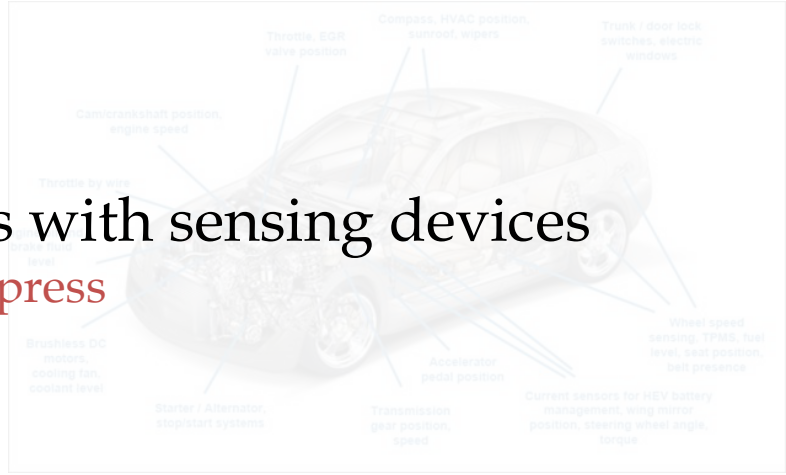
Ending Monolithic Apps for Connected Devices

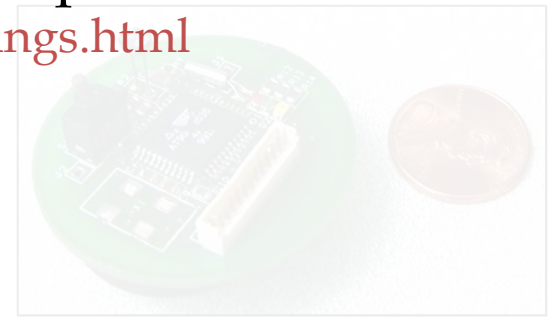
Rayman Preet Singh (Univ. of Waterloo),
Chenguang Shen (UCLA),
Amar Phanishayee, Aman Kansal, Ratul Mahajan (Microsoft Research)

Growth in Sensing Devices



In 2017, **90 million** homes with sensing devices
https://www.abiresearch.com/press

50 billion connected *sensing devices* by 2020
In 2008, #sensing devices > #people
https://share.cisco.com/internet-of-things.html



Homes and Buildings Automobiles

Personal Emerging

Sample App: Life Logging

Quantified Self App

Development Tasks

Sensor driver
Device discovery
Inference algorithms
Parameter tuning

Structure data processing (e.g., cloud service *if needed)*

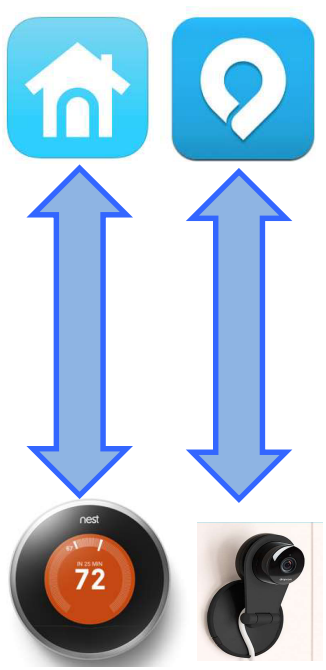
User mobility
Device disconnections and *failure*

User interface and functionality

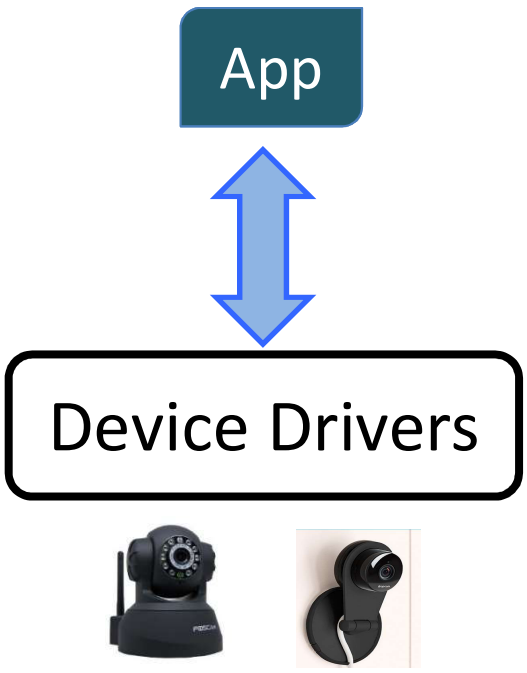
Existing Approaches

Monolithic

Nest Dropcam



Device Abstractions



Existing Approaches

Development Tasks	Device Abstractions
	HomeOS, HomeSeer, Revolv, …
Sensor driver Device discovery Inference algorithms Parameter tuning	✓
Structure data processing	
User mobility Device disconnections and failure	

Existing Approaches

Development Tasks	Device Abstractions	Mobile Sensing	Stream processing	Macro-programming
	HomeOS, HomeSeer, Revolv, …	Kobe, Auditeur, Senergy, …	Semantic streams, Task cruncher	Sensorware, Kairos, Envirosuite, …
Sensor driver	✓			✓
Device discovery				✓
Inference algorithms		✓	~	
Parameter tuning		✓		✓
Structure data processing		~		
User mobility				
Device disconnections and failure				

