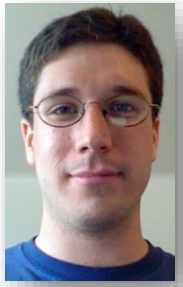
Towards commodity smarthomes

Ratul Mahajan

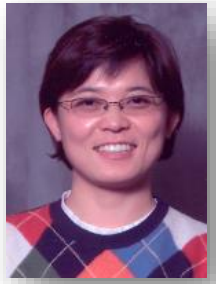
Microsoft Research

Partners in crime

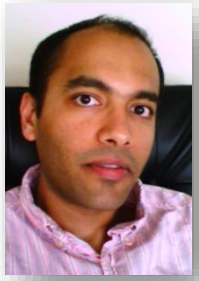
A.J. Brush



Colin Dixon



Bongshin Lee



Sharad Agarwal

Frank Martinez

Stefan Saroiu

Smarthomes

Abili ble,

dispara search]

Status of smarthome technology

Envisioned by many researchers and companies

Struggling to break into the mainstream

- Despite commercial availability since 1970s

Talk outline

What explains the gap?

What *really* explains the gap?

How to bridge the gap?

Study to understand the gap

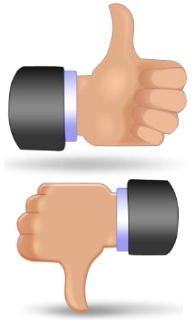
Visited homes with modern automation systems





Interviewed 31 people across 14 homes


Inventory

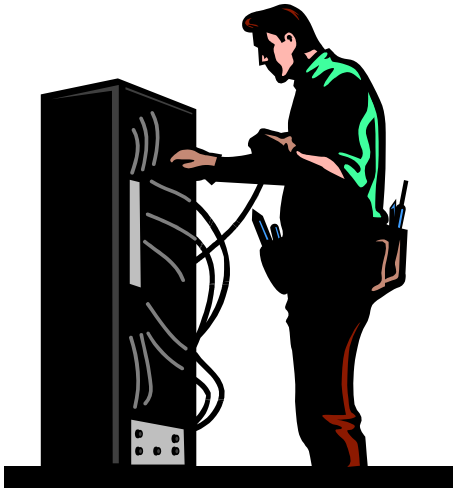

Semi-Structured Interview


Questionnaire


Home Tour

[Home automation in the wild: Challenges and opportunities, CHI 2011]

How smarthomes?



DIY (do-it-yourself)

$200 - $60k

Median: $5k

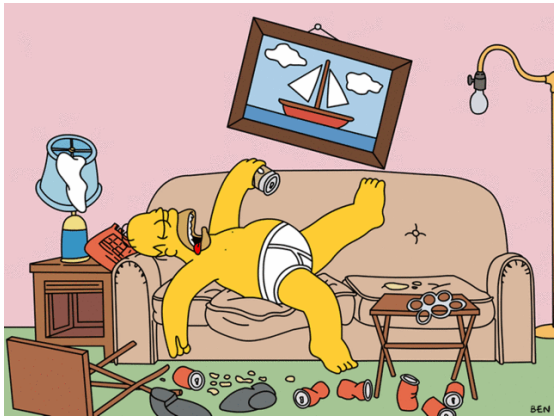
Outsource

$14k – $120k

Median: $40K

Why smarthomes?

Convenience



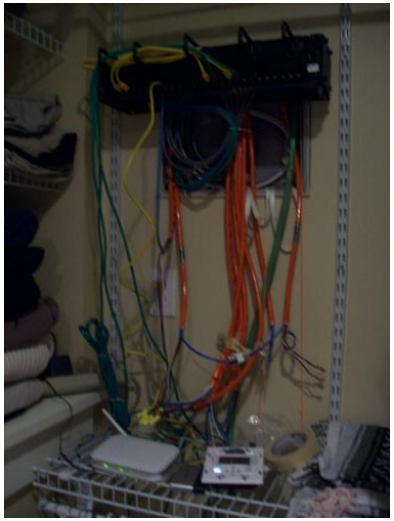
“It allows me to be lazy”

