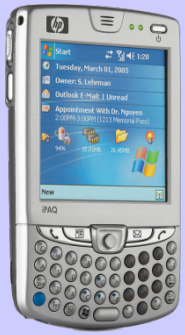
UIA:
A Global Connectivity Architecture for Personal Mobile Devices

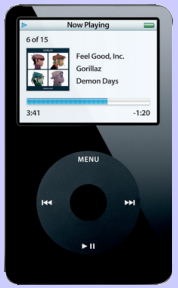
Bryan Ford
Massachusetts Institute of Technology

in collaboration with
Jacob Strauss, Chris Lesniewski-Laas,
Sean Rhea, Frans Kaashoek, Robert Morris

http://pdos.csail.mit.edu/uia

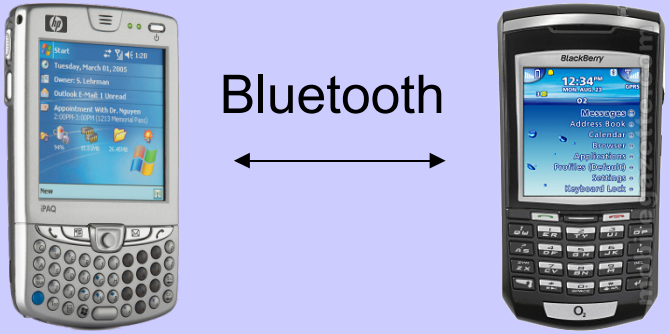
Personal devices everywhere



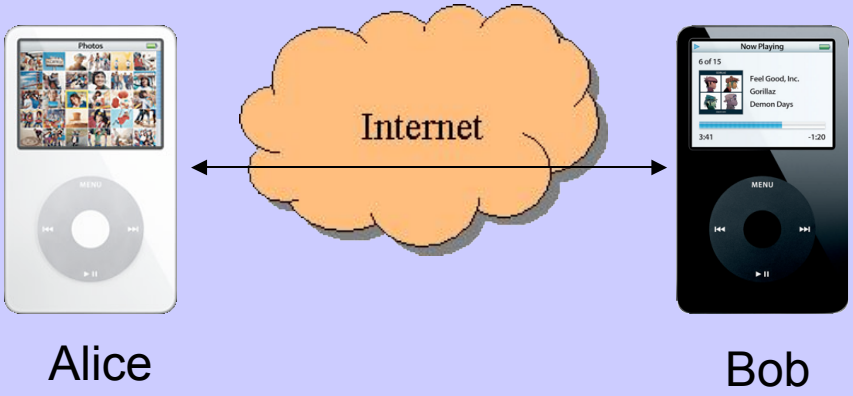


- Internally they are like real computers
- They will be part of the Internet
- They will store data that people want to share

Global connectivity enables information sharing



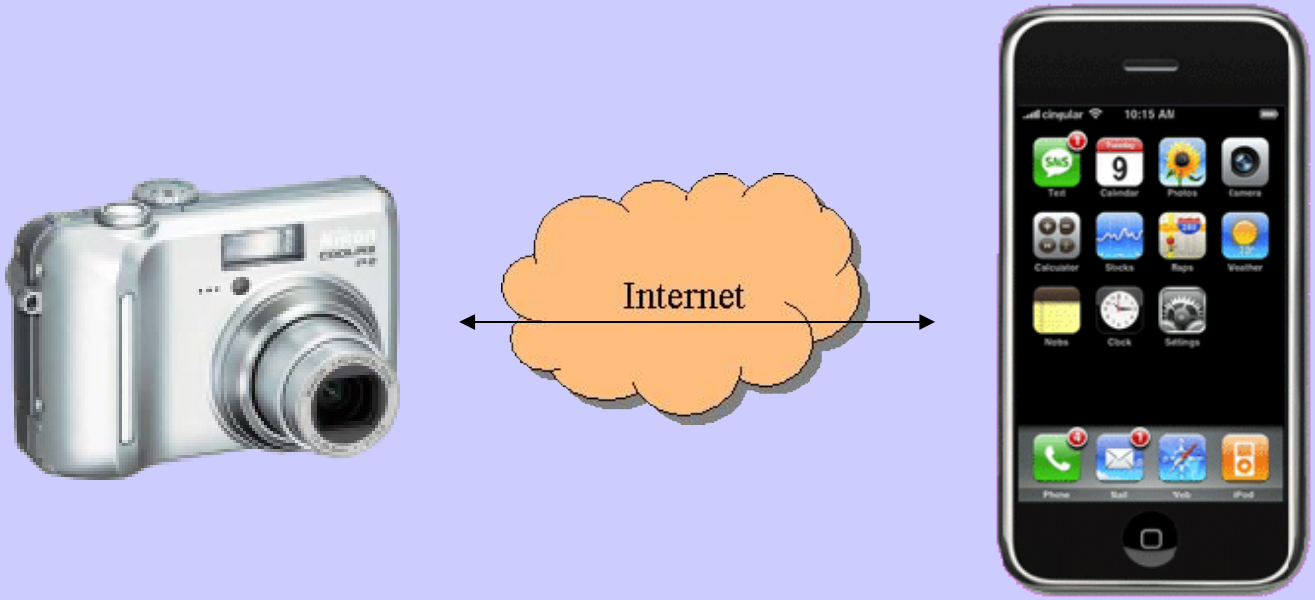
- Alice and Bob meet



- Alice & Bob later share stuff remotely

Other examples

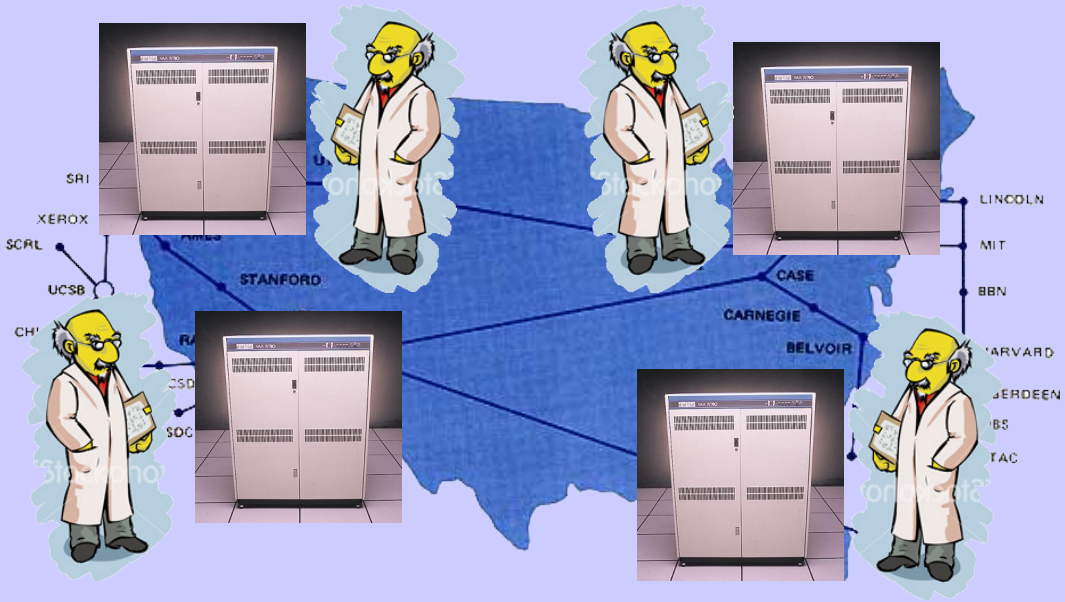
- Upload picture from camera to mom’s iPhone
- Stream video from ambulance to doctor’s PDA
- Car-to-car local traffic information



The Internet's Evolution

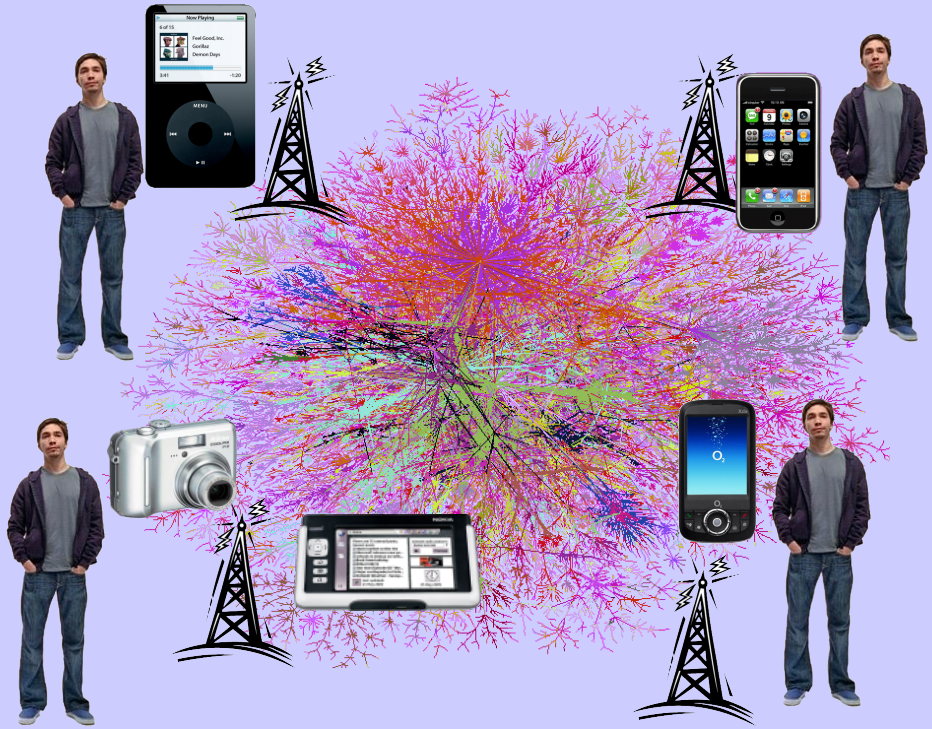
Internet designed for

- **wired** networks
- **fixed** computers
- **expert** operators



...but now supports:

- **wireless** nets
- **mobile** devices
- **unskilled** users



The Problem

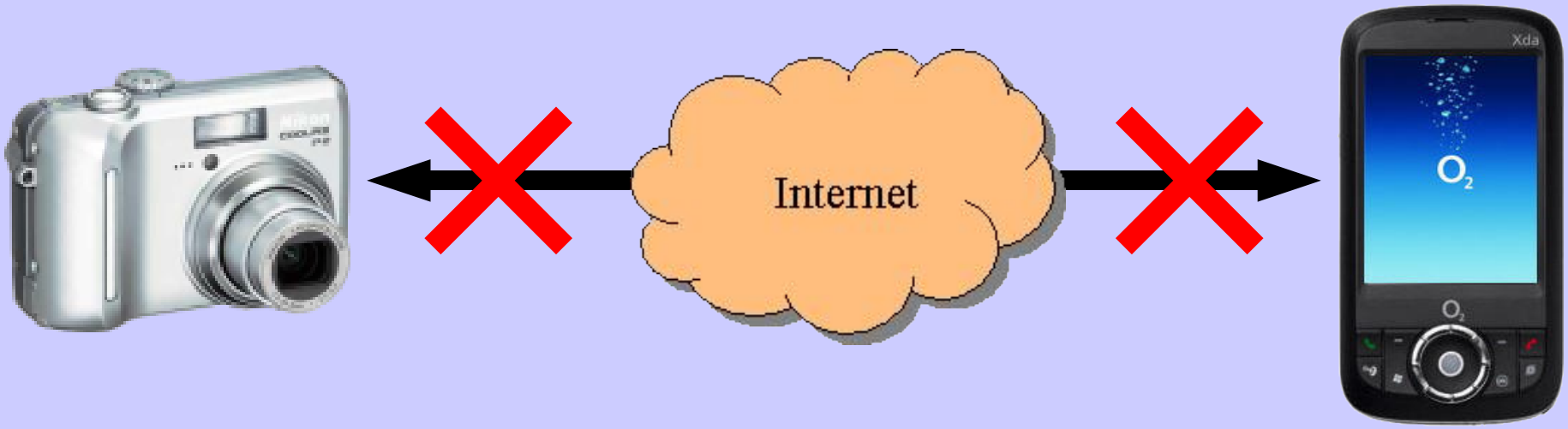
old design assumptions

+

Internet evolution

=

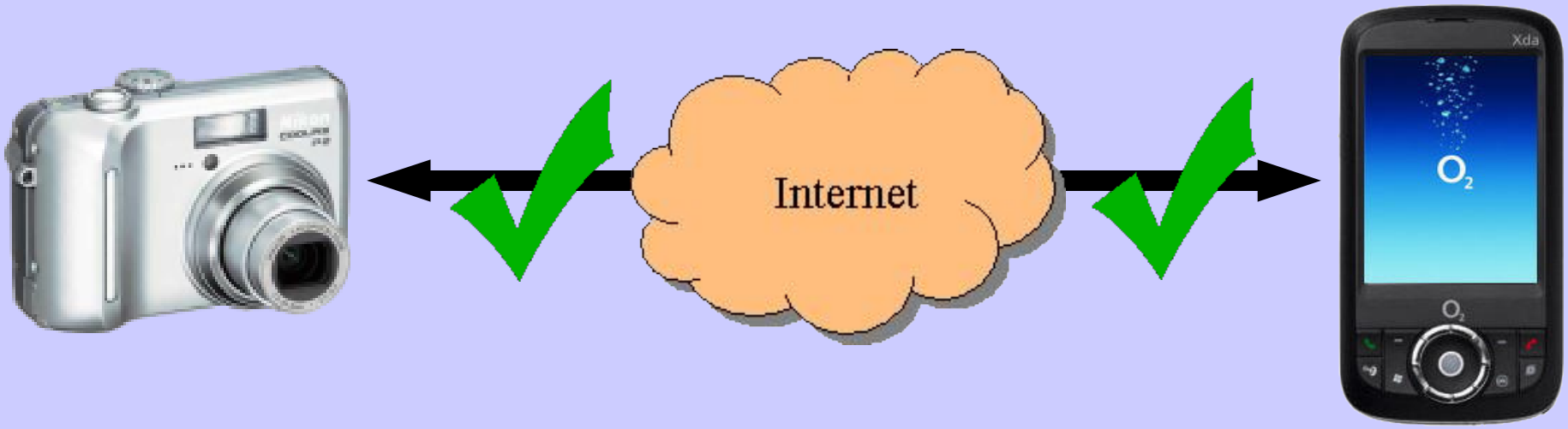
connectivity challenges for personal devices



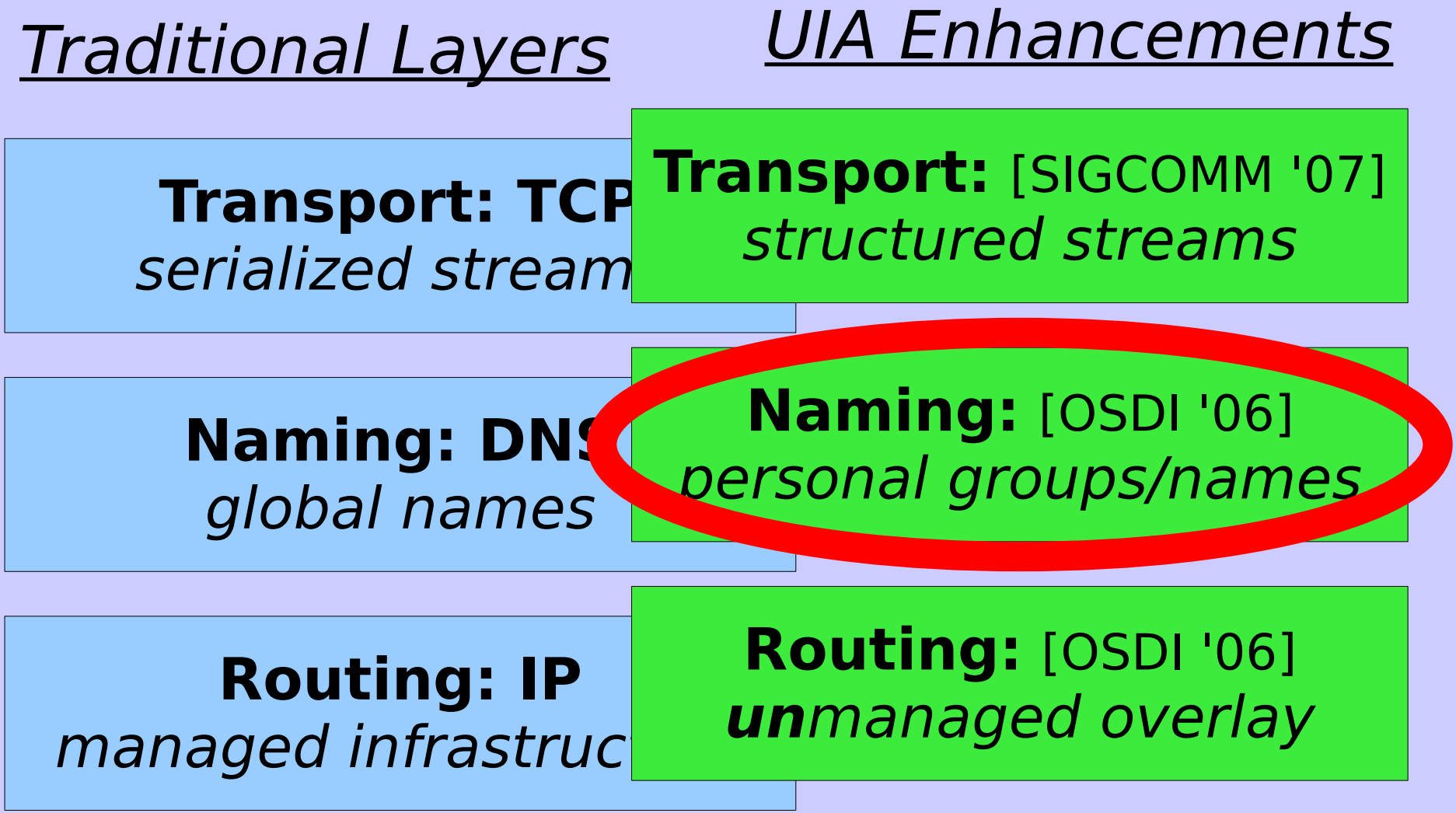
The Project

Unmanaged Internet Architecture (UIA)