~ denotes partial fulfillment

Insight

Ease app development by *decoupling* **apps** and **devices** using an inference framework

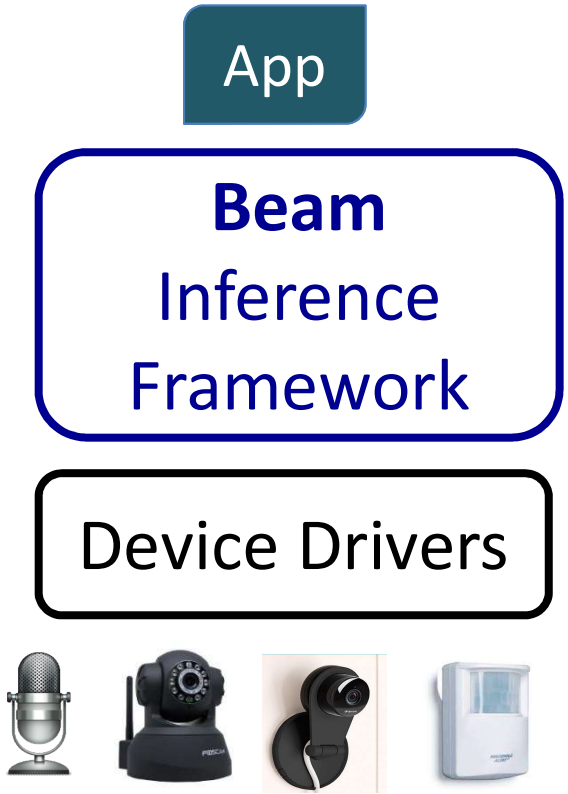
Framework performs device and algorithm selection

