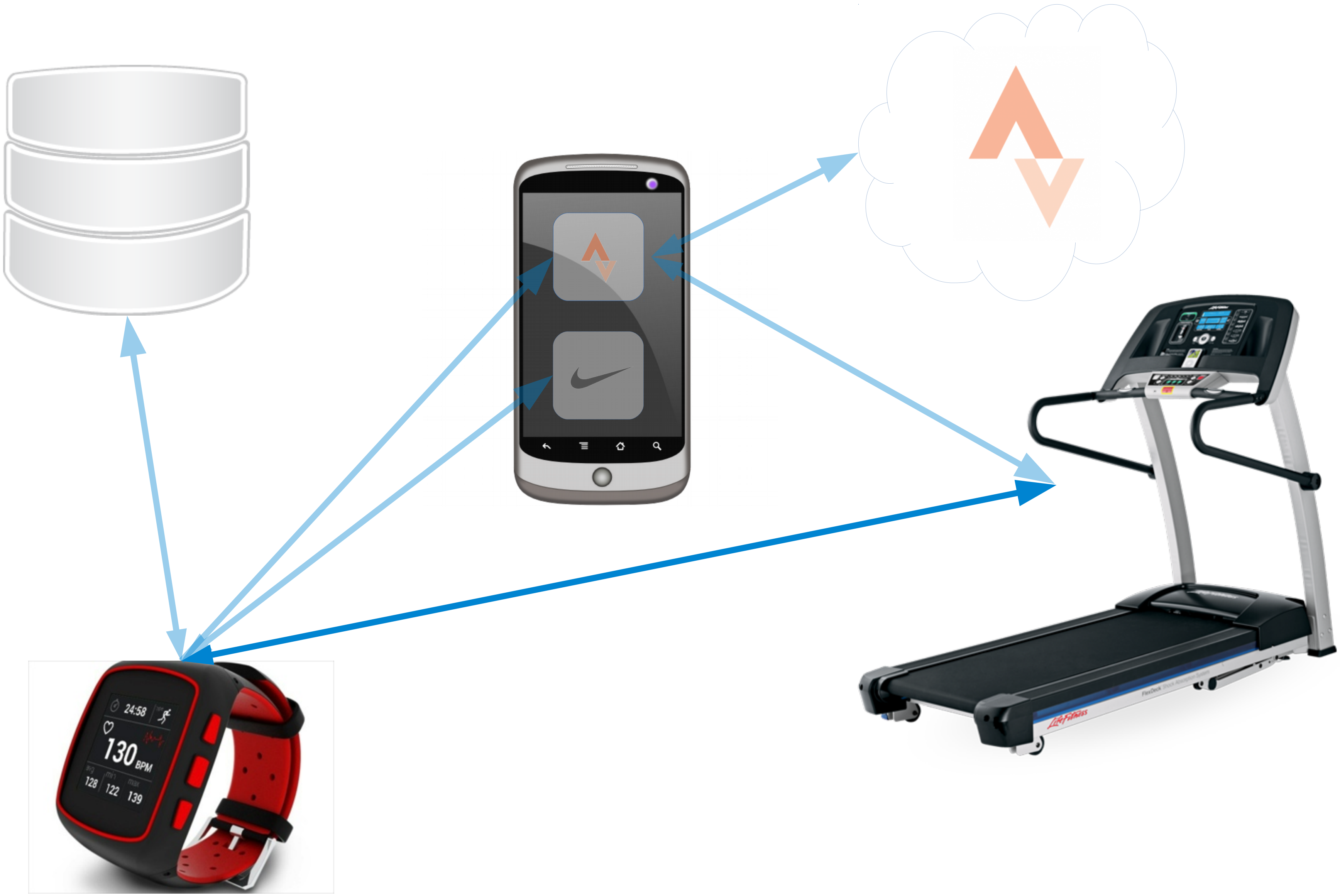
HAT: Characteristic Caching



HAT Creates a *Network*

- Re-exporting attributes on gateway enables peripheral-to-peripheral communication
- Aggregating attributes from multiple servers allows many-to-many peripheral communication
- HAT must maintain app-level protocol semantic
- Leverage knowledge of app-level protocol semantics to retain reasonable performance