Goal:
Make personal device connectivity “**just work**”
by **rethinking basic networking concepts**



Architecture Overview



Naming Scenario

Bob & Alice:

1. Meet at conference
2. Re-connect remotely over Internet
3. Meet again off-Internet

Naming Scenario (1)

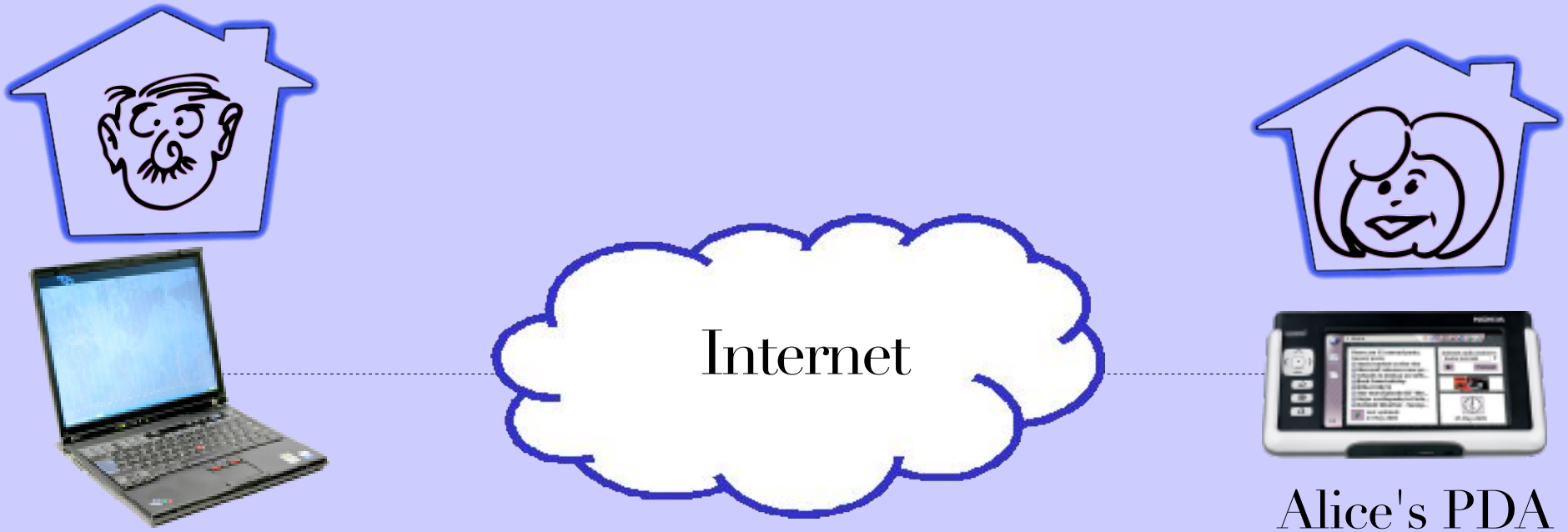
Local Area Network

Bob's Laptop

Alice's PDA

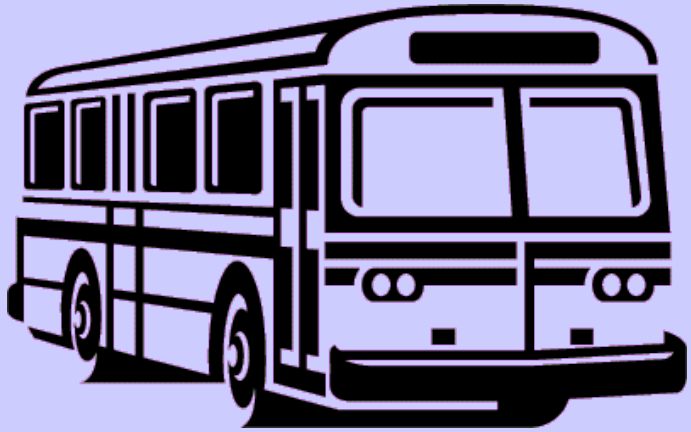
Bob & Alice meet, connect [Bonjour] –
using **local names** (e.g., “Alice-PDA”)

Naming Scenario (2)



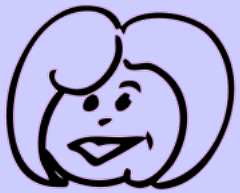
Wish to re-connect remotely –
need **different, global names** & more setup
(e.g., "pda.alice1234.herisp.com")

Naming Scenario (3)



Bob's Laptop

Partitioned
Ad Hoc Network



Alice's PDA

Meet again off-Internet – global names stop working!
Require **different, local names** (again)

Key Naming Challenges

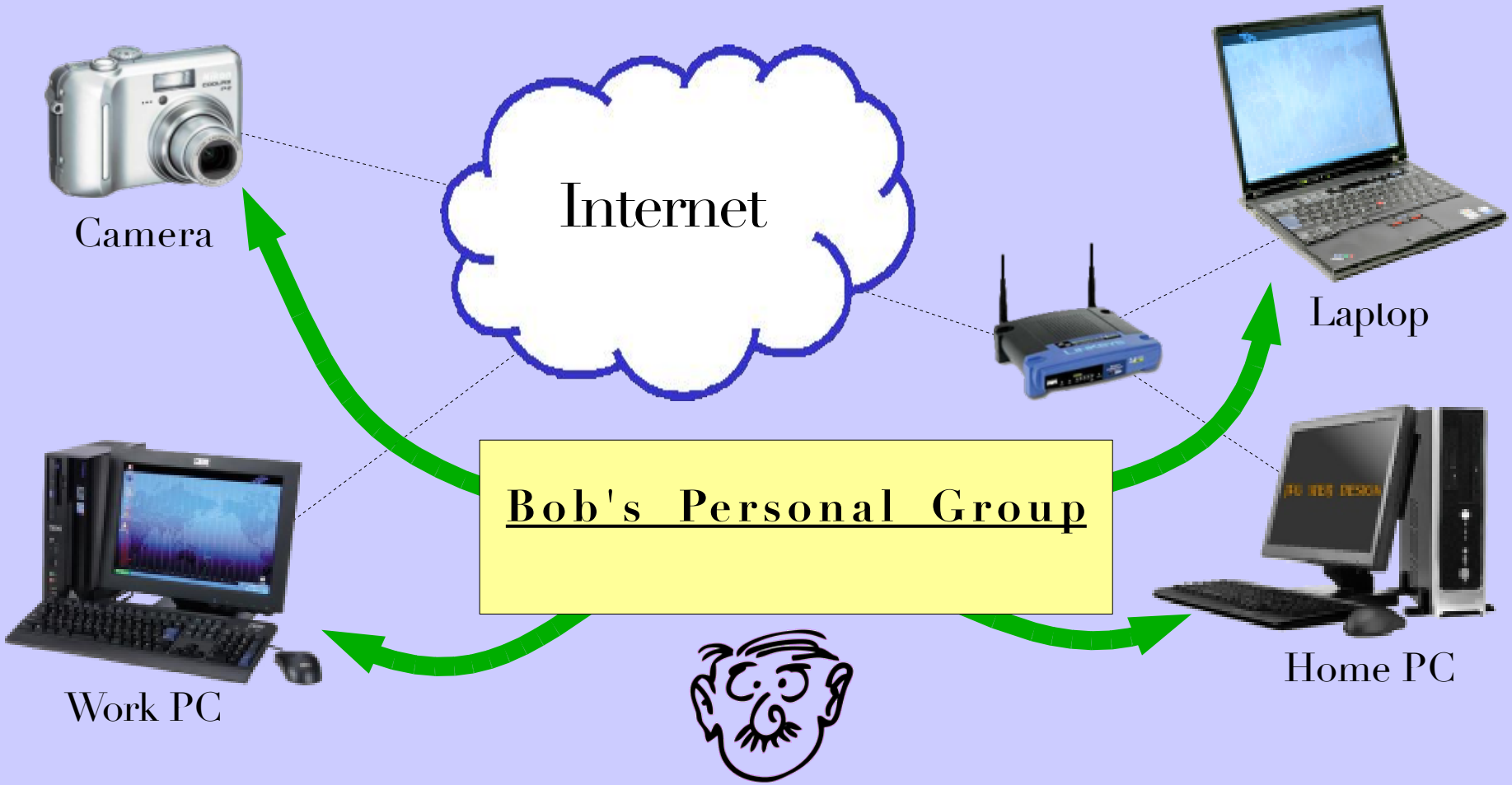
Personal device names should be:

1. **Convenient**
   - *short, personally meaningful*
2. **Consistent**
   - *usable on any device I own/manage*
3. **Available**
   - *works even under disconnect/partition*

#3 precludes central name service!

Key Naming Contribution

Personal Group:
distributed federation of personal devices



What is a Personal Group?

Combination of:

- A **distributed namespace** of devices, users, ...
- An **ad hoc virtual private network** (VPN)
- A **user identity** for **social networking**

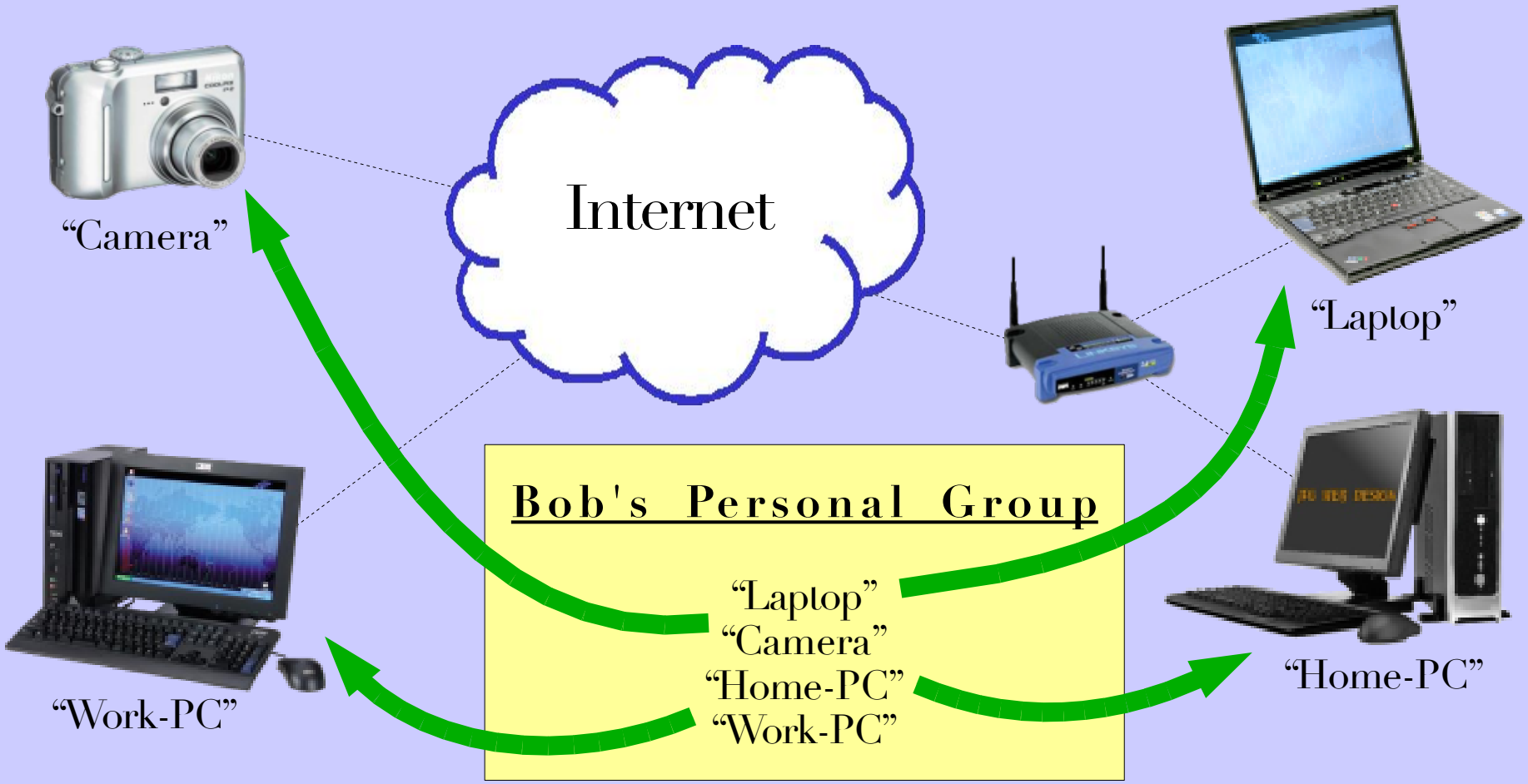
...with **fully decentralized, user-friendly management & operation**

Outline

- ✔ Introduction
- Personal Group Naming Model
  - *from user's perspective* $\Longrightarrow$ ***convenient***
- Implementing Personal Groups
  - *decentralized* $\Longrightarrow$ ***consistent, available***
- Evaluation
- Other thesis components
- Related work, conclusion

Personal Names

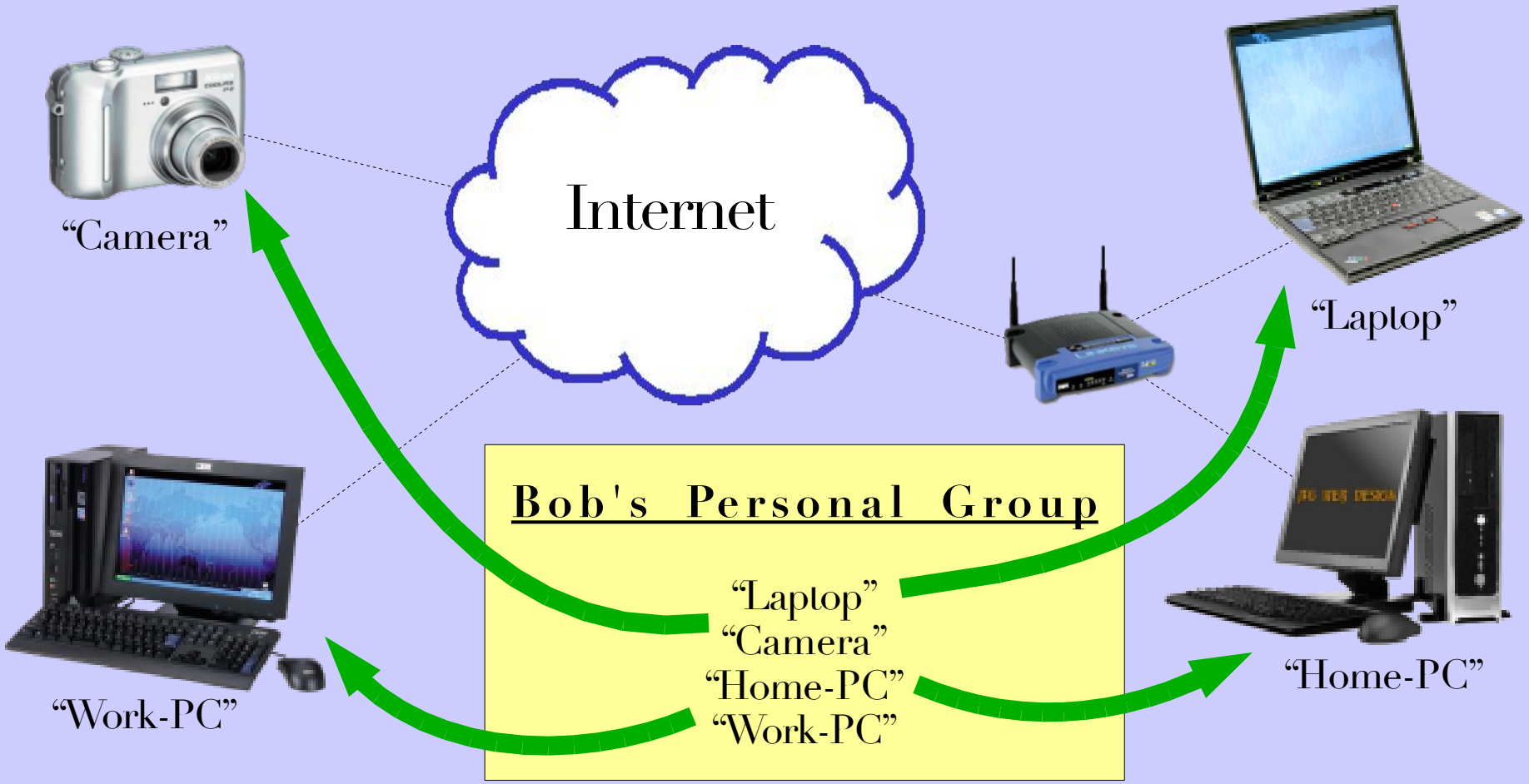
Each personal group includes a distributed ***personal namespace***



Personal Names

...are **short**, **local** to personal group

→ "laptop", *not* "laptop.bob345.his-isp.com"



Why Local?