Decouple *Inferences,* Apps, and Devices

Monolithic

Nest Dropcam

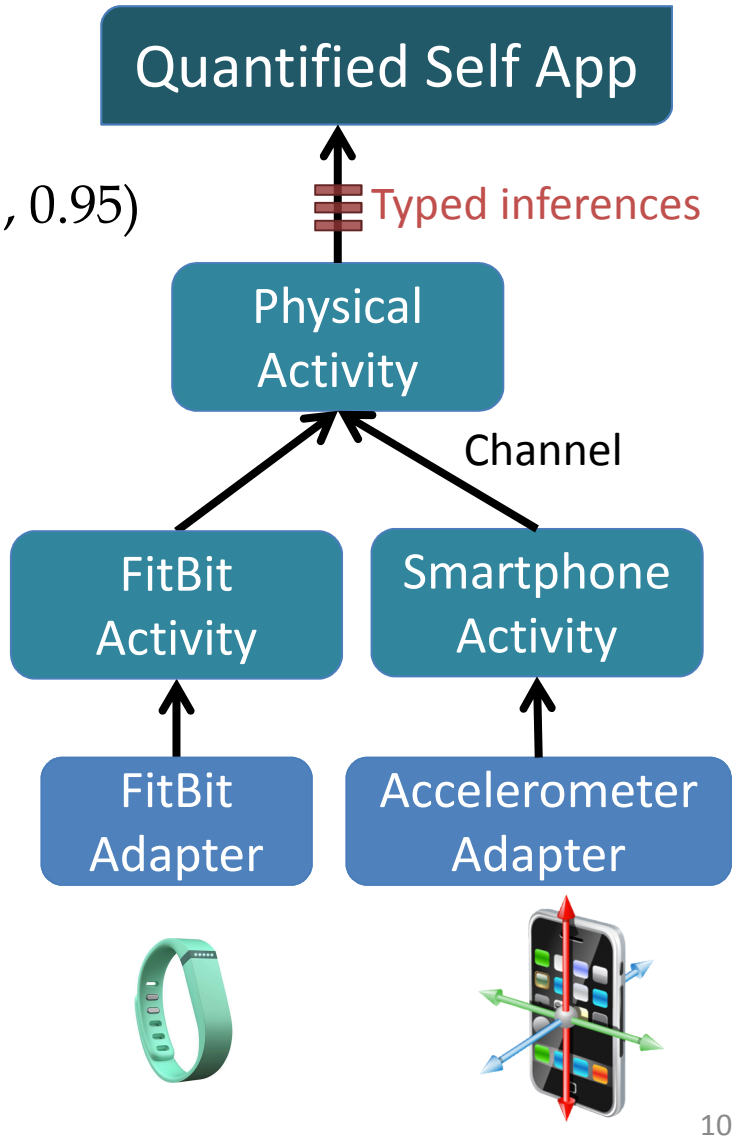
Device Abstractions



Beam Inference Framework

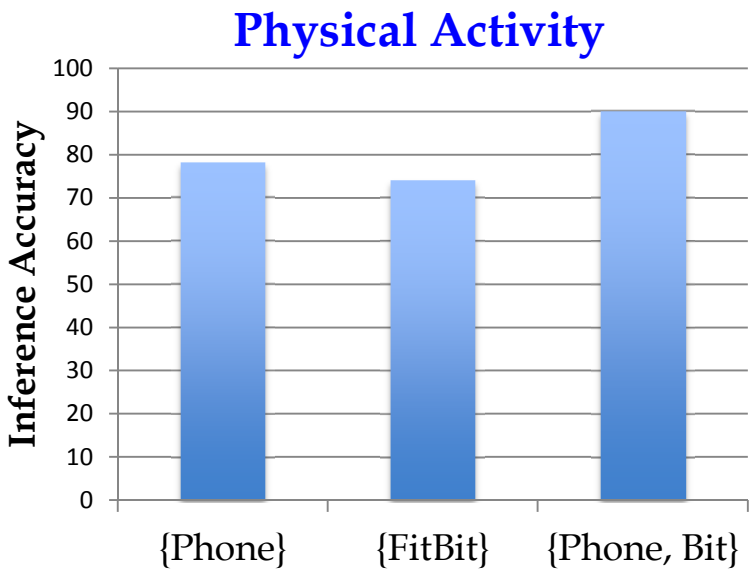
- Apps receive *typed inferences*
  - *(timestamp, state info, error)* tuple
  - E.g., ('2015-01-01 10:10:11', 'walking', 0.95)

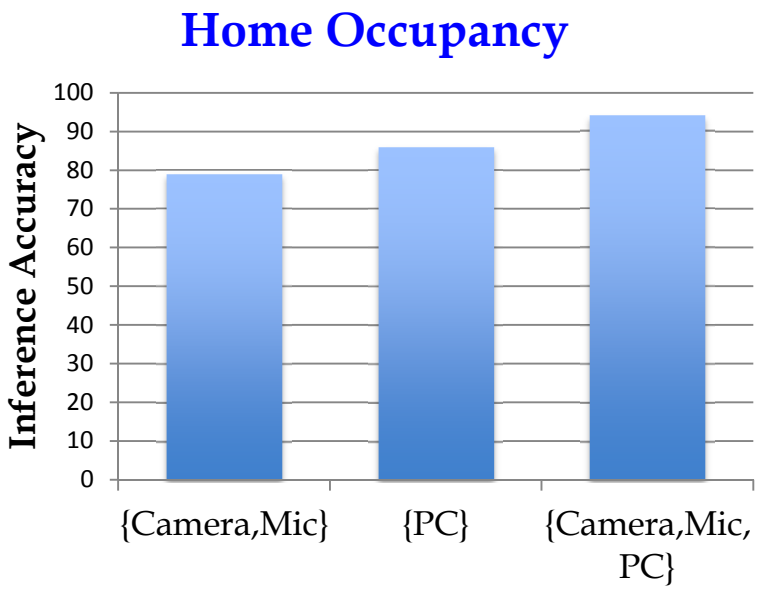
- **Inference graph**
  - Adapters
  - Inference modules
  - Channels
    - Local, remote
    - Supports disconnections
    - Delivery optimizations