Peace of mind

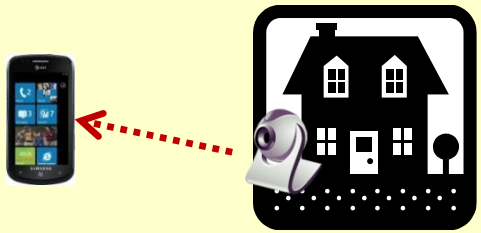
Control

Why not smarthomes?
Barriers to mainstream adoption

Hardware inflexibility



Poor extensibility



Adding tasks

Adding devices

Management nightmare

Access control

Talk outline

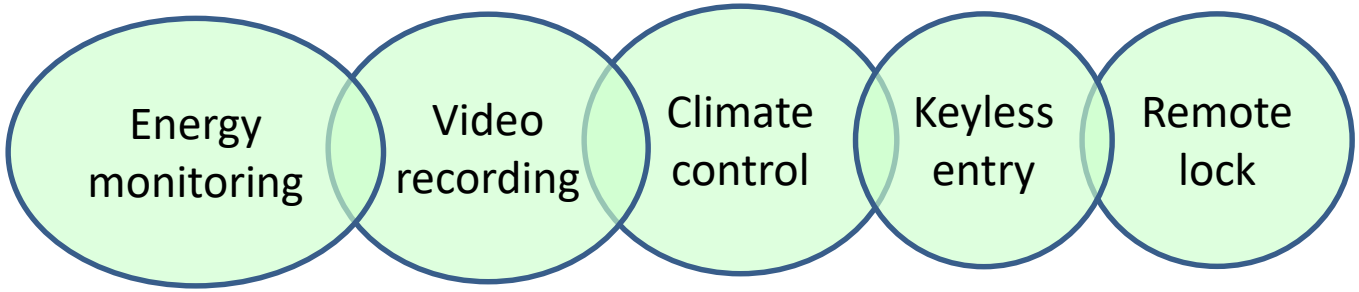
What explains the gap?

What *really* explains the gap?

How to bridge the gap?

The home computing environment

Tasks (software)



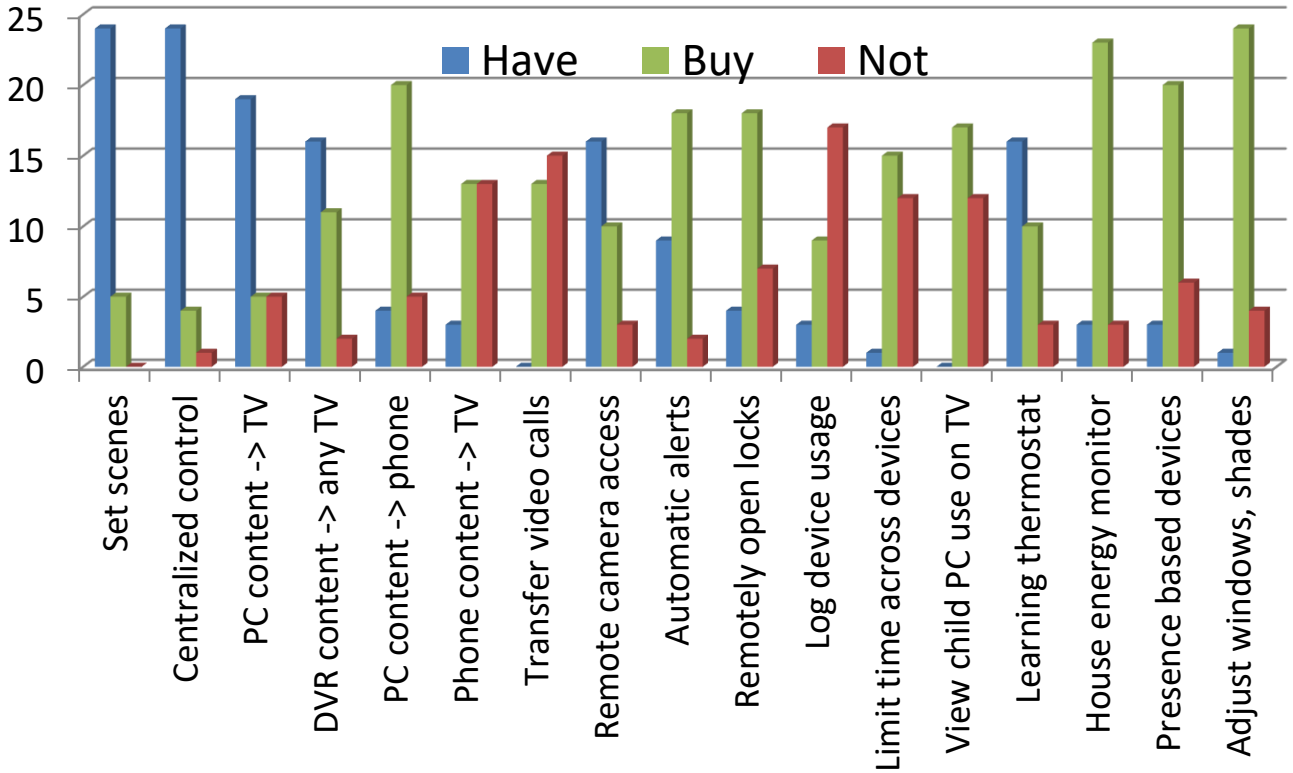
Devices (hardware)