Global names:

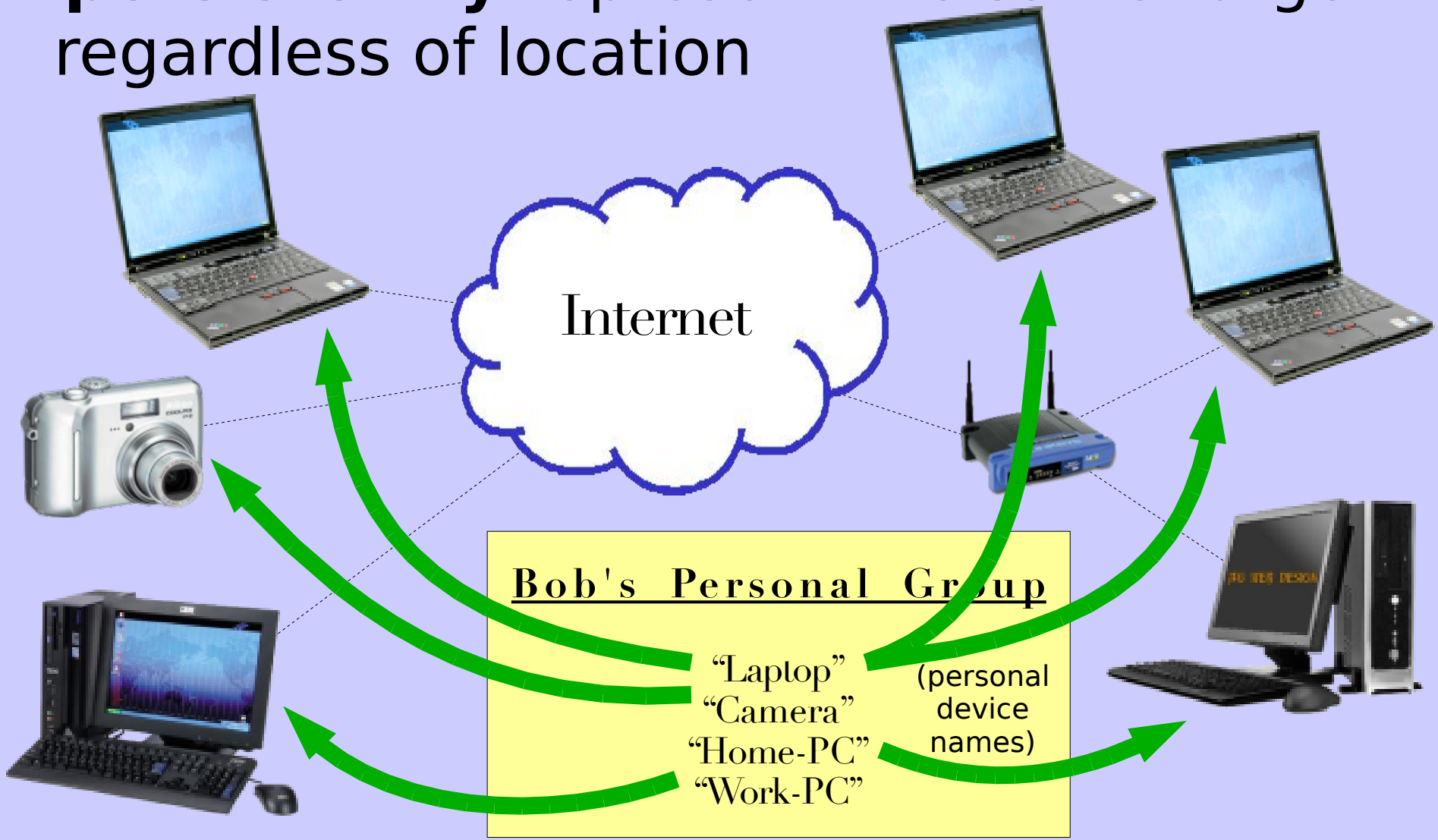
- Perfect when global usability is the point
- Expensive, cumbersome in personal context

Personal names:

- *Not* globally unique, *thus* short, convenient
- *…but* still usable for global connectivity!

Personal Names

...**persistently** represent the same target regardless of location



How to Build Personal Groups?

Convenience goal precludes:

- assigning or entering IP addresses, MAC addresses, ...
- generating or distributing crypto keys, certificates

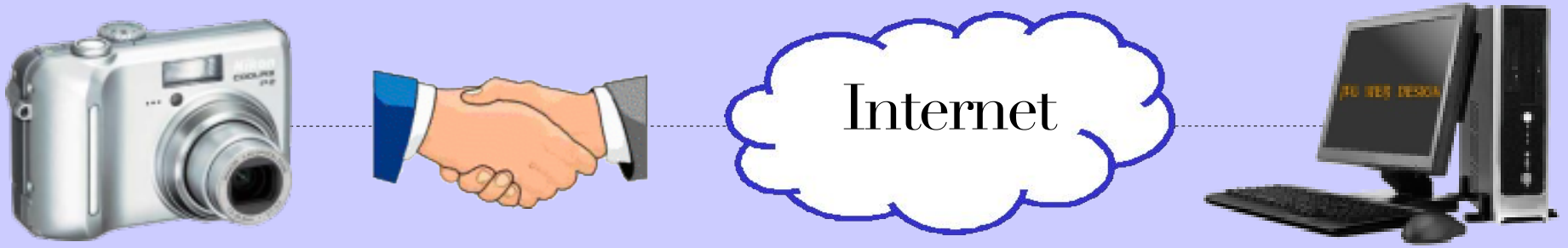
Name Bootstrap Problem:

- How to securely indicate device to be named, *without* referring to low-level identifiers?

Building Groups via Introduction

Common case: **local**, on home/office LAN

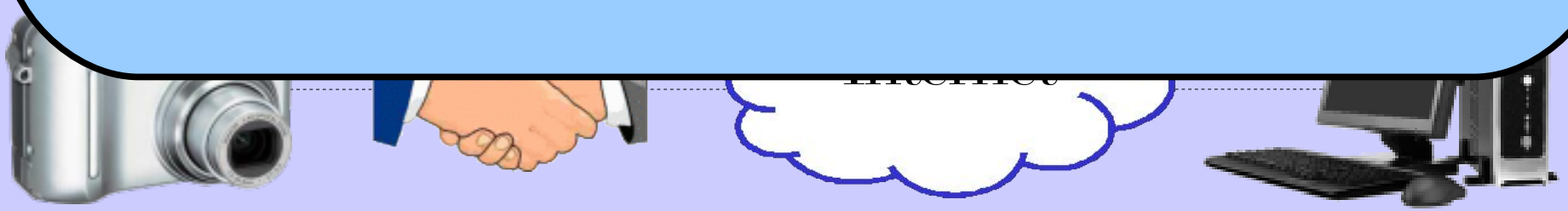
Also supported: **remote**, via global names



“bobs-pc.workplace.com”

Building Groups via Introduction

use
Device Mobility
to build a
Global Naming Federation
from
Local Pairwise Introductions

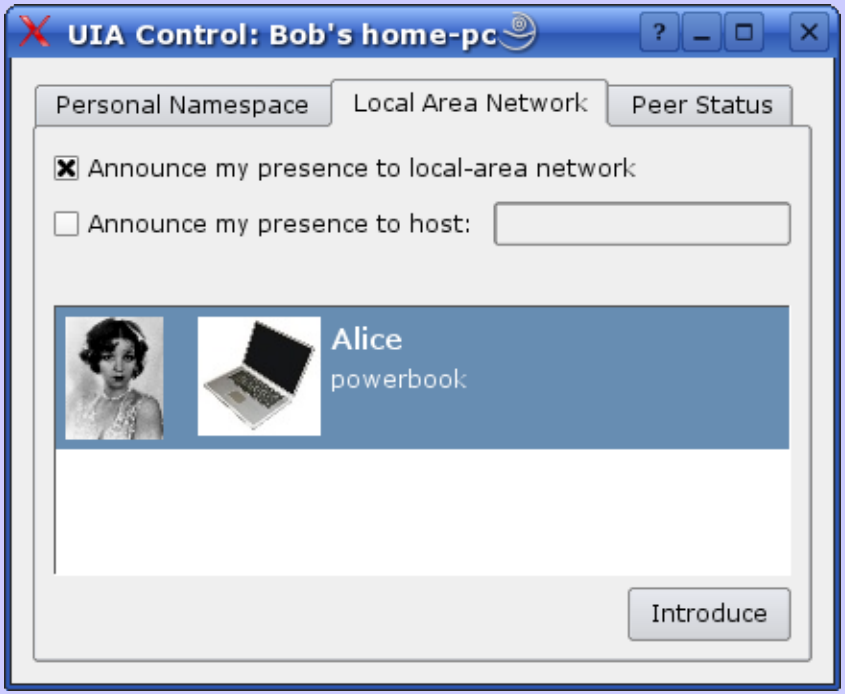


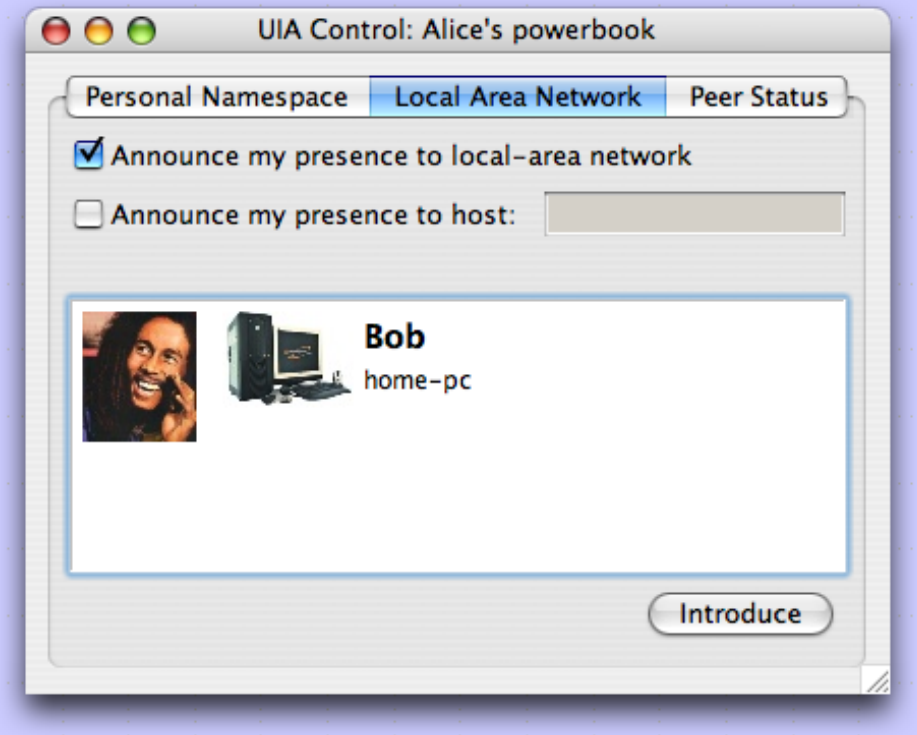
"bobs-pc.workplace.com"

UIA Introduction Procedure

2-step process:

1. Identify other device locally [Bonjour]
2. Avoid MITM attacks [Dohrmann/Ellison]



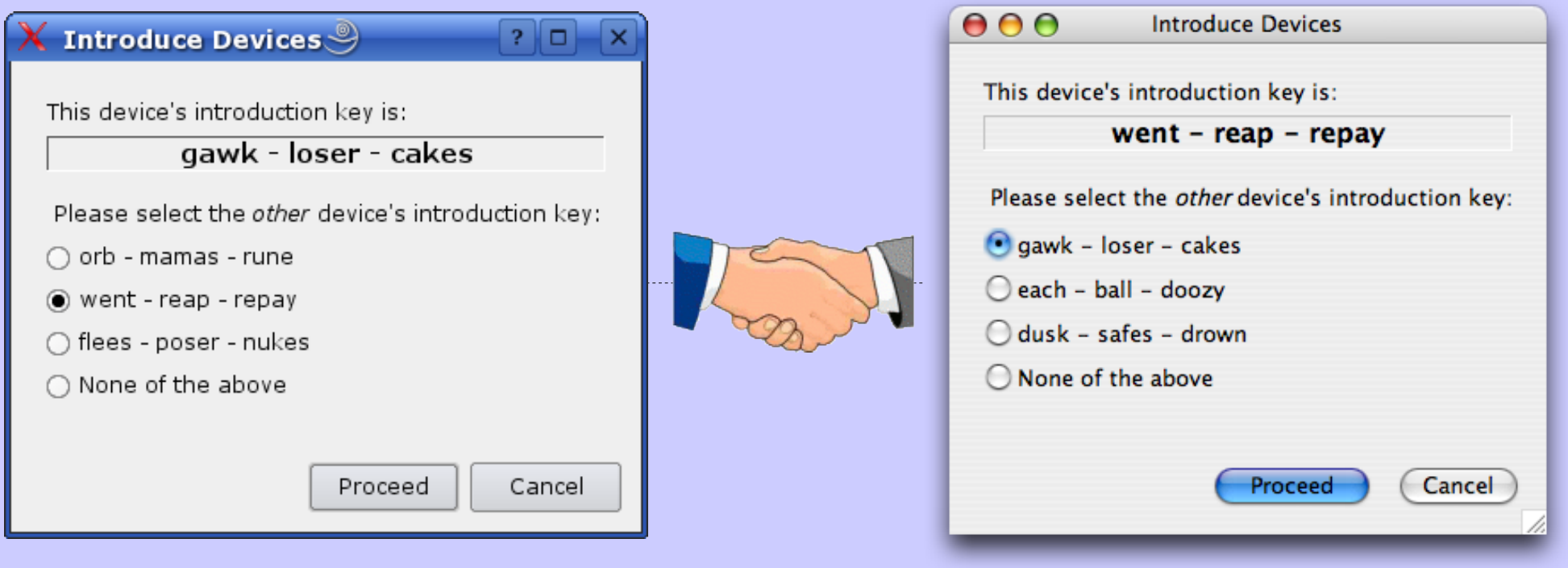


(screen shots from working UIA prototype)

UIA Introduction Security

Refines prior introduction protocols

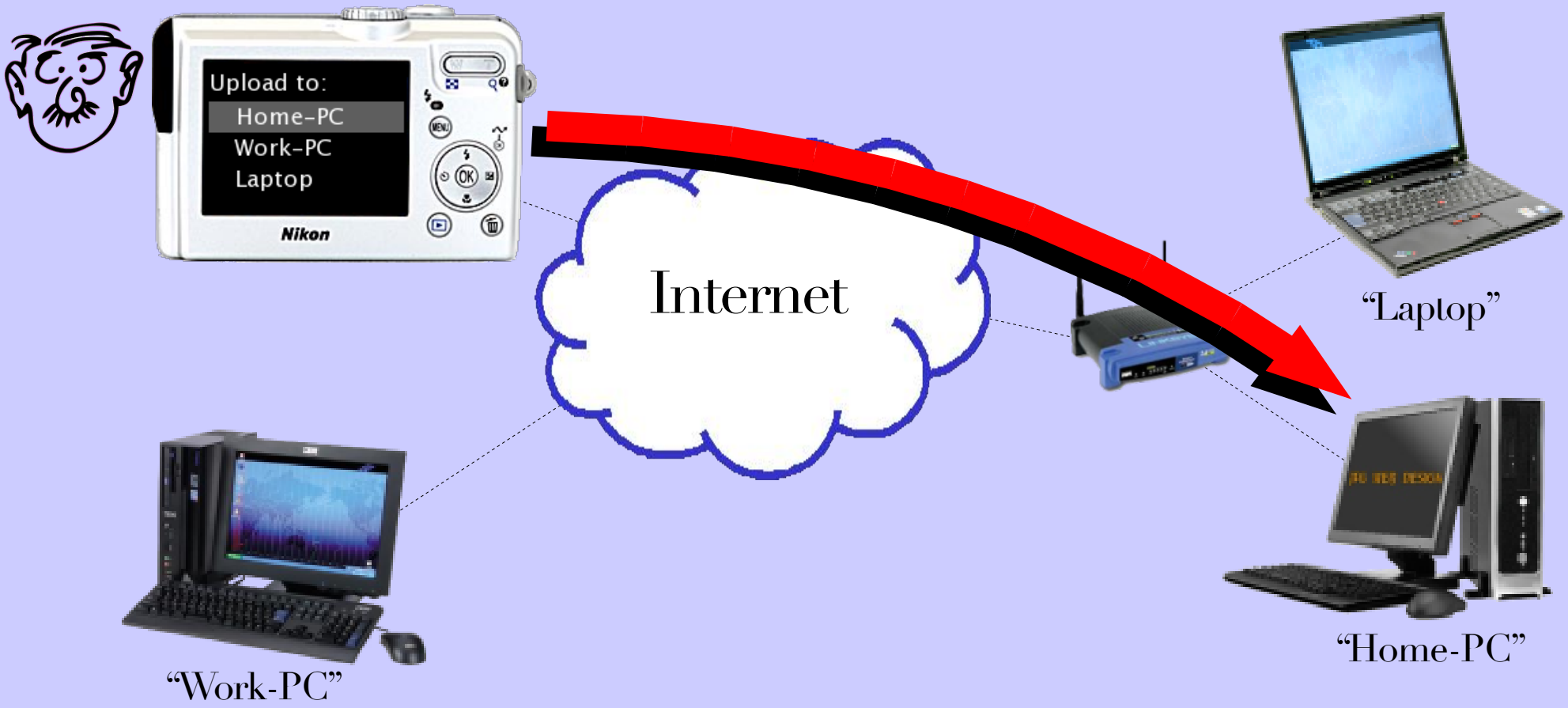
- Online protocol: resist attacks with fewer bits
- Multiple-choice: ensures user participation