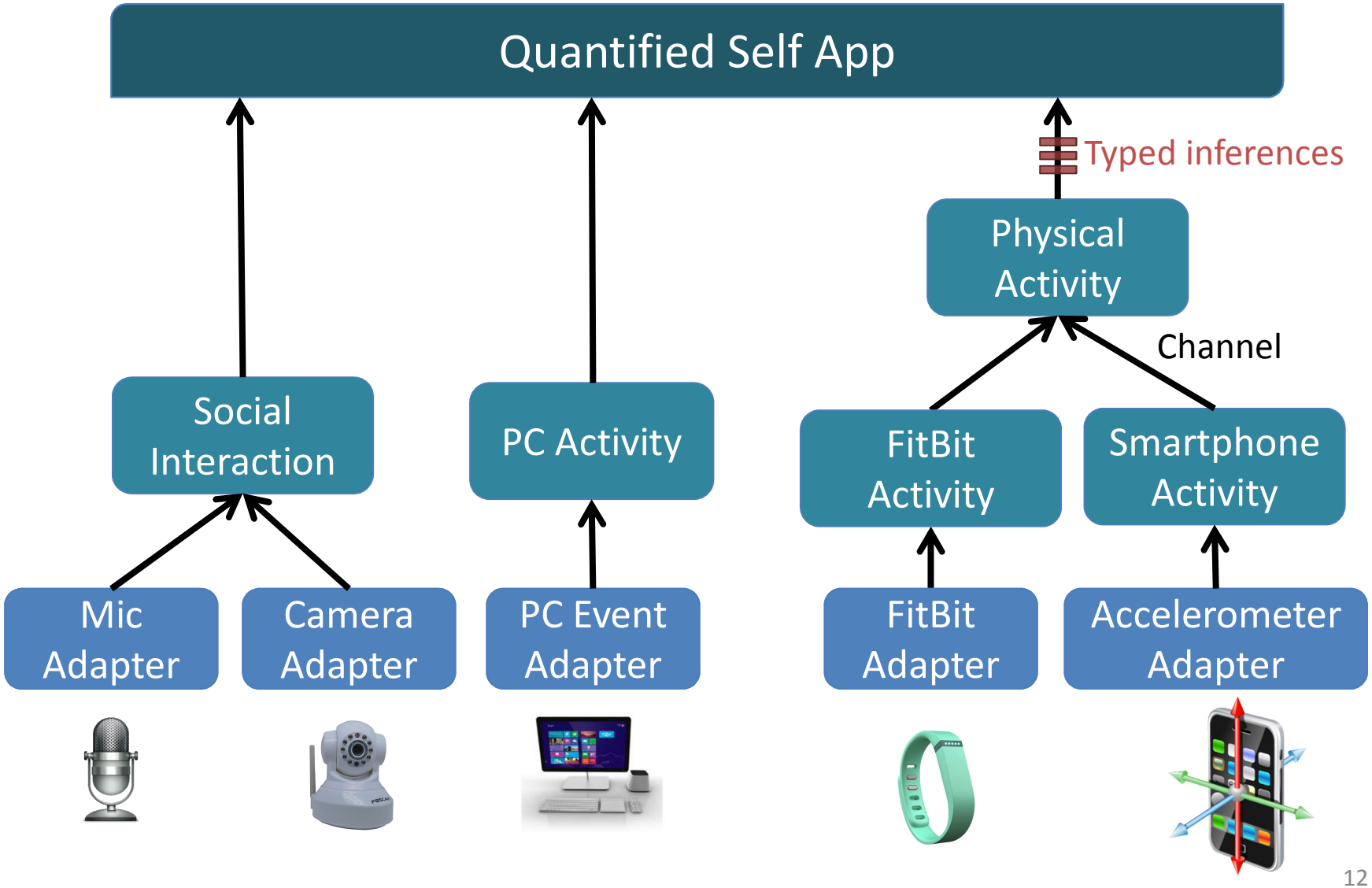
Advantages of Decoupling

- Ease of development
- Support for *heterogeneous* deployments
- Sharing inferences (and resources) across apps
- Improved inference accuracy by combining sensors