24:58
130 BPM
128
122
139


24:58
130 BPM
128
122
139
LifeFitness

Virtual Devices

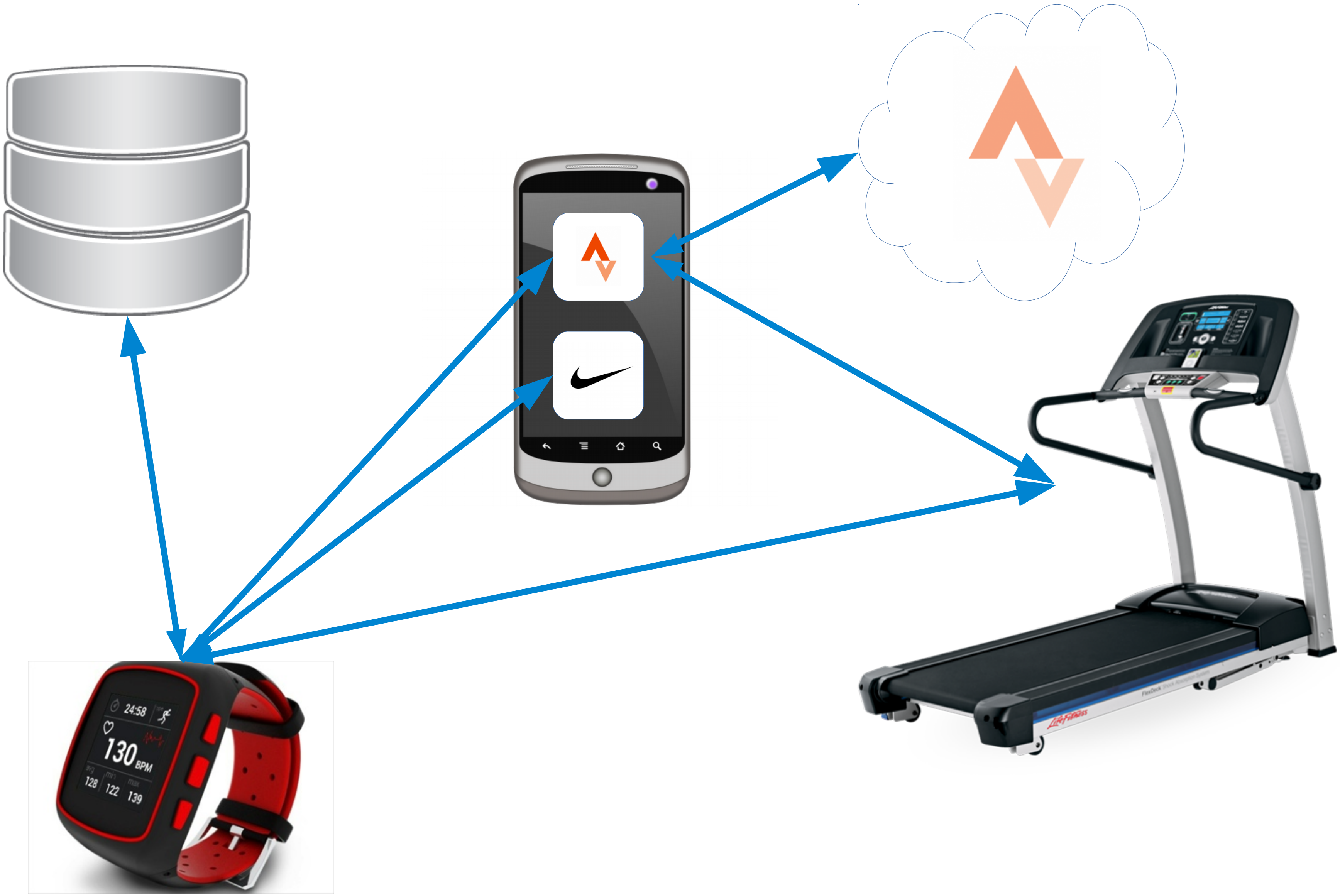
- Virtual devices speak GATT for non-BLE links:
  - IPC, TCP/IP, USB, etc
- Provide access to non BLE services
  - GPS
  - Emulated device with test data
  - Legacy Internet services (e.g. HTTP)
- Complexity handled by HAT

Virtual Devices: Local

- A user-level process that speaks GATT
- Access to Beetle over IPC (e.g. UNIX domain sockets)
- Similar to programming an app now
- Very useful:
  - Multiple user apps
  - Expose local, non-BLE, sensors
  - Prototyping hardware
  - Custom multiplexing