Key characteristics of the environment

Users are not technically savvy

Extreme heterogeneity across homes

- Tasks
- Devices
- Topology
- Control, coordination



Prevalent abstractions for organizing home technology

Network of devices

=> Interoperability protocols

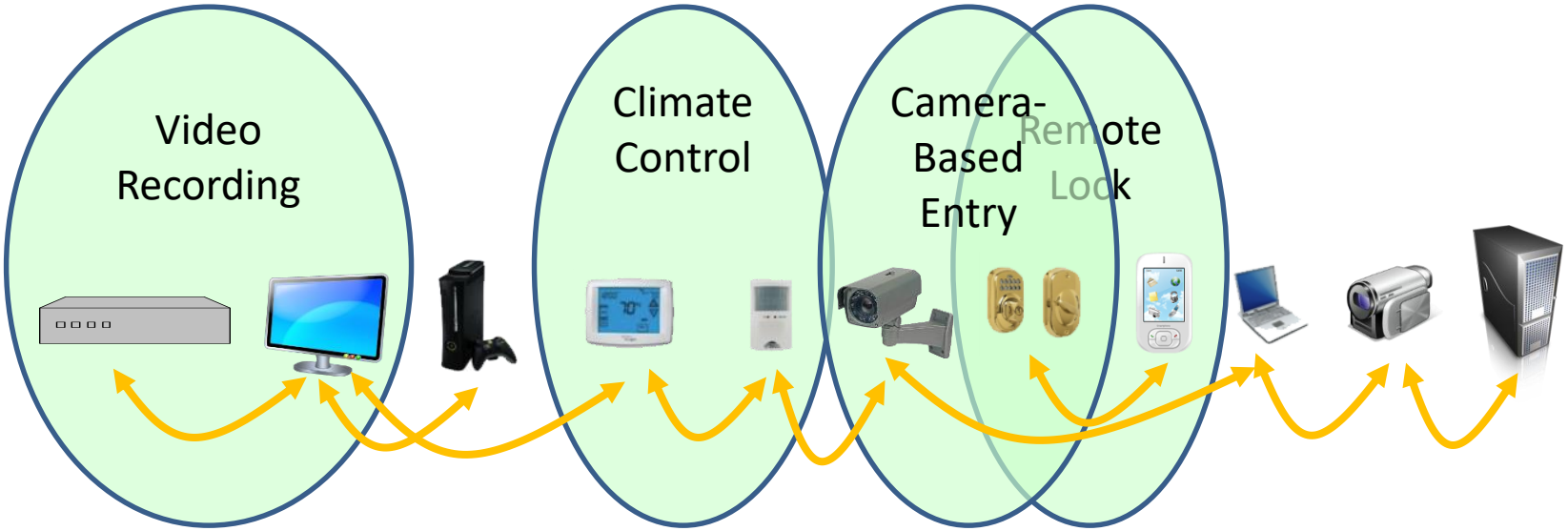
- DLNA, Z-Wave, Speakeasy, …..

Closed, complete systems

=> Vertical integration

- Crestron, Control4, EasyLiving, …

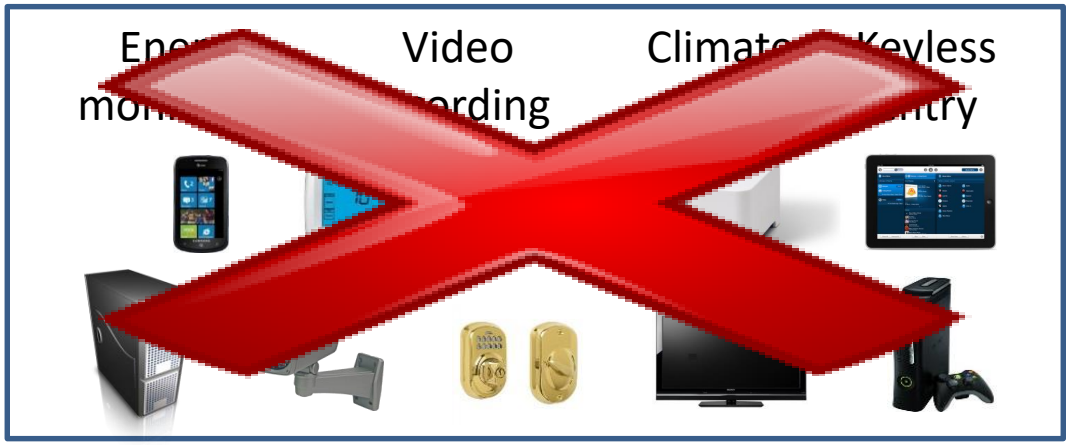
Interop alone leaves too much for users

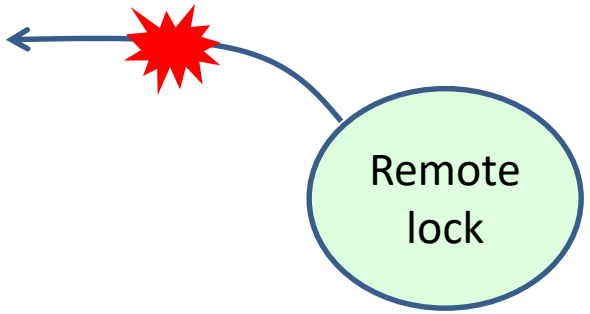


No systematic manner to add tasks

Does not handle management and security

Vertical integration limits extensibility





Keyless entry

Climate control

Video recording

Storage, printing

Talk outline

What explains the gap?

What *really* explains the gap?

How to bridge the gap?