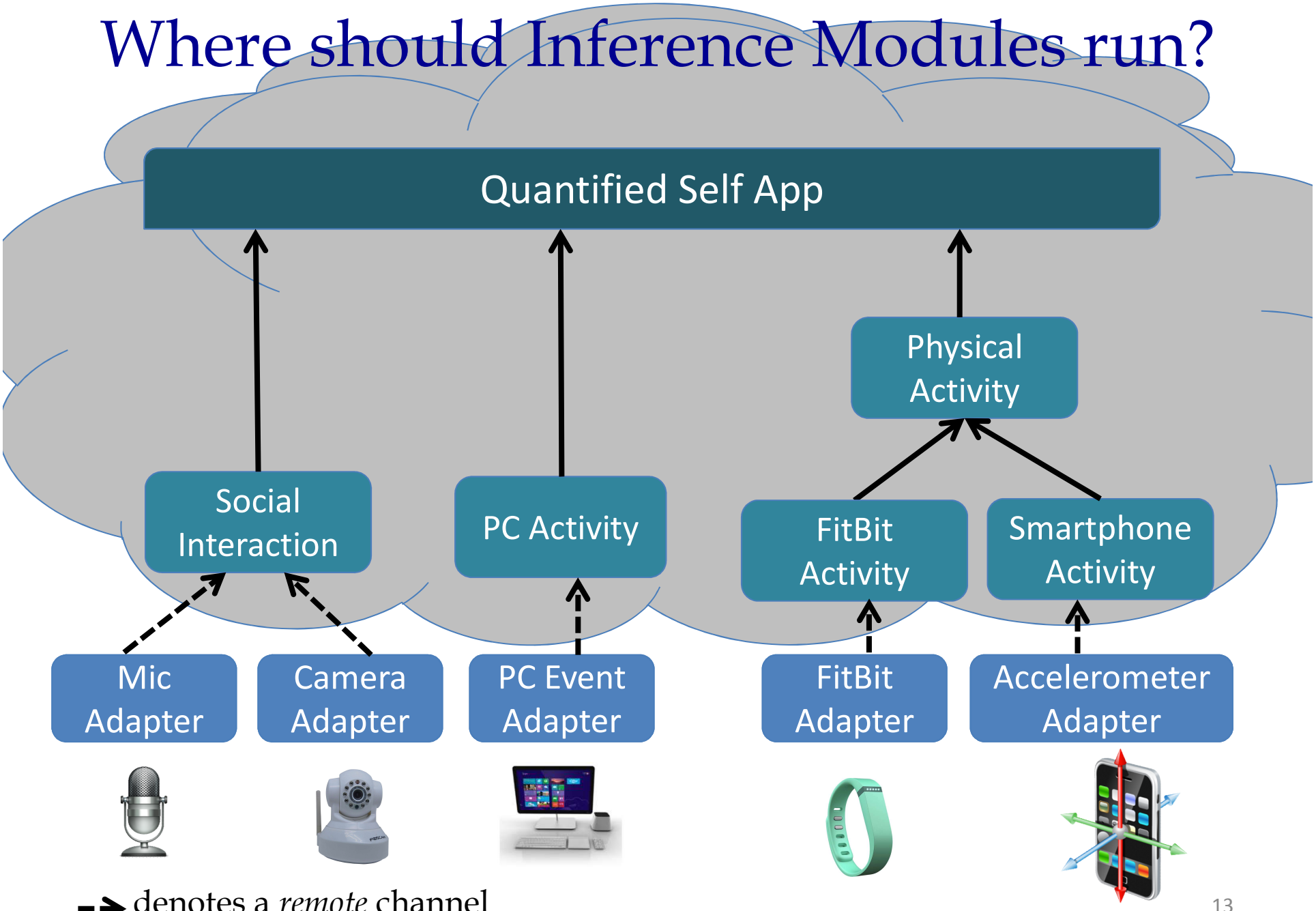




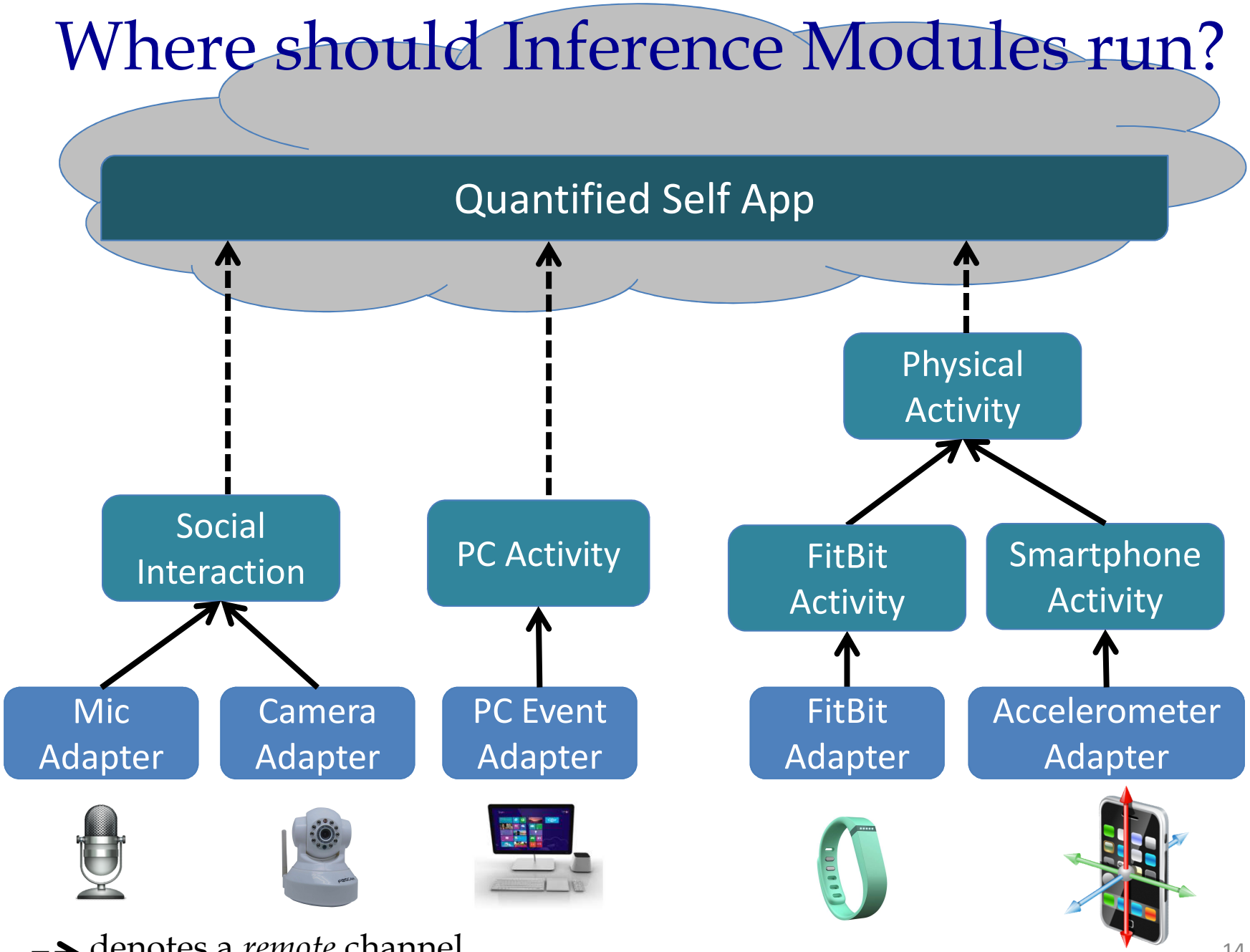
Inference Graph for Quantified Self



Where should Inference Modules run?