But many other schemes possible! [MyNet]

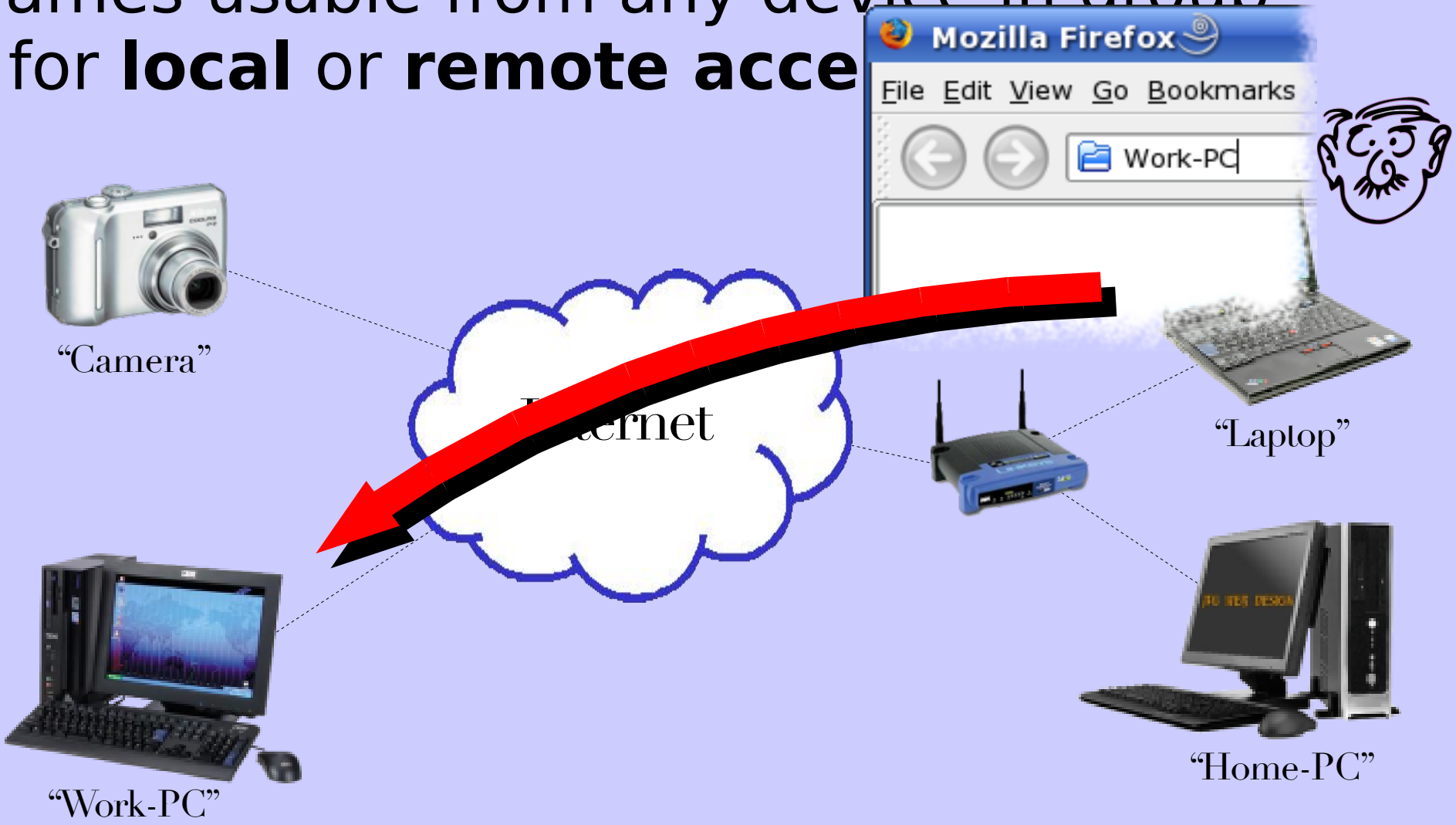
Remote Access

Names usable from any device in group for **local** or **remote access**



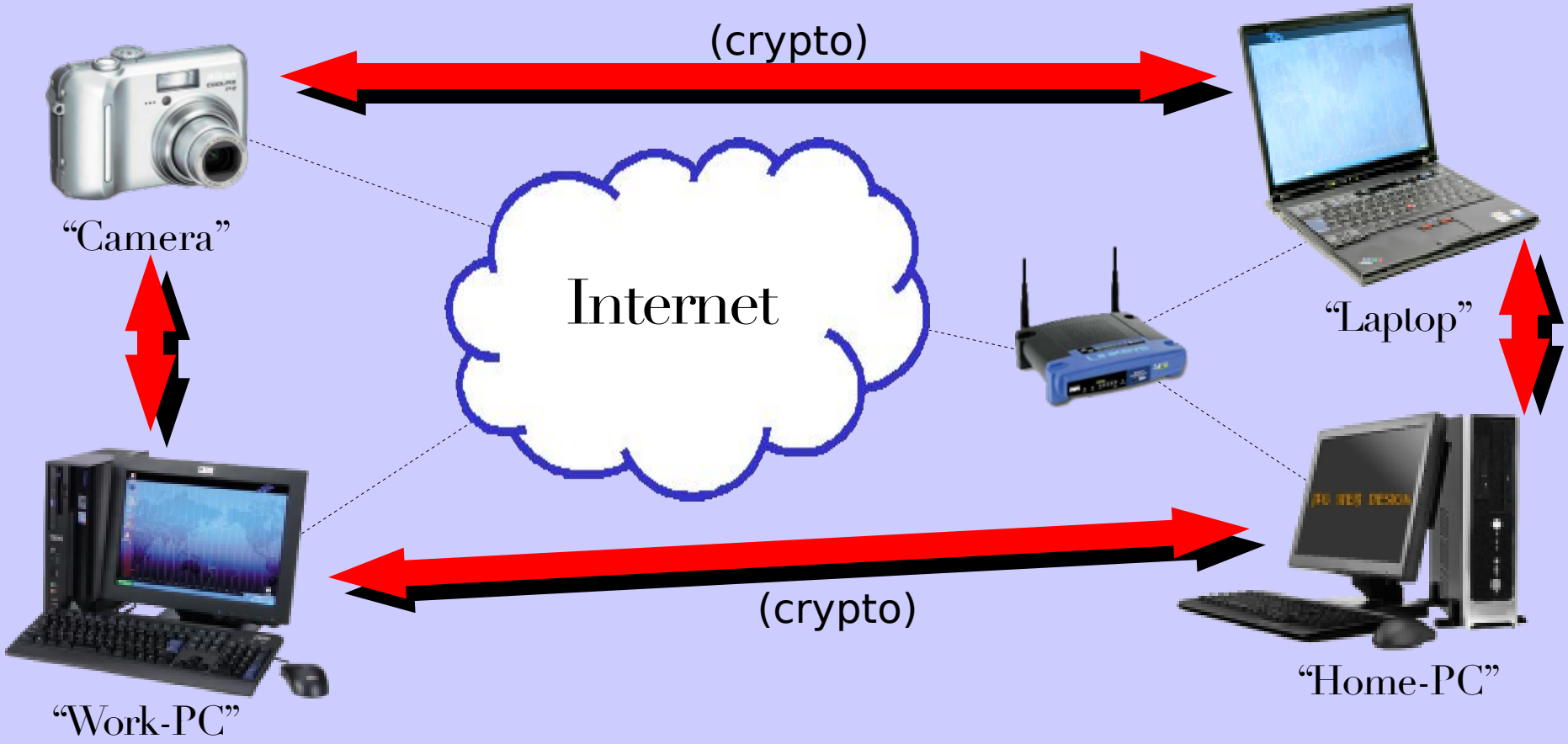
Remote Access

Names usable from any device in group
for **local** or **remote acce**



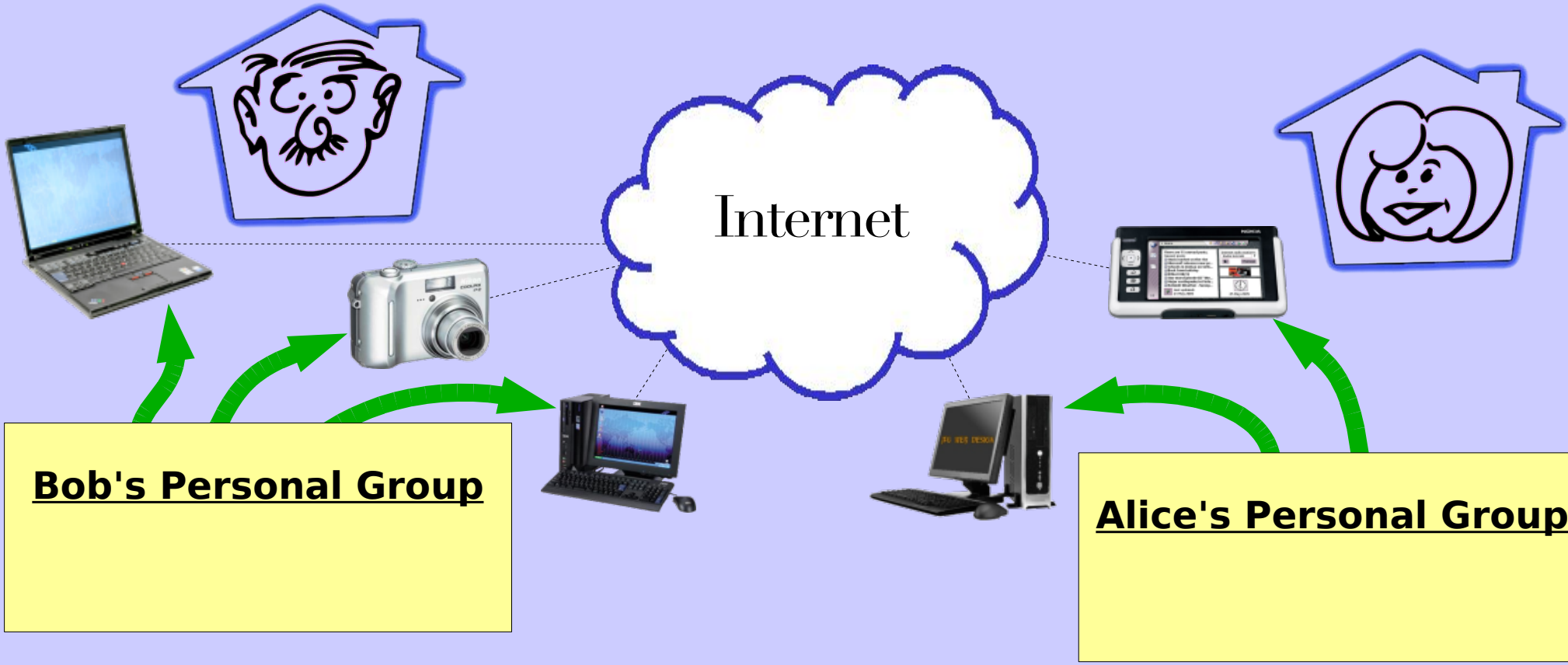
Security

All communication **privacy-protected**
as in virtual private network (VPN)