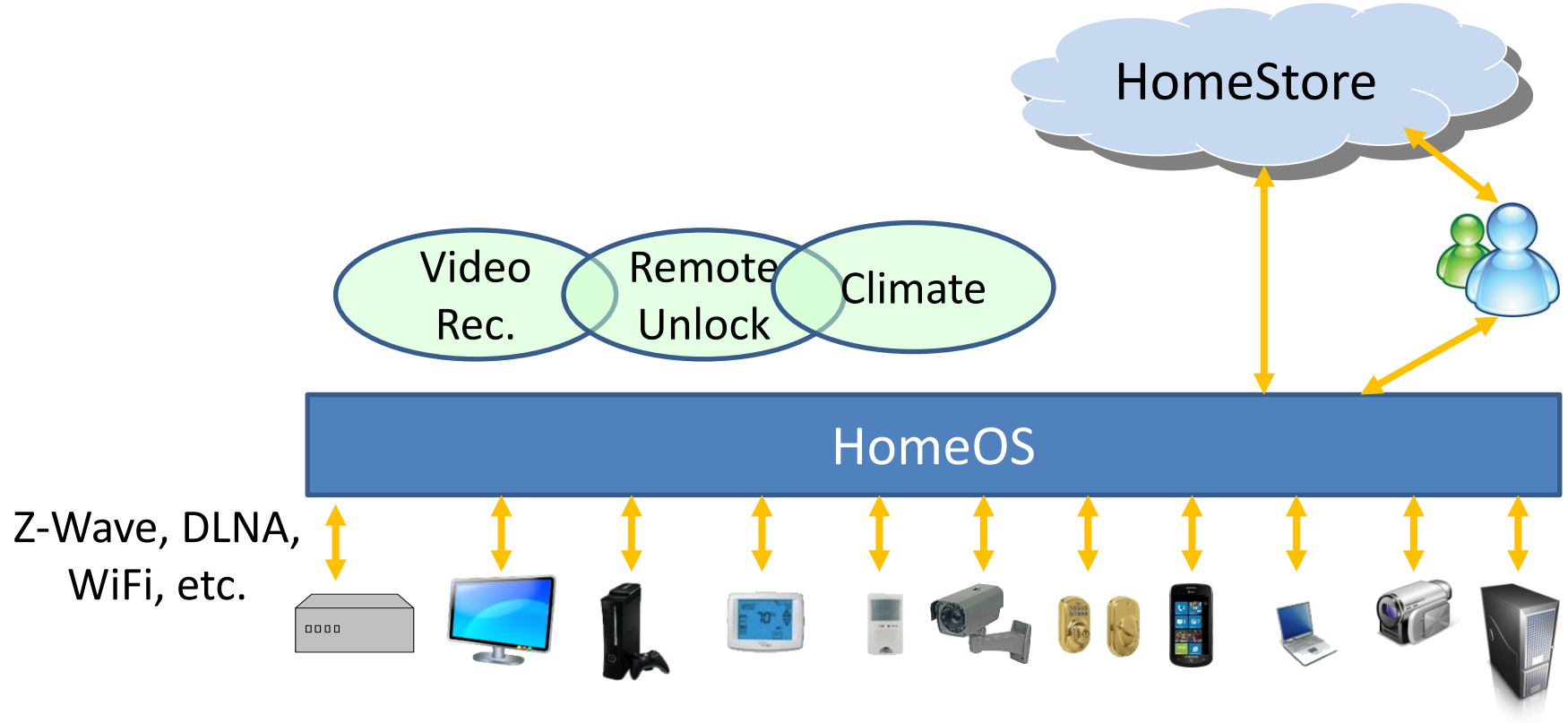
Our abstraction

View the home as a computer

- Networked devices =~ peripherals
- Tasks over these devices =~ applications
- Adding devices =~ adding a peripheral
- Adding tasks =~ installing an application
- Managing networked devices =~ managing files

[The home needs an operating system (and an app store), HotNets 2010]