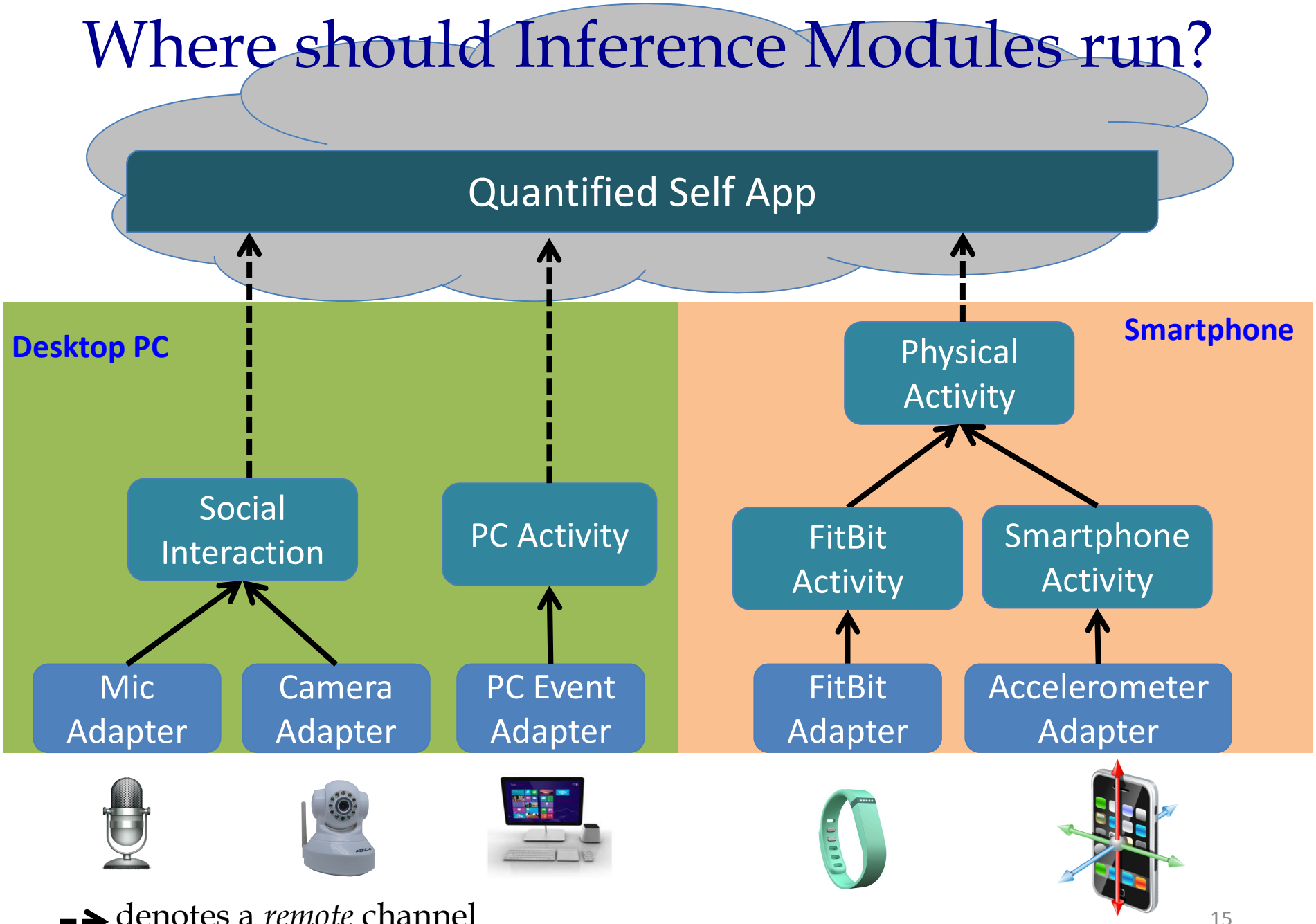


Where should Inference Modules run?