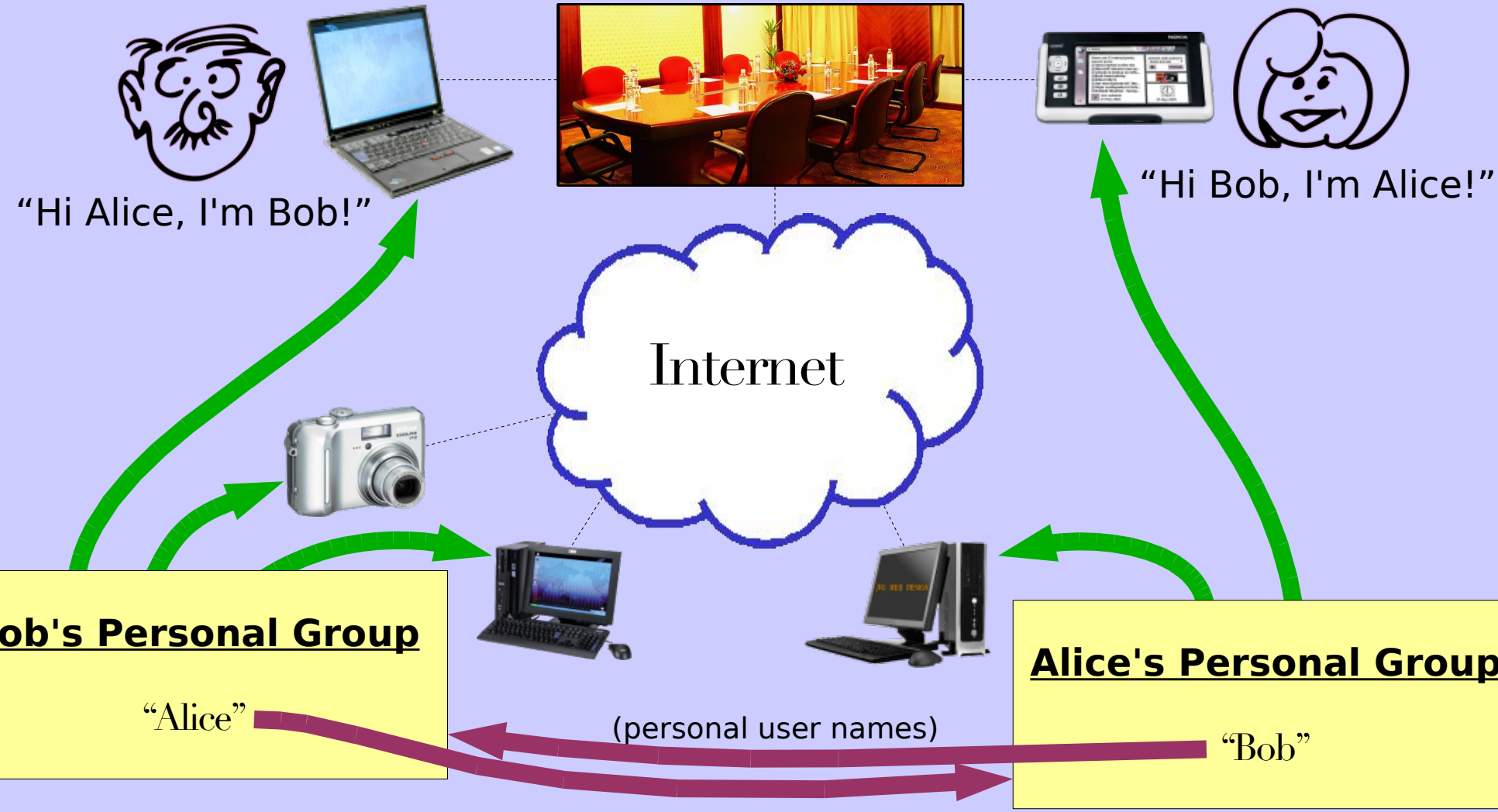
Social Networking

Personal group provides **user identity**



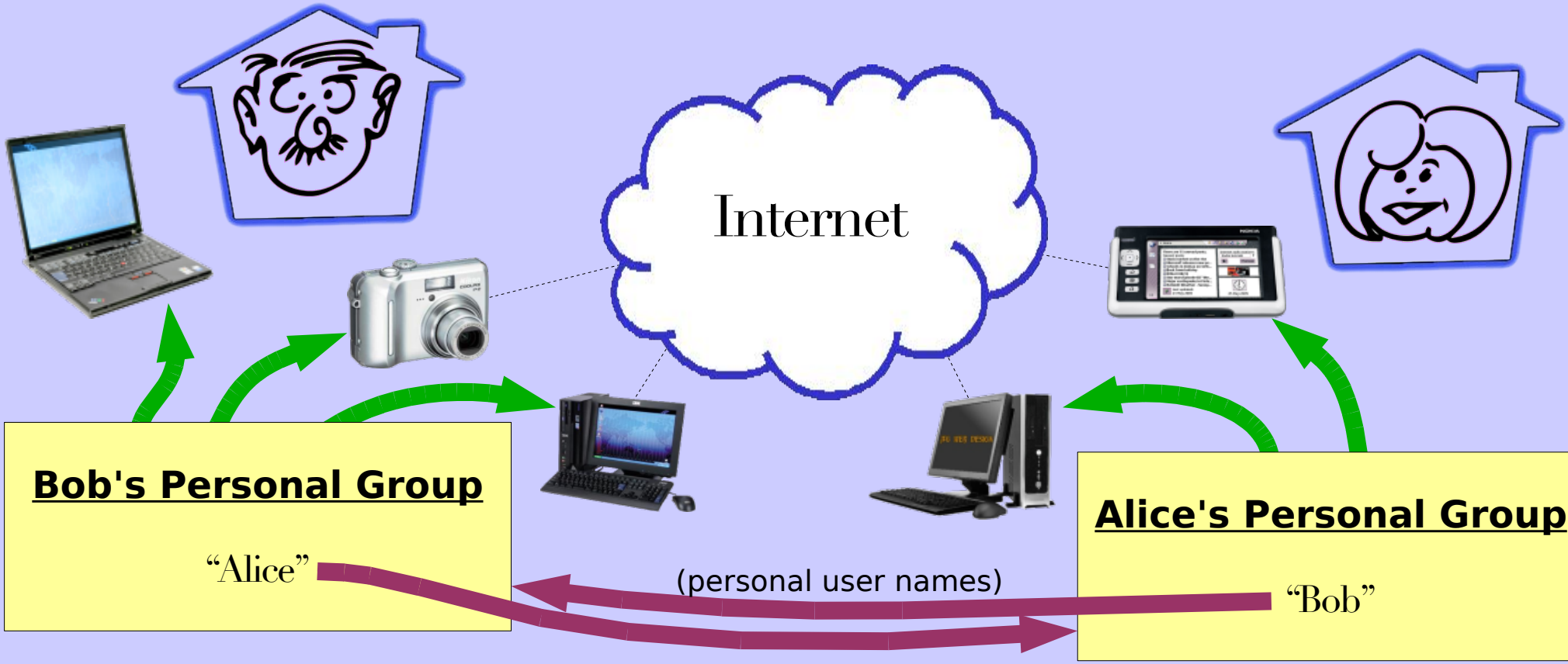
Social Networking

Personal group provides **user identity**



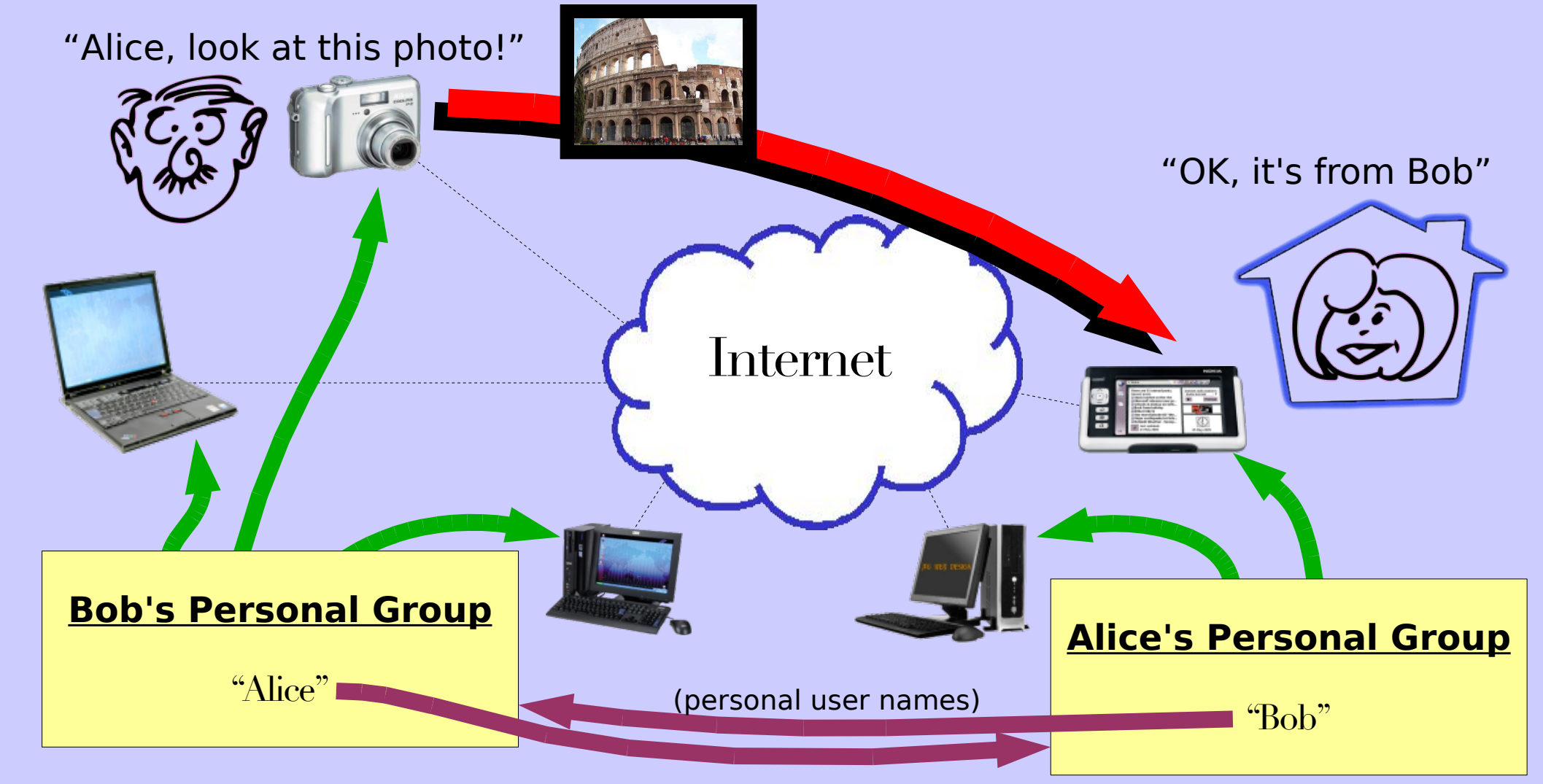
Social Networking

Personal user names also **persist**



Social Networking

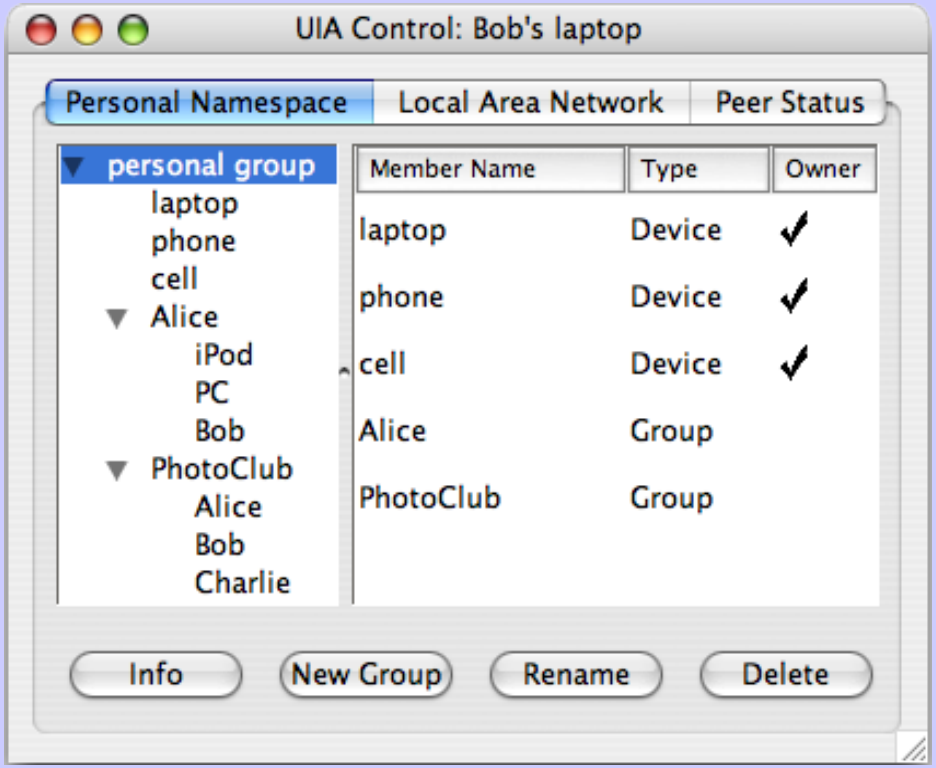
all devices in group **represent same user**

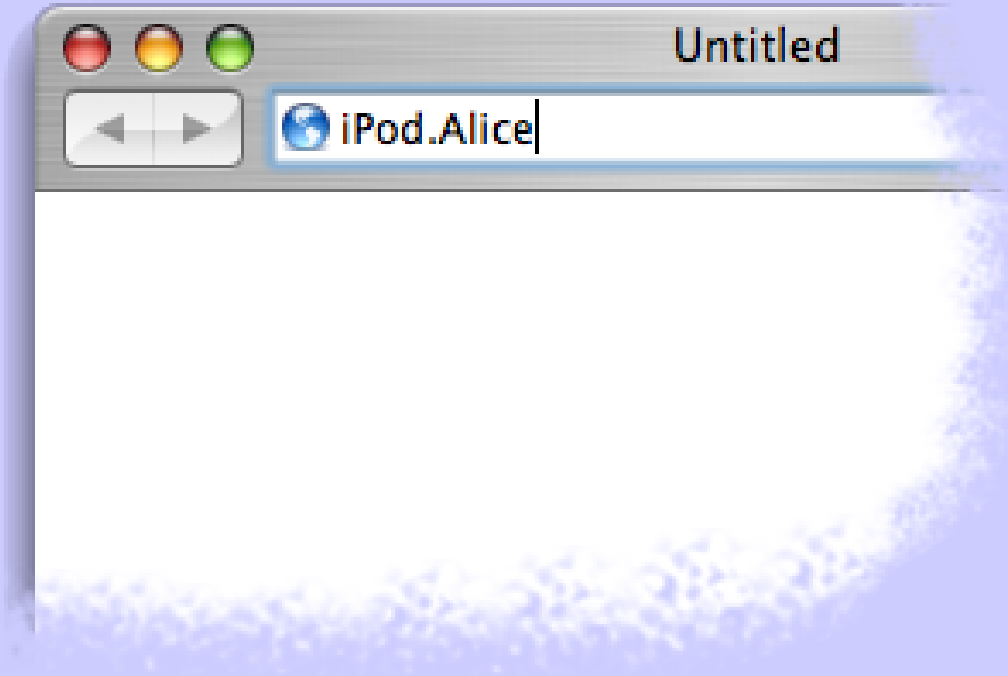


Using Personal Groups/Names

Browse groups, control access

Enter user-relative domain names





Implementing Personal Groups

...while maintaining
consistency and **availability**
in a fully decentralized design

Key Technical Challenges

- Device Location Independence
- Network Partition Tolerance
- State Synchronization, Consistency
- Distributed Ownership, Revocation

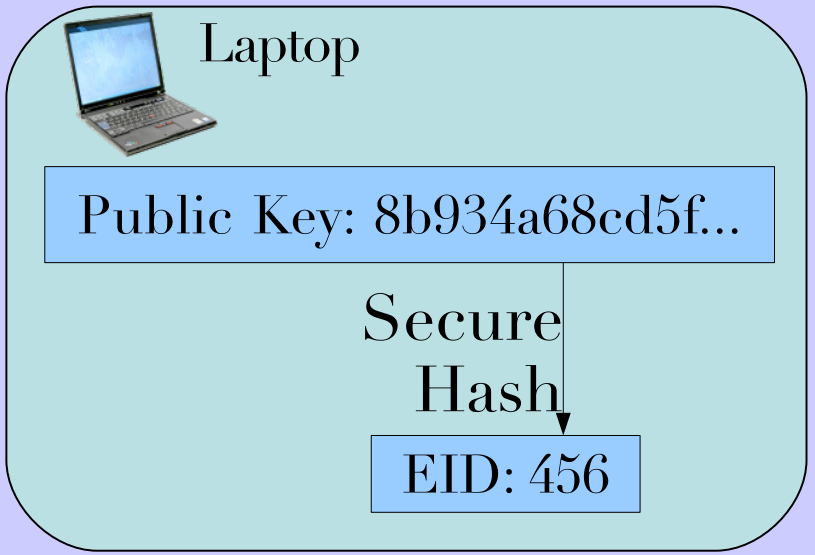
Challenge:
Location Independence

How to identify personal devices
as they move, change IP addresses?

Solution: Endpoint Identifiers

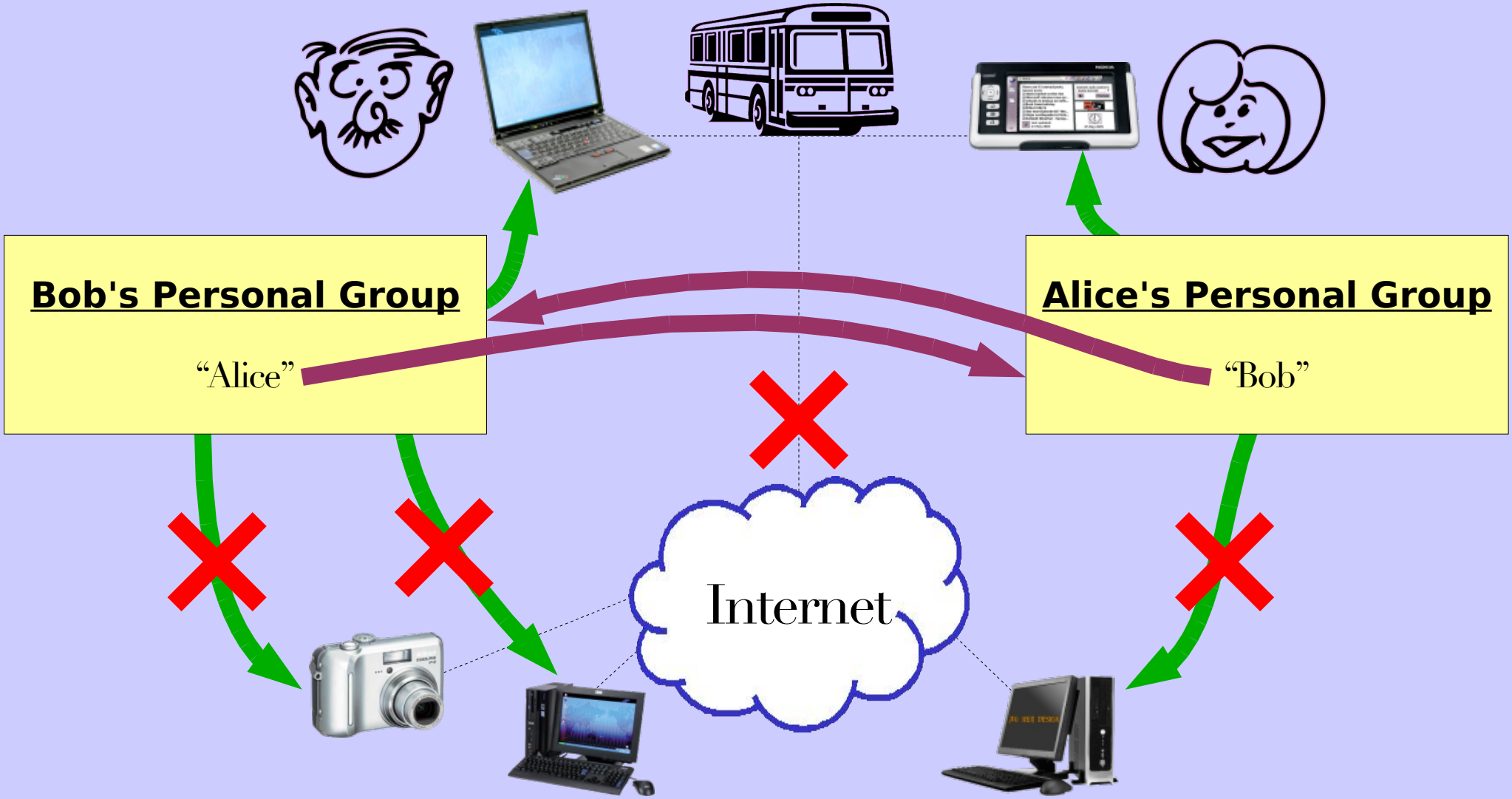
Each device has ***endpoint identifier*** (EID)

- Hash of device's public key [SFS]
- Self-configured, stable, location-independent [HIP]