HomeOS: An OS for the home

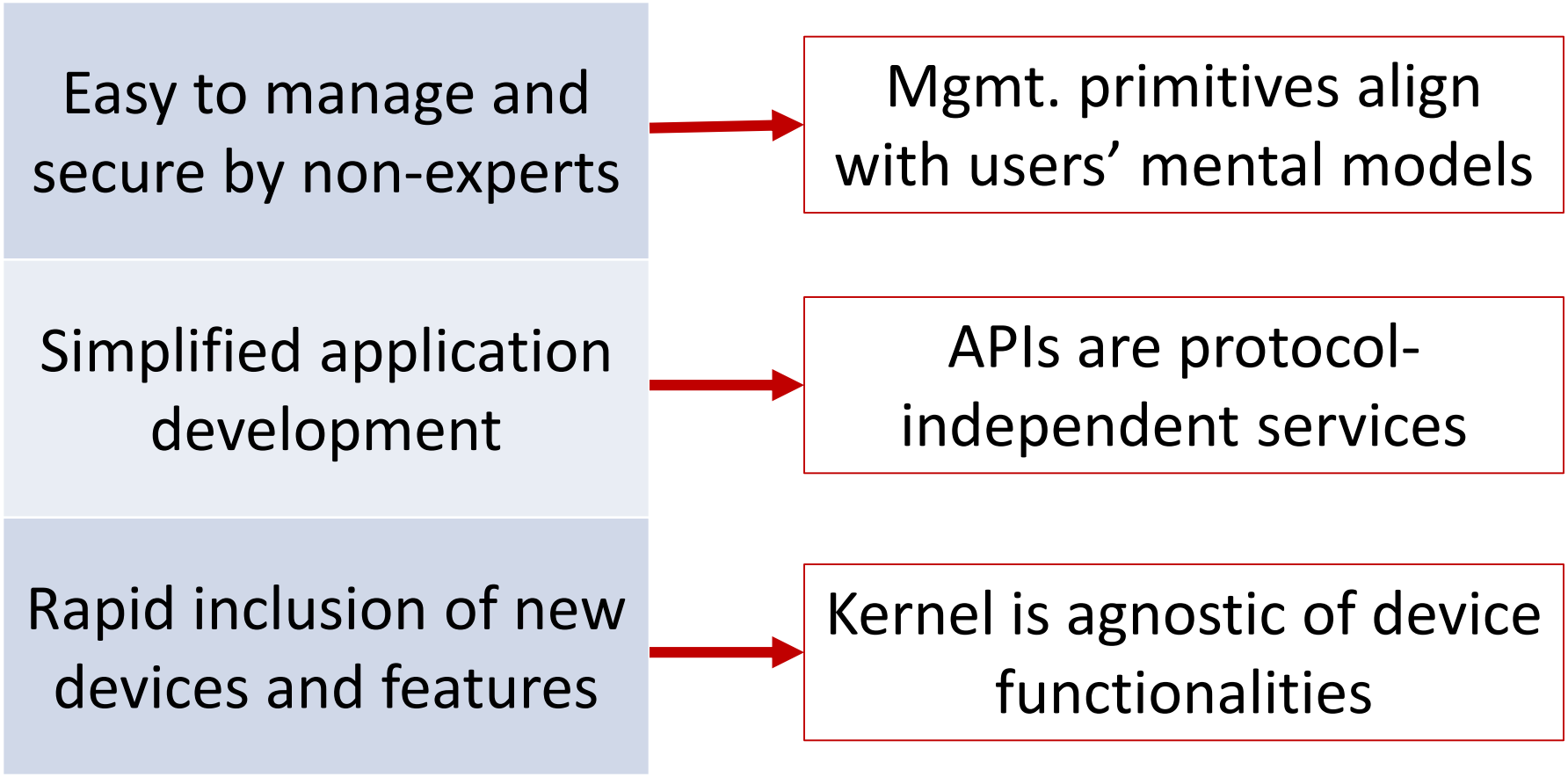


HomeOS logically centralizes all devices

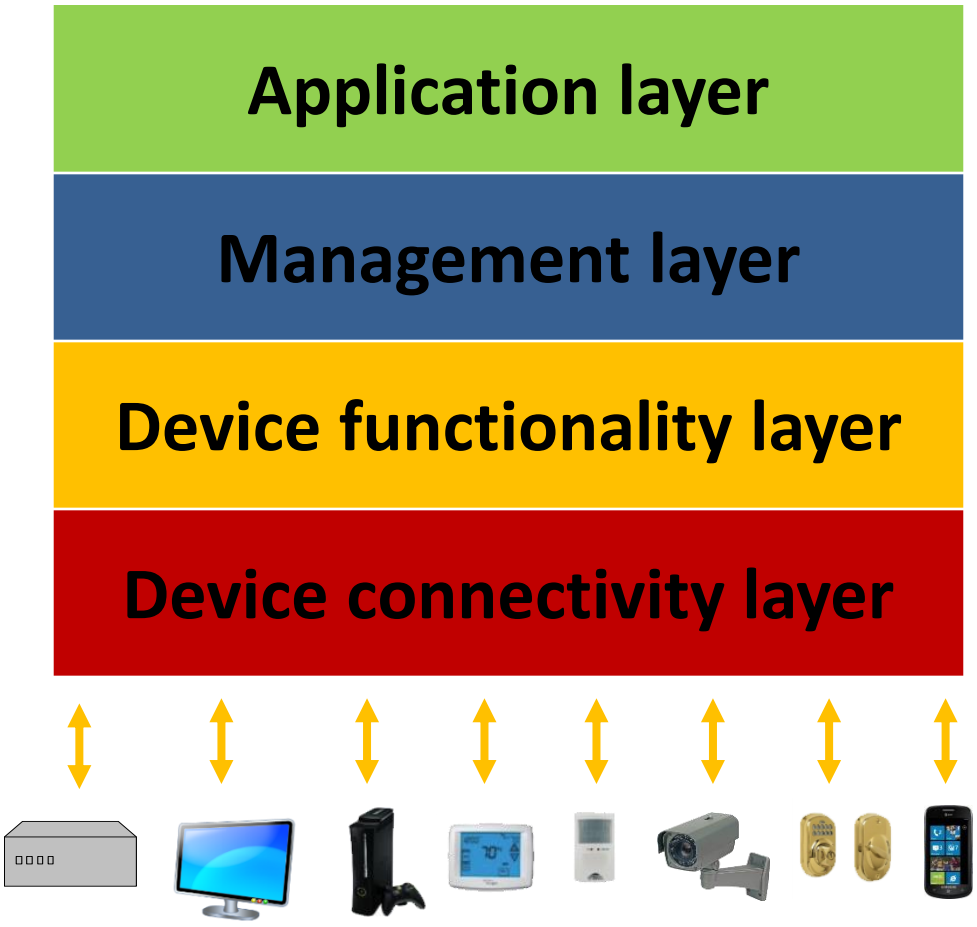
Users interact with HomeOS, not individual devices