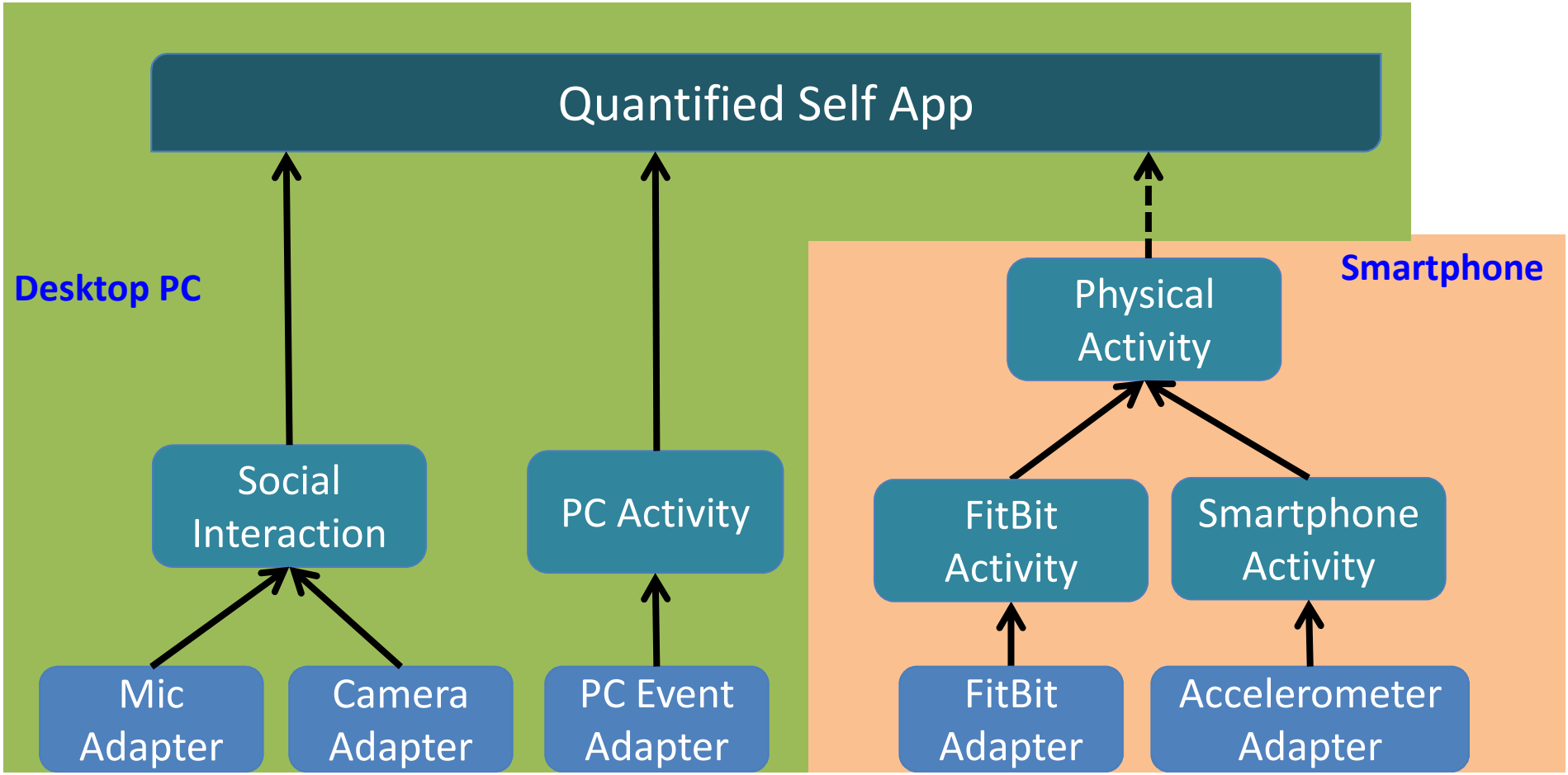
--> denotes a *remote* channel

Where should Inference Modules run?



--> denotes a *remote* channel

Where should Inference Modules run?