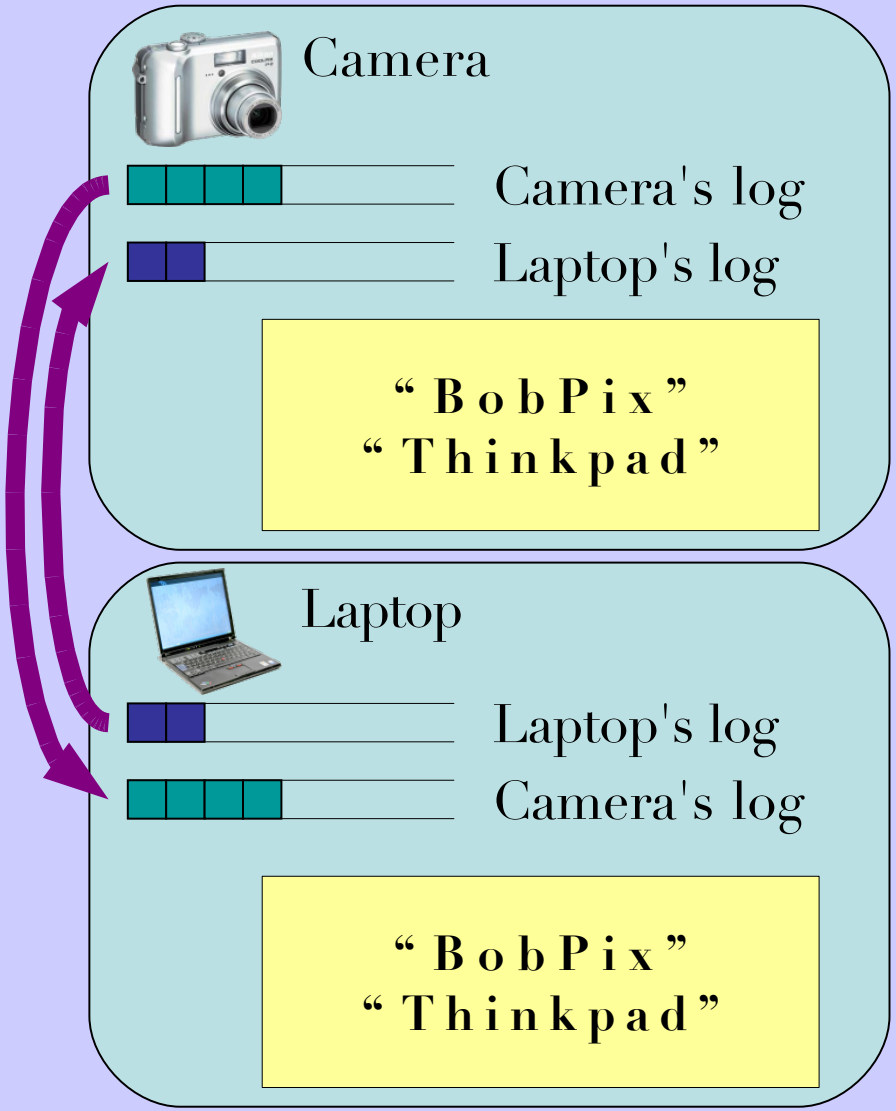
Challenge: Partition Tolerance

Names must keep working **off-Internet**



Solution: State Replication

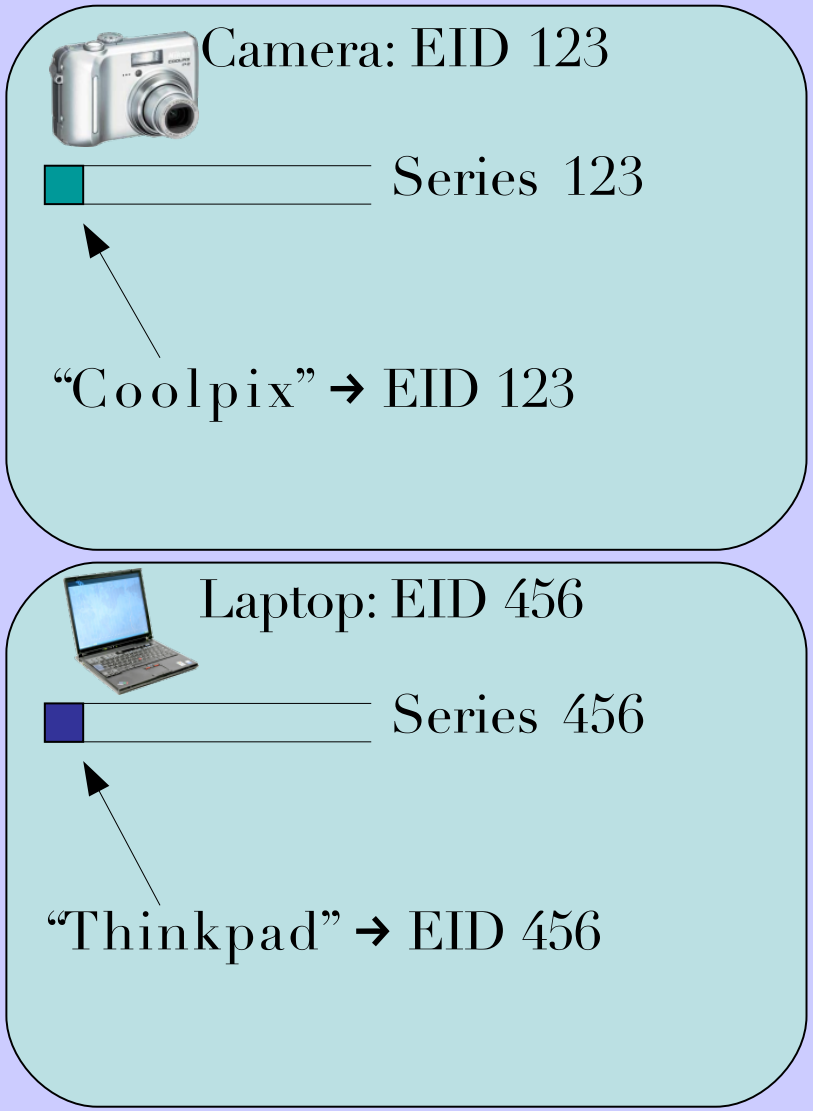
- Each device keeps **change log**
- Grouped devices **replicate** each others' state
- Log entries are **self-certifying**, **fork-consistent**



Implementing Names and Groups

Device keeps a *series* of change records

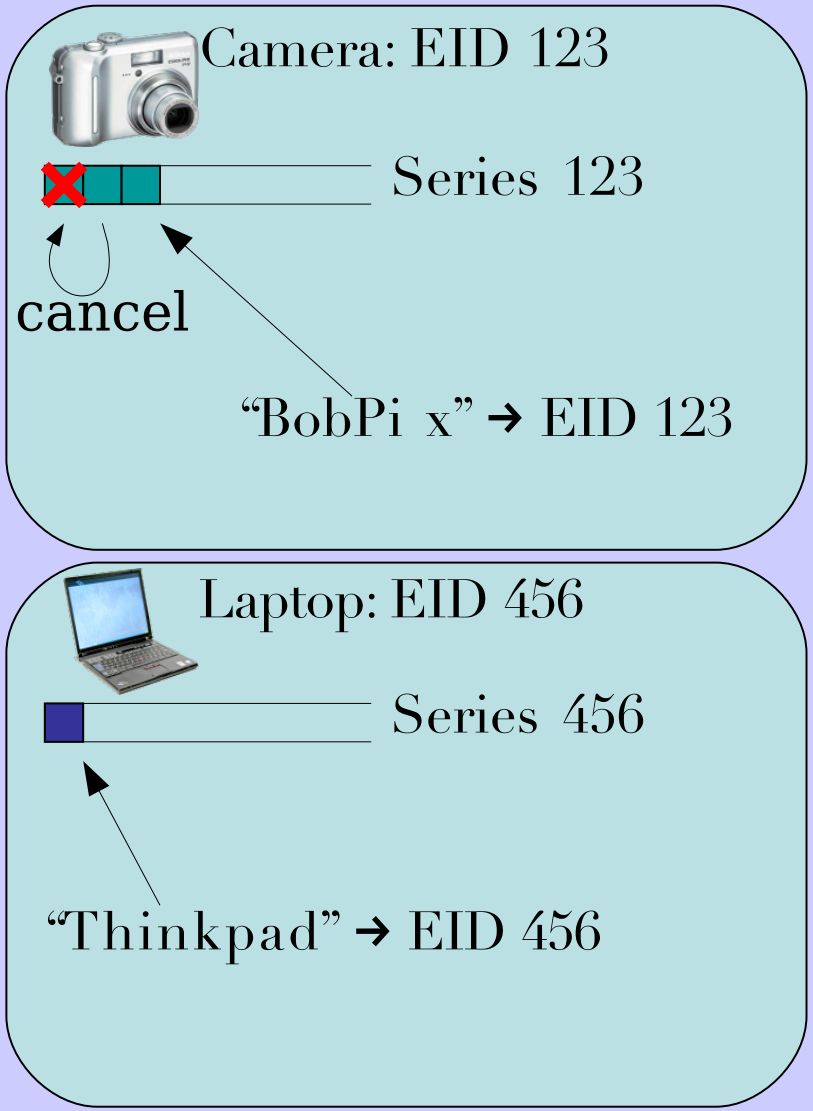
- Start with default name



Implementing Names and Groups

Device keeps a *series* of change records

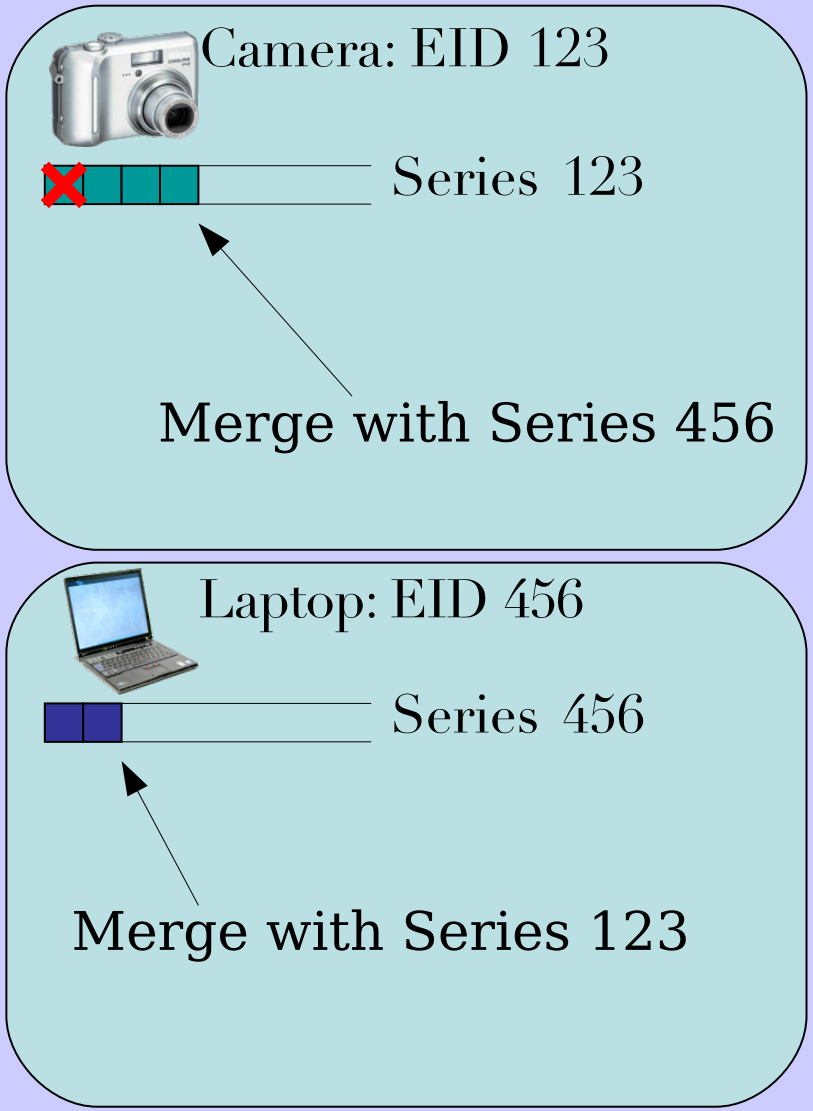
- Start with default name
- To rename: cancel old, write new name record



Implementing Names and Groups

Device keeps a *series* of change records

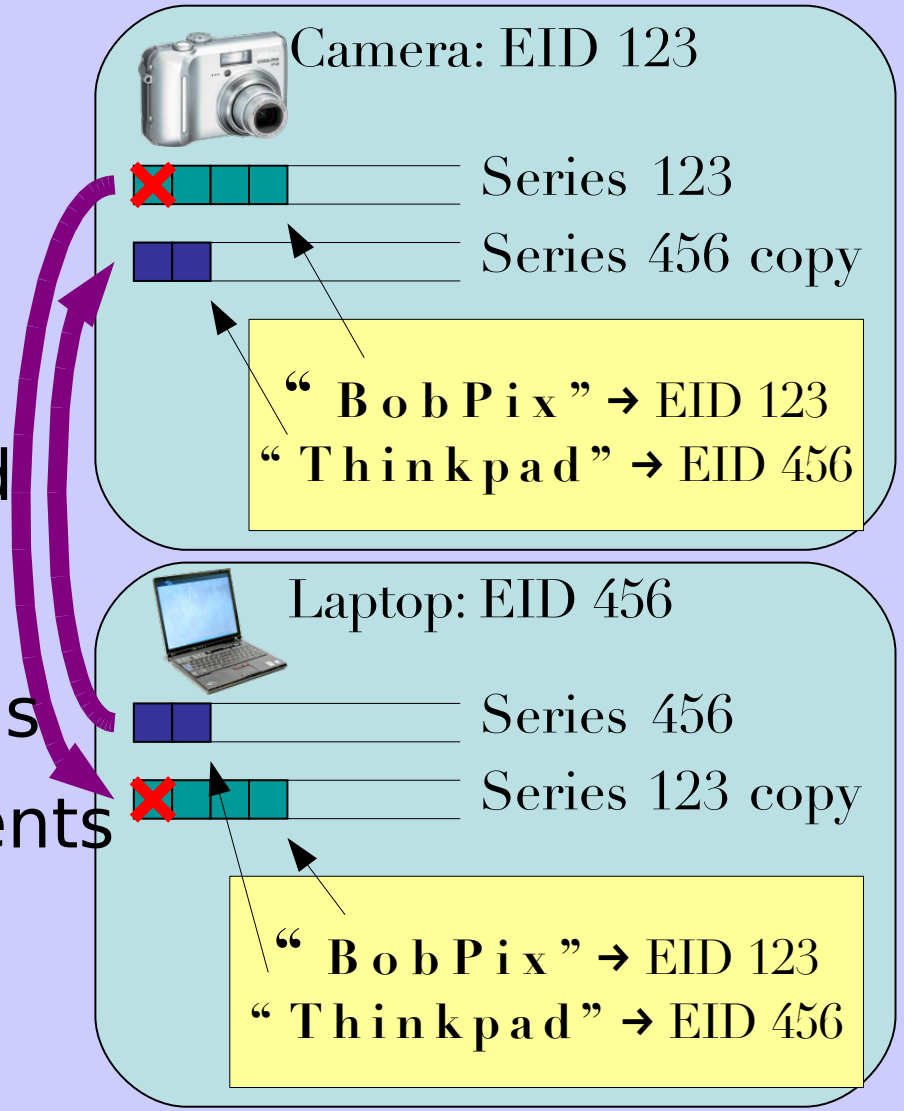
- Start with default name
- To rename: cancel old, write new name record
- To merge:
  - Write merge records



Implementing Names and Groups

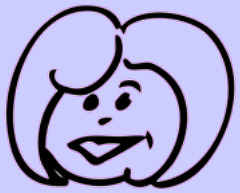
Device keeps a *series* of change records

- Start with default name
- To rename: cancel old, write new name record
- To merge:
  - Write merge records
  - Gossip series contents



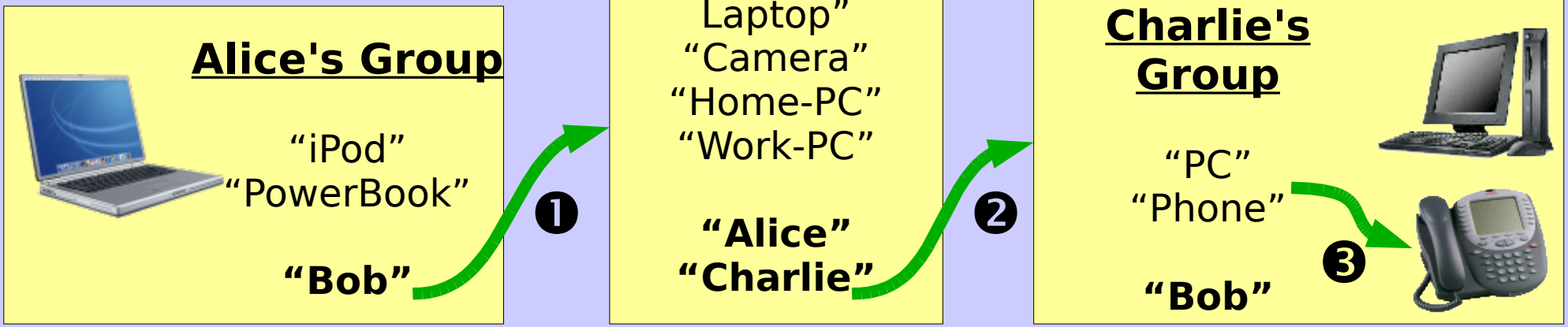
Serverless Name Resolution

- Use replicated state – *no communication*
- Resolution starts in device's own group
- Resolve components right-to-left