HomeStore helps find compatible devices and apps

Goals of HomeOS



HomeOS layering model



DCL and DFL

DCL provides basic connectivity to devices

DFL exports device functionality as a service

- Services are protocol-independent
- Identified using roles and operations

Dimmer	**MediaRenderer**	**PTZ Camera**
Set(level) Get*() → level	Play(uri) PlayAt(uri, time) Stop() Status*() → uri, time	GetImage*() → bitmap Up(), Down() Left(), Right() ZoomIn(), ZoomOut()

DCL and DFL: Reflections

Device interaction is split across two layers

- Allows smaller modules, avoids functionality duplication

Functional specs are more stable than device protocols

- Allows independent evolution of protocols and apps

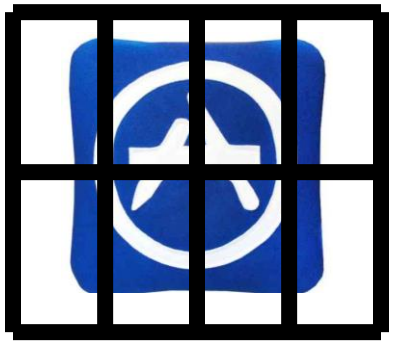
DFL modules can export multiple roles for a device

- Allows easy extensibility along with backward compatibility

Management layer: Requirements

Time-based access control

Apps as security principals



Easy-to-understand settings

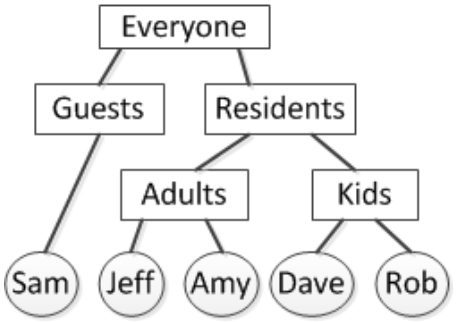
Management layer: Primitives

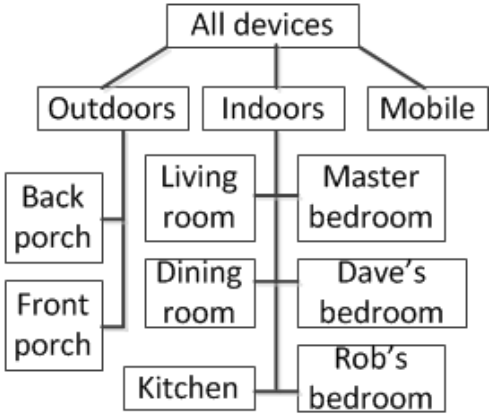
Access control policy as Datalog rules

[device, user group, app, t_{start}, t_{end}, dayOfWeek, priority, accessMode]

Time-based user accounts

Hierarchical user, device groups





Application layer

Apps consume and compose DFL services

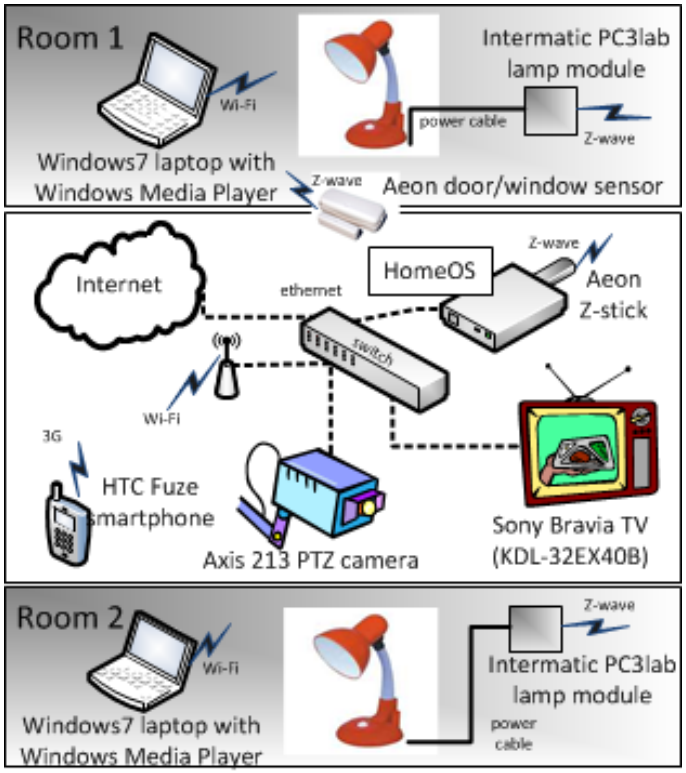
Manifests help with testing compatibility