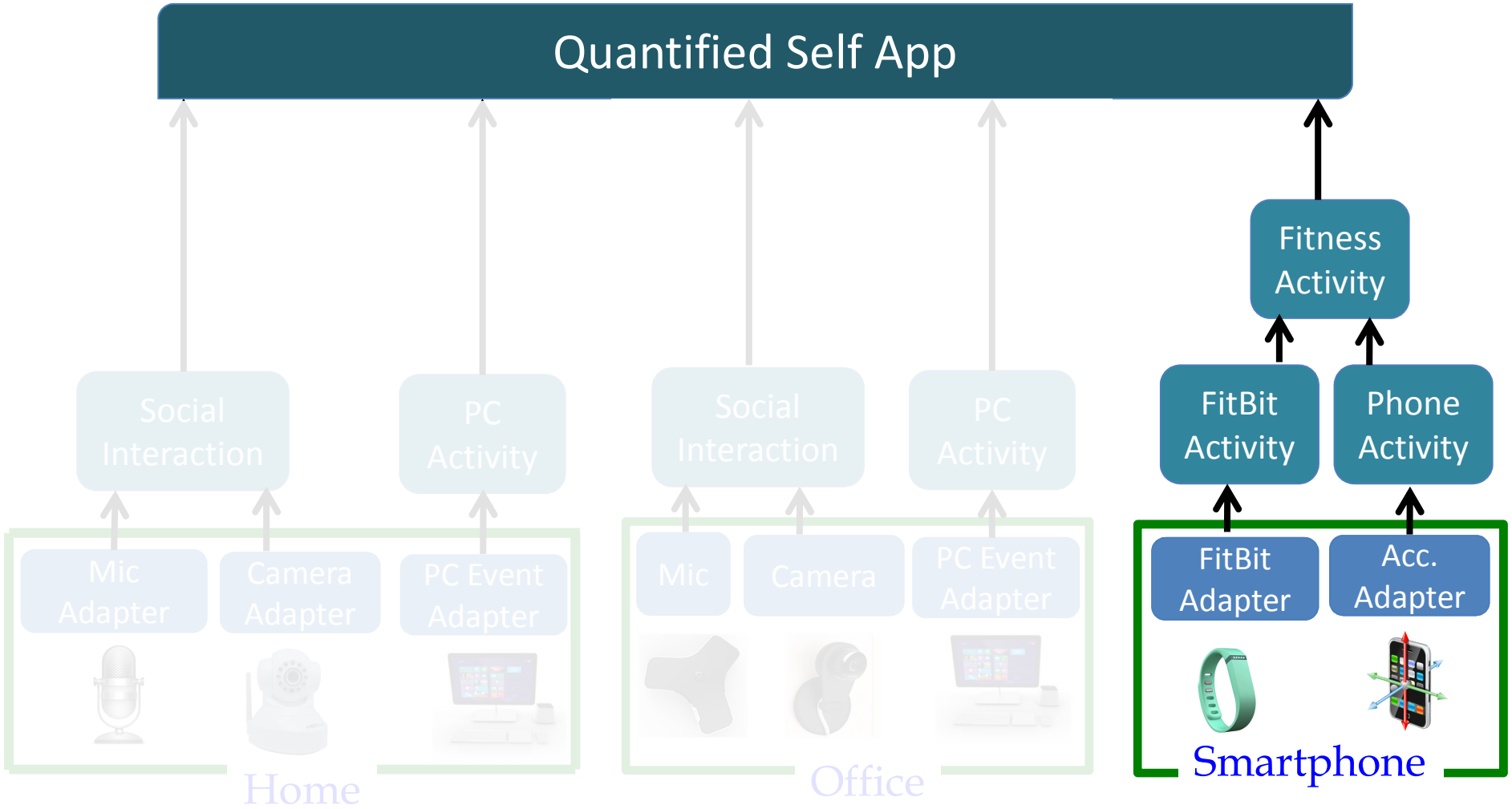




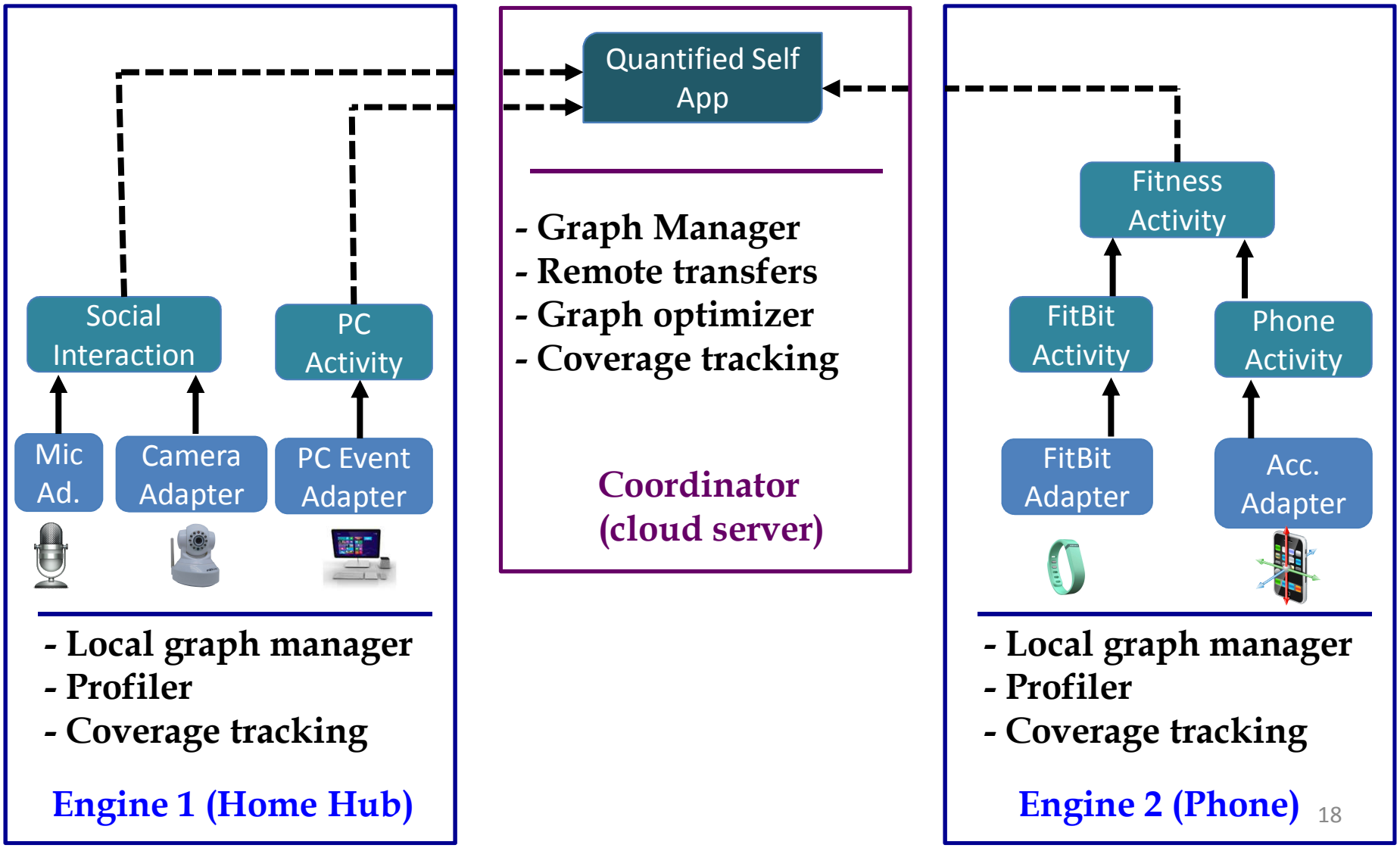


→ denotes a *remote* channel

Which devices should be selected?



Beam Design


Quantified Self App
- Graph Manager
- Remote transfers
- Graph optimizer
- Coverage tracking
Coordinator (cloud server)
Social Interaction
PC Activity
Mic Ad.
Camera Adapter
PC Event Adapter
- Local graph manager
- Profiler
- Coverage tracking
Engine 1 (Home Hub)
Fitness Activity
FitBit Activity
Phone Activity
FitBit Adapter
Acc. Adapter
- Local graph manager
- Profiler
- Coverage tracking
Engine 2 (Phone)

Current Prototype

- Cross-platform portable service
  - .NET 4.5, Windows Store 8.1, Windows Phone 8.1

Adapters
PC event
Phone GPS
Accelerometer, FitBit
Energy meter
Camera
PC, tablet mic
HomeOS adapter

Inference Modules
PC activity
Semantic location
Fitness Activity
Appliance use
Camera occupancy
Mic occupancy
Social interaction

- *Off-the-shelf* algorithms: Batra '14, Brush '13, Hao '13, Mark '14, Reddy '10, …
- Optimizations
  - Reactive, e.g., min #remote channels
  - Proactive, e.g., min remote data rate

Sample Apps

- Quantified self

- Rules (like IFTTT)
  - Alert if high-load *appliance on* and home not *occupied*

- Compared to monolithic approaches
  - Up to 3x increase in inference accuracy (user tracking)
  - Up to 12x reduction in developer effort (SLOC, #dev tasks)

Conclusion

- Apps today are built as *monolithic silos*
  - Requires handling several complexities
  - Device abstractions fall short

- *Inferences* as programming abstractions

- Beam inference framework
  - Unified view of inference logic across devices
  - Decouples, handles dynamics, optimizes resource use
  - Future work: *optimizers, coverage trackers, managing error*