Virtual Devices: Network Services

- Virtual devices can exist on the Internet
  - In the cloud, local area network
- Scenario 1: Internet service supports Beetle
  - Beetle OS service connects directly over TCP
  - Don't need to write a tailored app
- Scenario 2: Legacy Internet service (e.g. HTTP/REST)
  - A local virtual device exports data over the legacy protocol

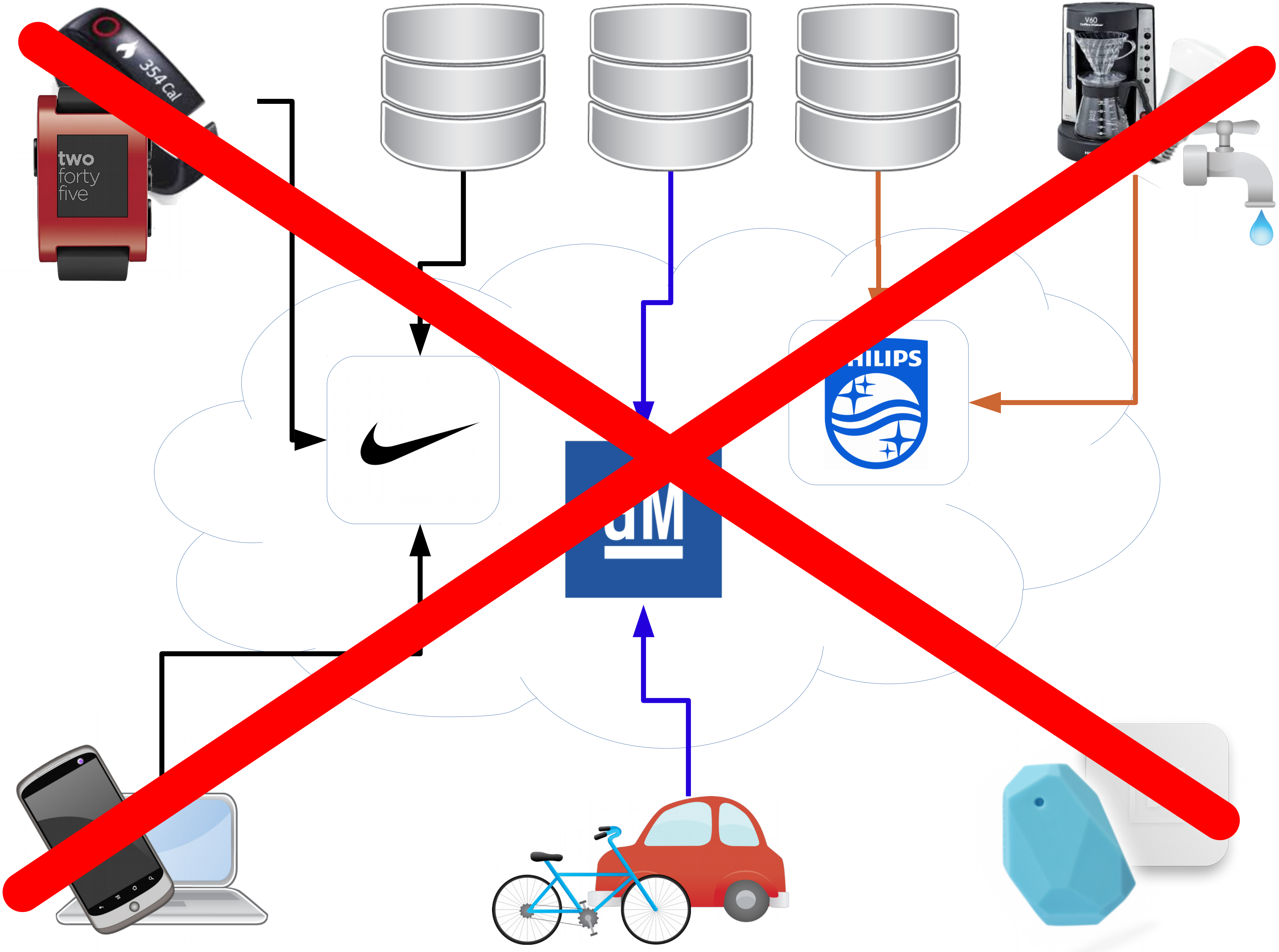

24:58
130 BPM
128
122
139

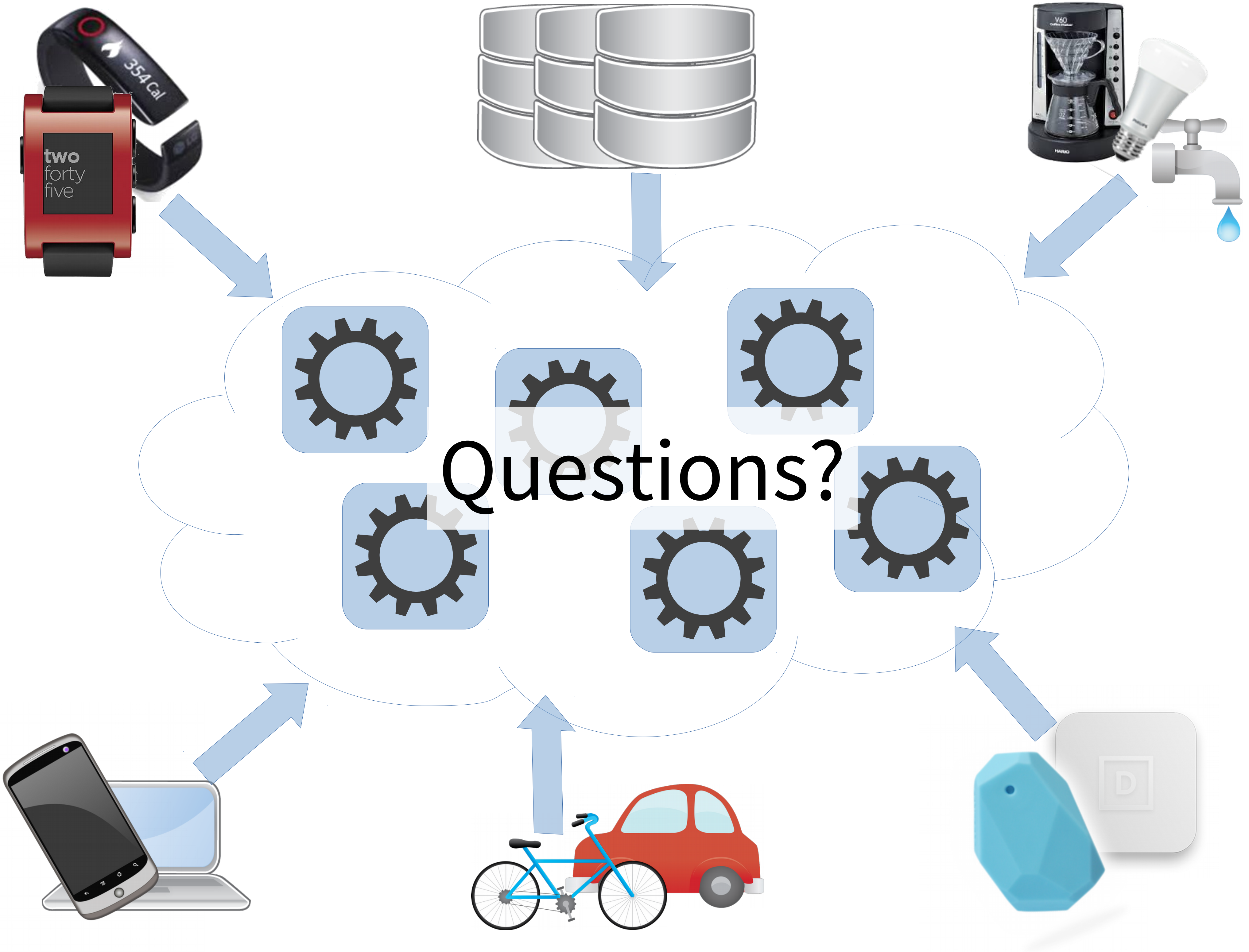
Service Export Control

- So much connectivity!!
- Need a way to control who sees what
  - Strava shouldn't only see my current heart rate when I allow it
- Routing at app-level protocol gives us more flexibility
- Many possible criteria for access control
  - Physical location
  - Identity
  - Pre-established trust
  - Out-of-band authentication (e.g. user login)

Beetle

- Gateway should route communication but not mediate application data
- Beetle is an OS service on the gateway that creates a network from BLE
- Three key mechanisms:
  - HAT for peripheral communication
  - Virtual devices for multiple-apps, device emulation and connecting other networks
  - Service export control pushes policies to more featureful gateway devices
- Completely backwards compatible with existing BLE devices


354 Cal
two
forty
five
V60
HILIPS
GM



Questions?