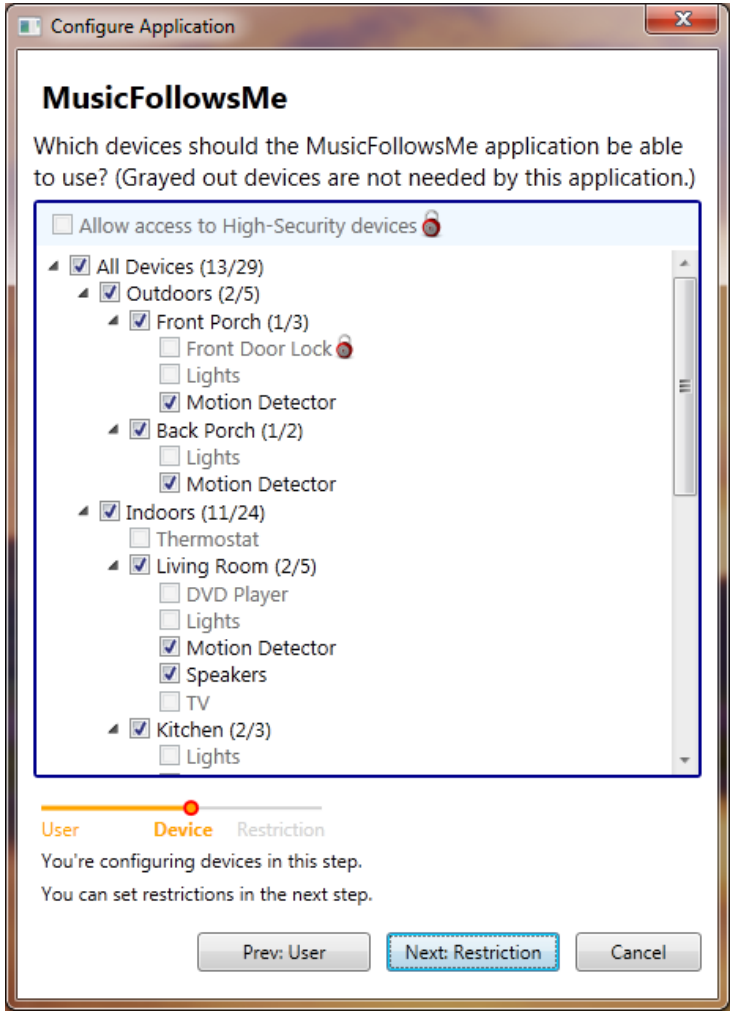
- Lists of mandatory and optional requirements
- E.g., mandatory: *{TV, SonyTV}, {MediaServer}*
  optional: *{Speaker}*

Implementation overview

Component-based OS

Uses C# and .NET





Evaluating HomeOS

Key questions:

- Can non-technical users manage HomeOS?
- Can developers easily write apps and drivers?

Method:

- Field experiences
- Controlled experiments

Field experience with HomeOS

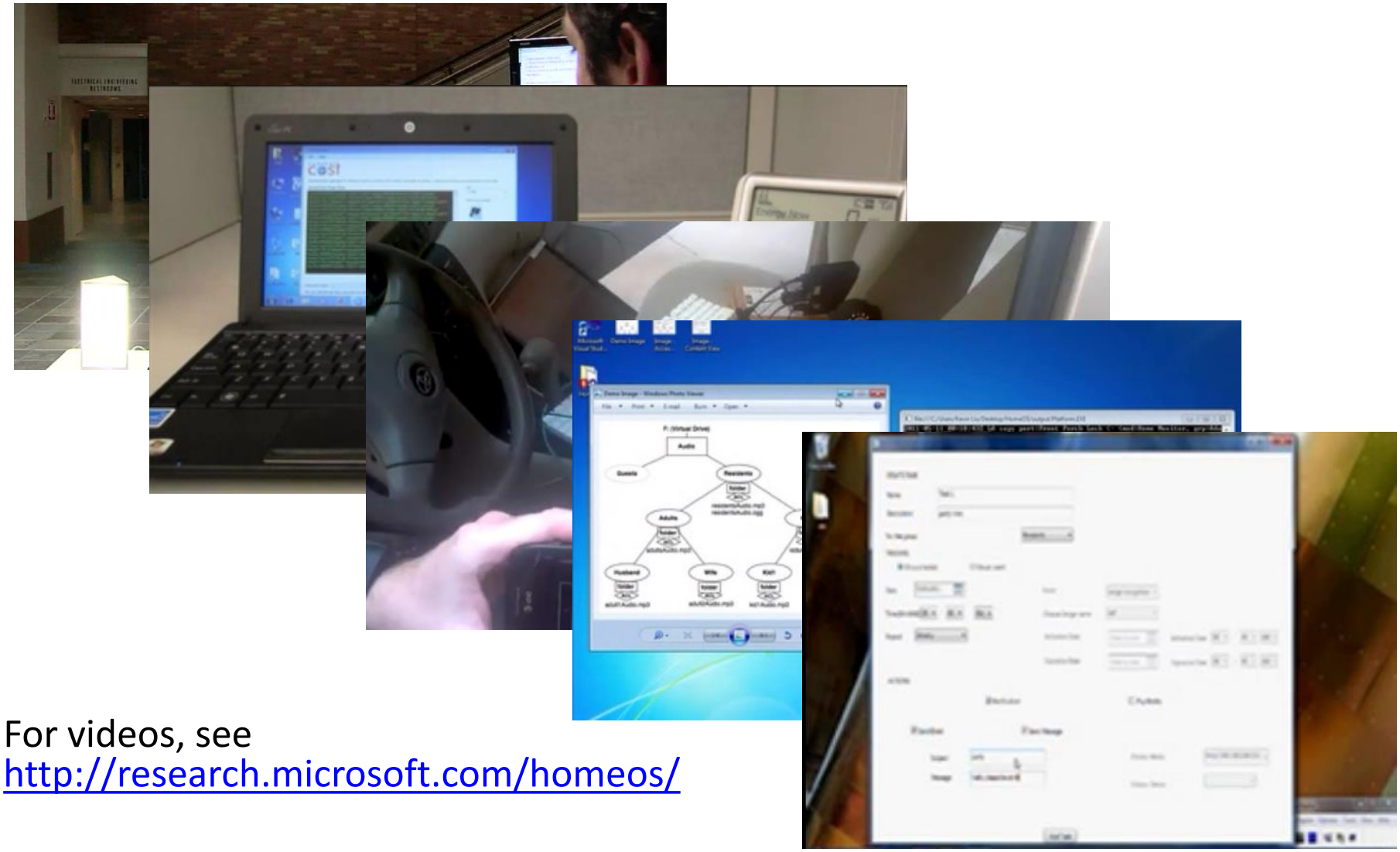
12 homes running HomeOS for 4-8 months

- Using different devices and applications
  - E.g., Cameras, light controllers, door-window sensors

41 student developers across 10 research groups

- Developed new drivers and apps
  - E.g., energy meters, IM, appliance controllers

Example applications



For videos, see
http://research.microsoft.com/homeos/

Field experience: The good

Users could manage their HomeOS deployments

Users particularly liked the ability to organically extend their technology

Developers found the programming abstractions and layering to be “natural”

Field experience: The bad

Users found it hard to diagnose faults

Interoperability protocols can be fragile

Not all device features may be exposed over the network

Results from controlled studies

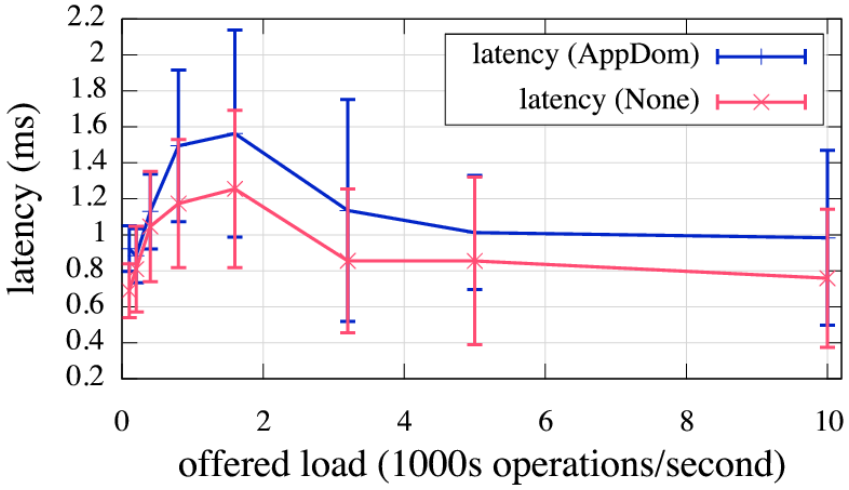
10 developers asked to write a realistic app

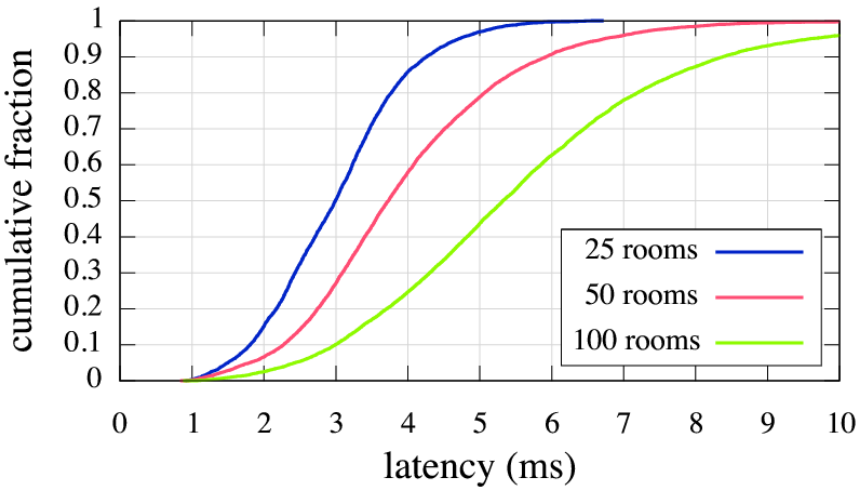
- 8 finished in under 2 hours

12 non-technical users given 7 mgmt. tasks w/o training

- 77% completion rate; 89% after removing an outlier task

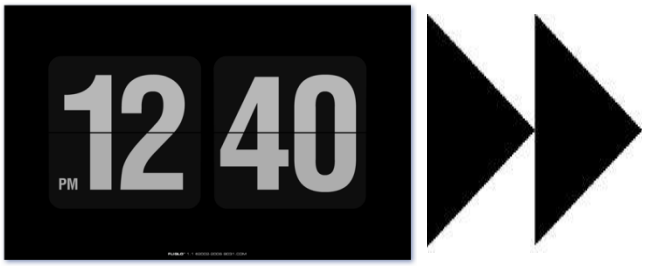
System can support interactive apps and large homes





Ongoing work

Predictable control through fast forwarding



Sensor data privacy and neighborhood watch

Conclusions

Extensibility and management hurdles are keeping smarthomes out of mainstream

Organize home technology as a computer

HomeOS is one way to realize this organization