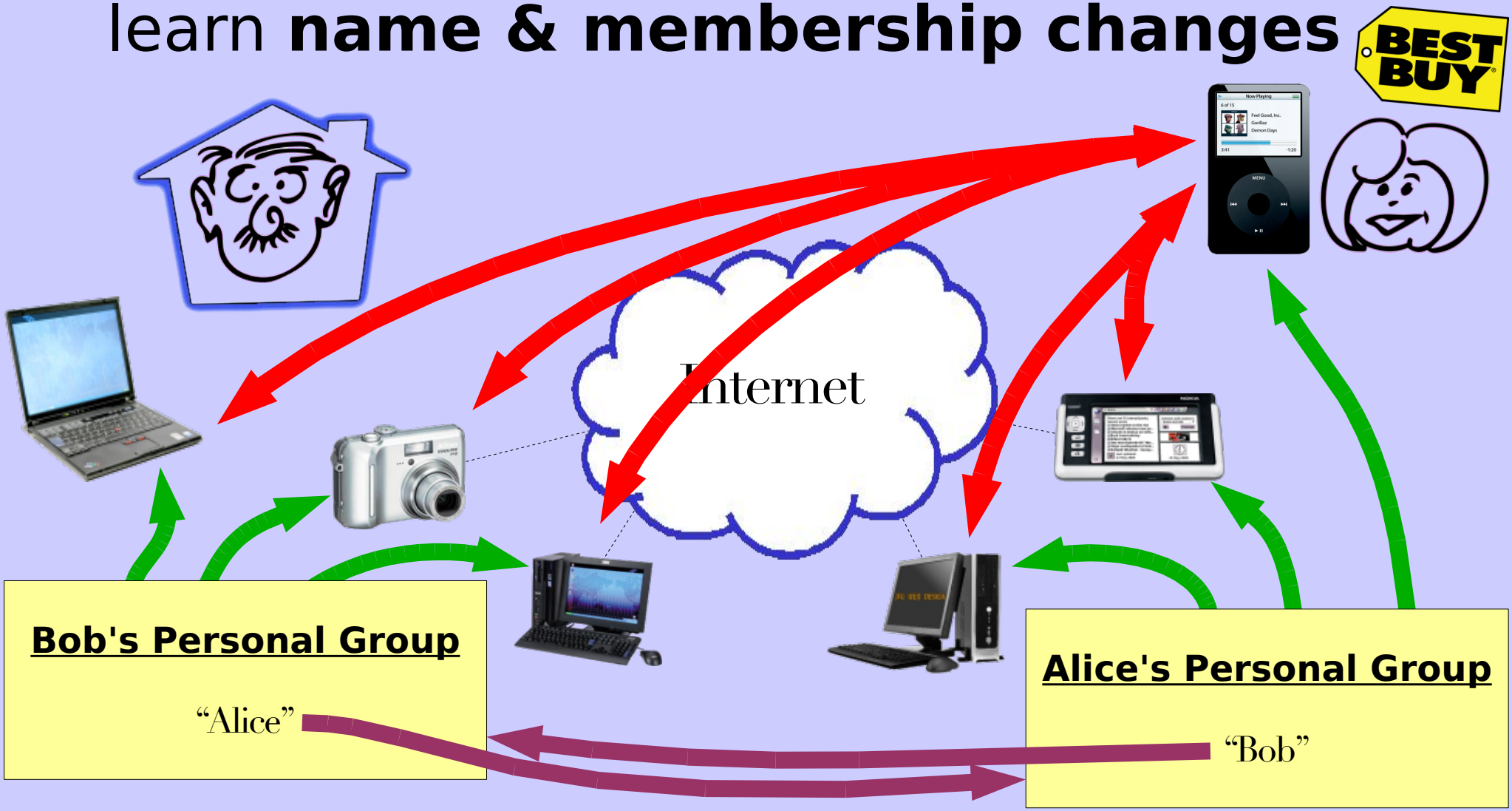
Phone.Charlie.Bob

❸ ❷ ❶



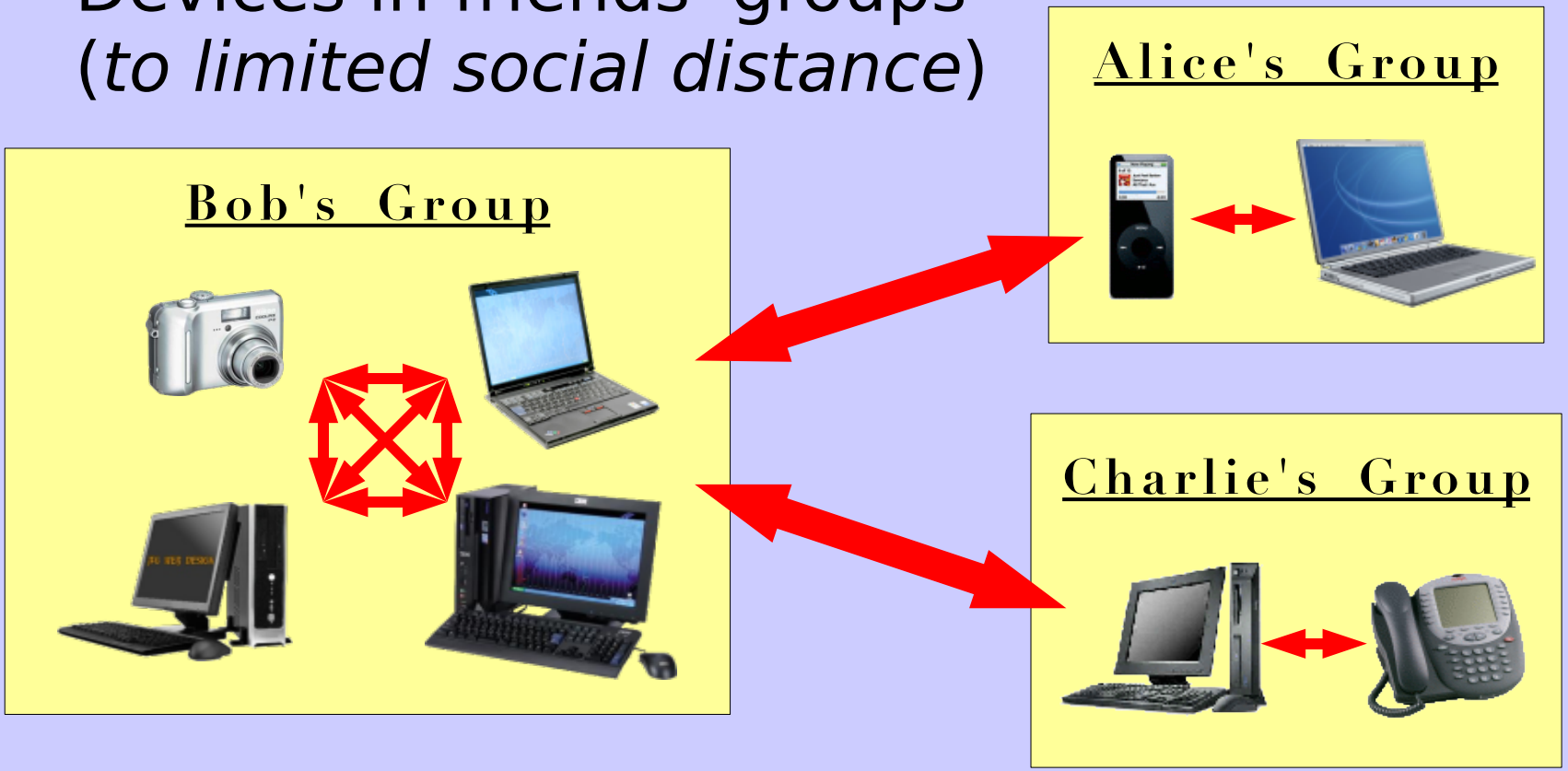
Challenge: Consistency

All devices in group must automatically learn **name & membership changes**



Solution: Change Record Gossip

- Devices gossip whenever possible with
  - Other devices in personal group
  - Devices in friends' groups (*to limited social distance*)



Name Conflicts

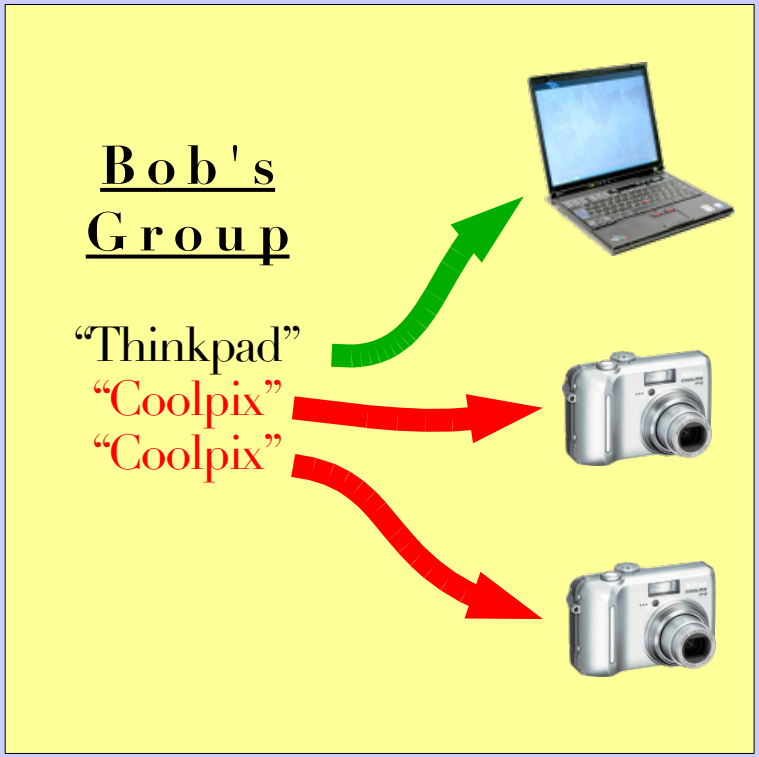
What if user groups two devices w/ same name?

⇒ merge succeeds, but **creates conflict**

(can't use name)

Resolve by renaming

(on either device)



Name Conflicts

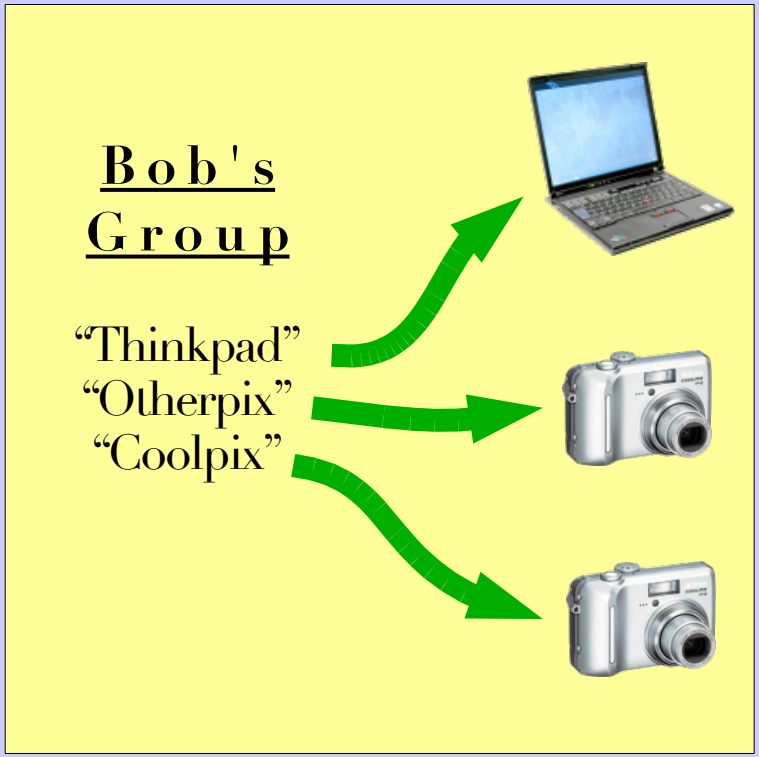
What if user groups two devices w/ same name?

⇒ merge succeeds, but **creates conflict**

(can't use name)

Resolve by renaming

(on either device)



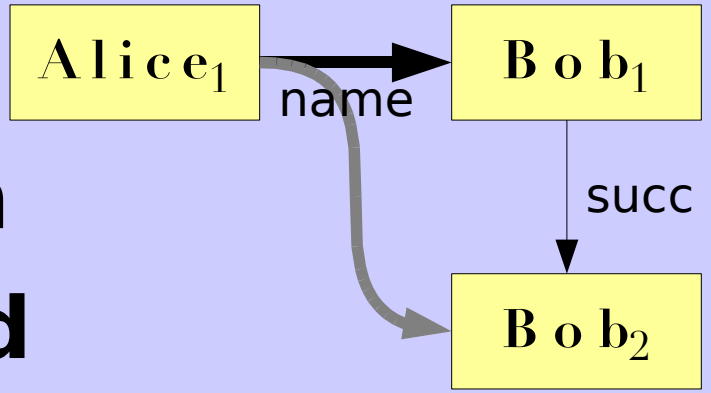
Challenge: Ownership, Revocation

- Key problem:
  - **Access control** depends on **membership**, **membership changes** depend on **access**
  - Devices can't tell **true owner** from **thief**
  - Maintain device/group **availability** even under **lack of consensus**

Solution: Group Versions, Successorship

On revocation:

- create new group **version**
- write **successor record** in old version

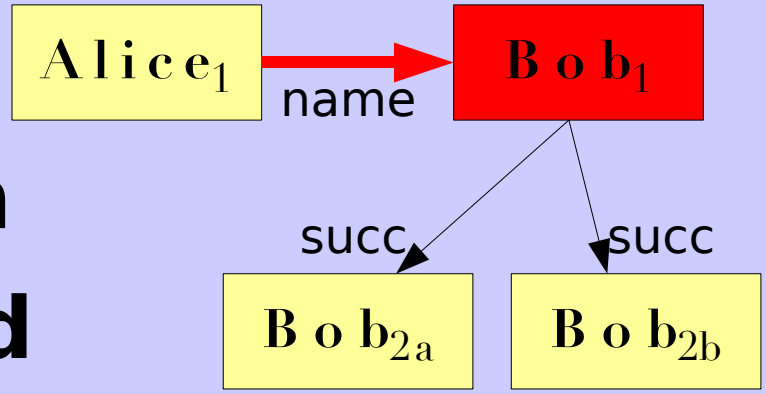


One “head” → **OK**

Solution: Group Versions, Successorship

On revocation:

- create new group **version**
- write **successor record** in old version



One “head” → **OK**

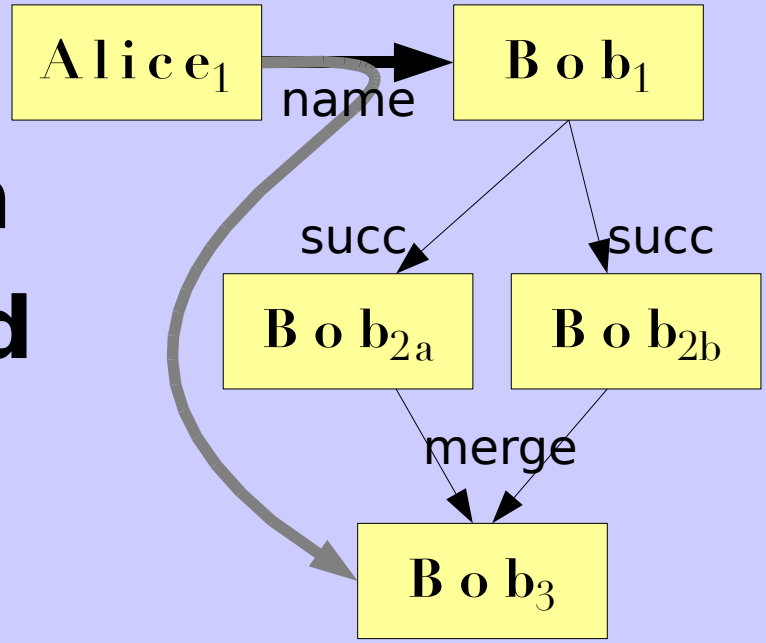
Multiple “heads” → **ownership conflict**

Resolve conflicts by:

Solution: Group Versions, Successorship

On revocation:

- create new group **version**
- write **successor record** in old version



One “head” → **OK**

Multiple “heads” → **ownership conflict**

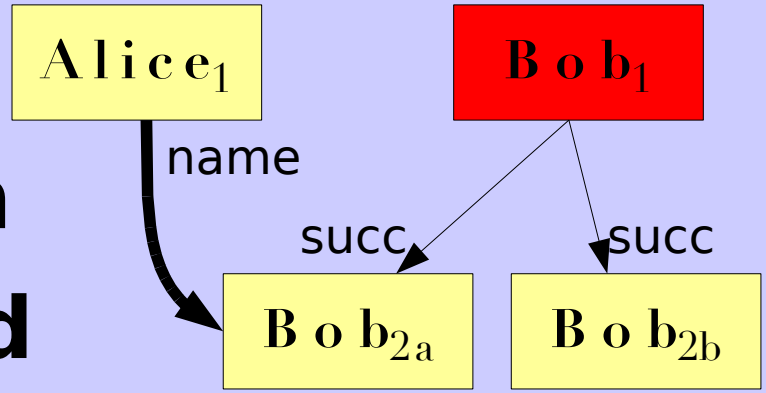
Resolve conflicts by:

- merging heads

Solution: Group Versions, Successorship

On revocation:

- create new group **version**
- write **successor record** in old version



One “head” → **OK**

Multiple “heads” → **ownership conflict**

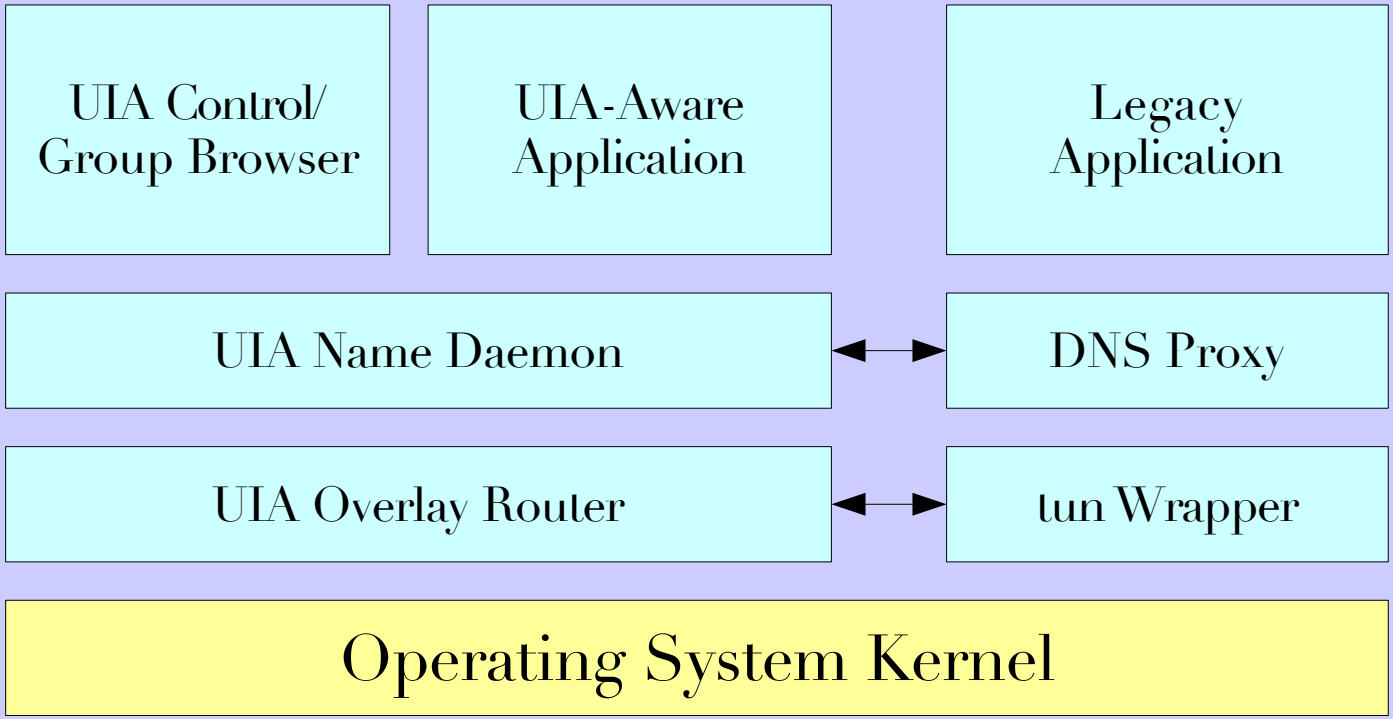
Resolve conflicts by:

- merging heads
- re-introducing friends

Implementation Status

"Version 1" prototype:

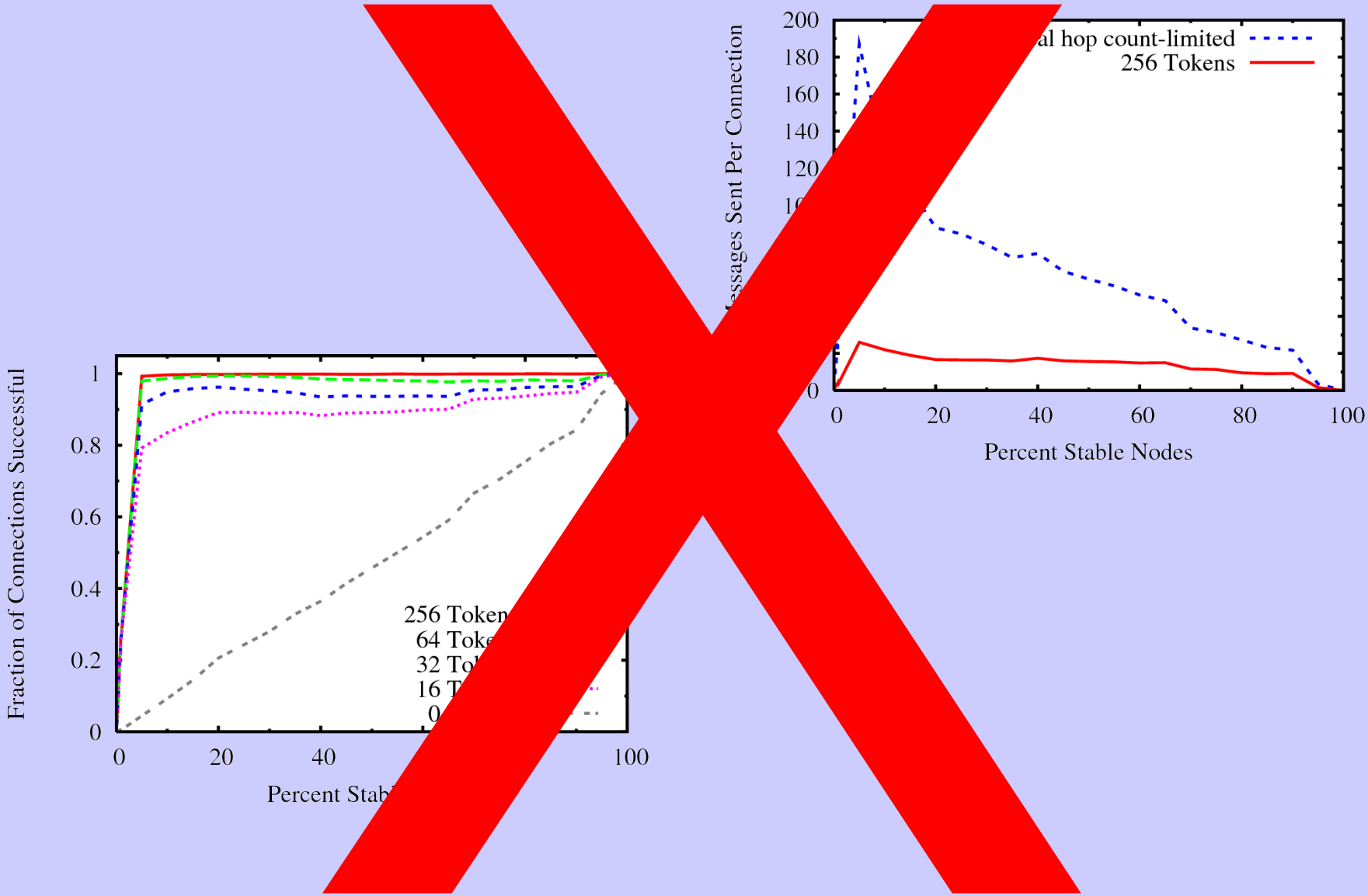
Runs on Linux, Mac OS X, Nokia Internet Tablet



Implementation Status

- “Version 2” prototype under development
  - More robust ownership/revocation algorithm
  - Scalable routing protocol (compact routing)
  - Structured stream transport (SST) integration
  - Fewer dependencies, easier to install
  - …

Evaluation


Fraction of Connections Successful
1
0.8
0.6
0.4
0.2
0
0
20
40
100
Percent Stab
256 Token
64 Toke
32 To
16 T
0
Messages Sent Per Connection
200
180
160
140
120
0
al hop count-limited
256 Tokens
0
20
40
60
80
100
Percent Stable Nodes

[Video]

Implementation Observations

Proof-of-concept prototype

- Works, many rough edges...

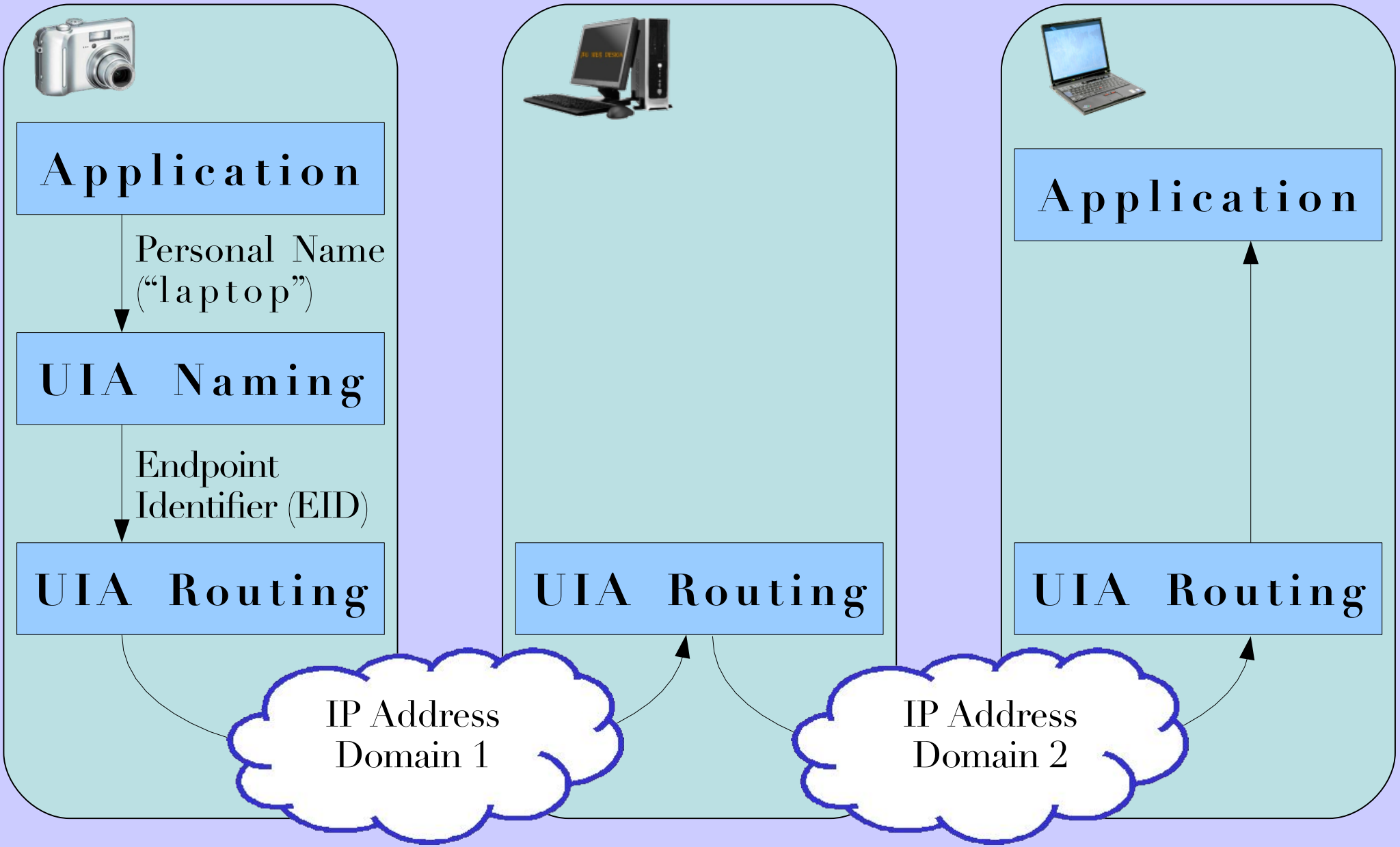
But demonstrates the architecture

- Logs not too big: ~40K in example
  - Small name records, infrequent changes
- Router tables, overhead not too large
  - Only track “social neighbors”, not whole world

Routing

(brief summary)

Routing to Personal Devices

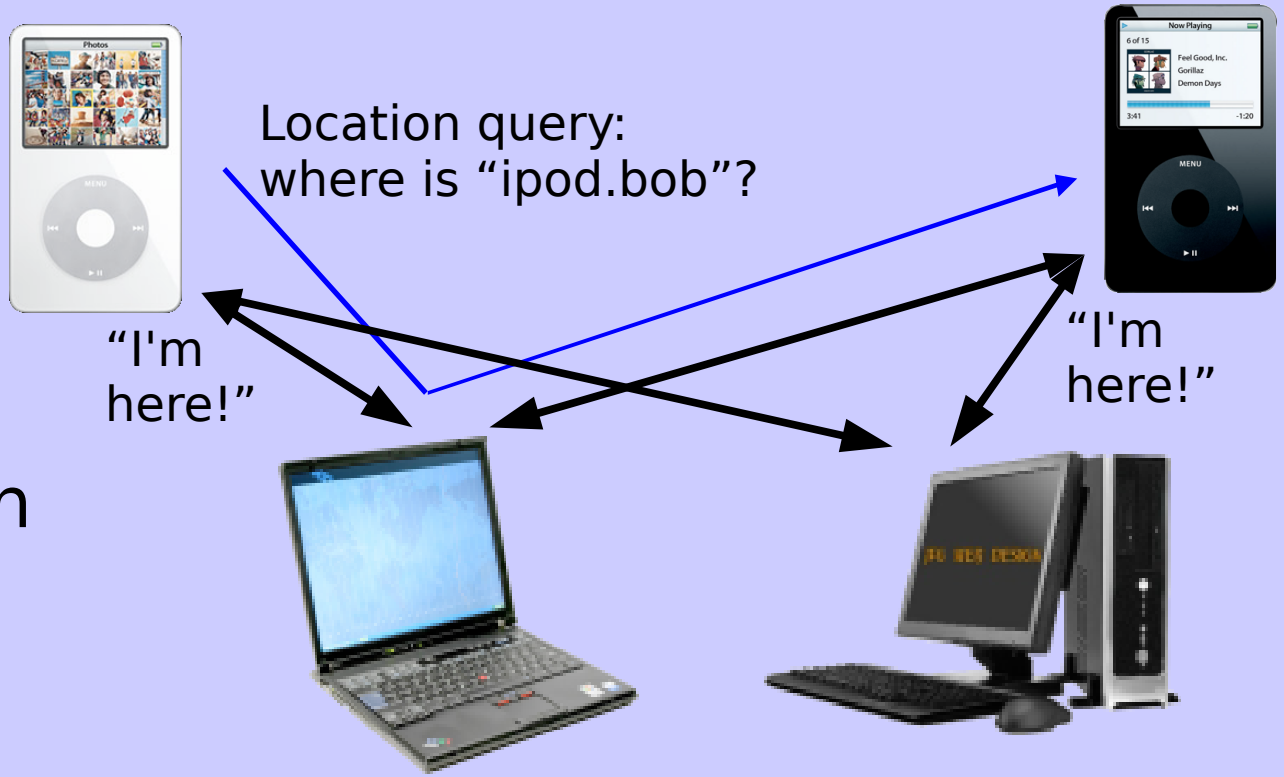


Routing Requirements

- Challenges:
  - Avoid management by users
  - Handle mobility, network partitions
  - Minimize overhead
- Opportunities:
  - Use global Internet when available
  - Use social network

Opportunistic routing via social networks

Gossip waypoint information



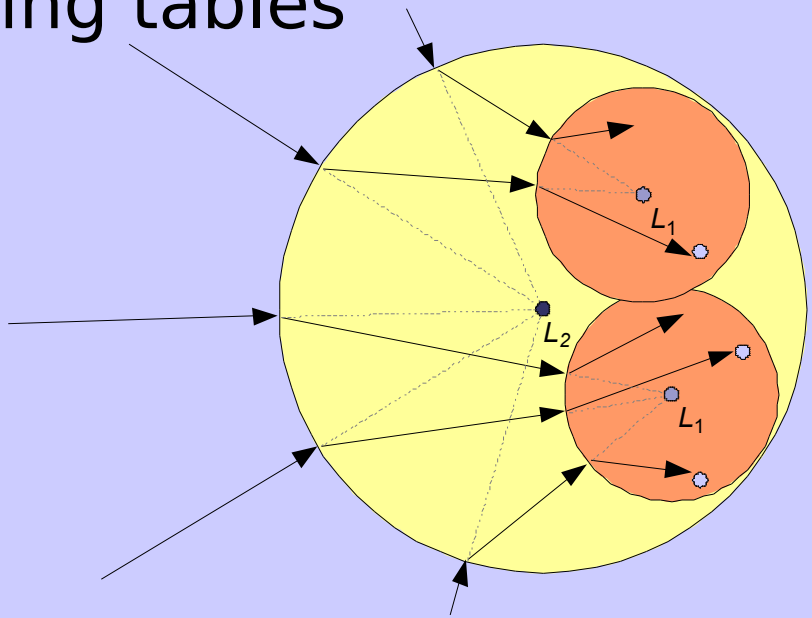
Simple, works when communication is between:

- User’s devices
- Immediate friends

“Persistent Personal Names for Globally Connected Mobile Devices”, *OSDI 2006*

Scalable compact routing

Provable stretch, small routing tables
[TZ 2001, etc.]



Extend TZ to:

- be a distributed protocol
- limit path congestion
- provide fault tolerance

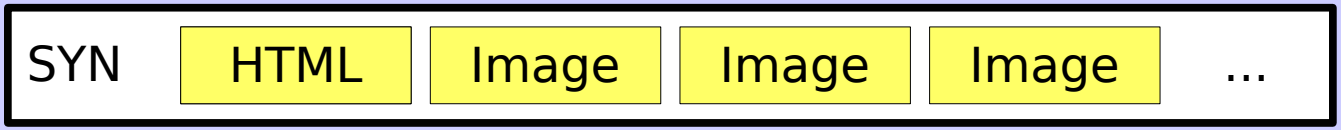
Work in Progress

Transport

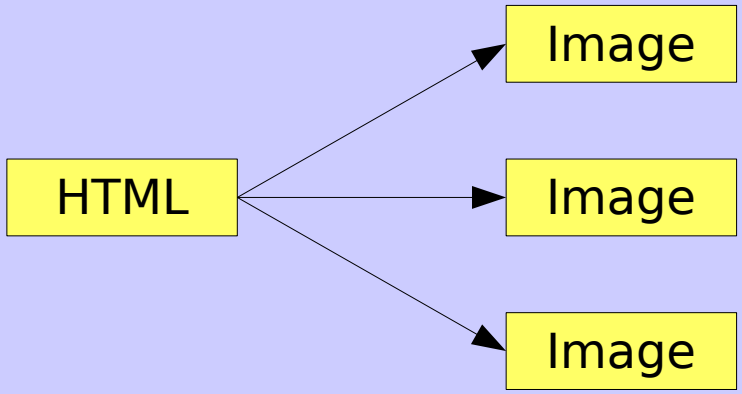
(brief summary)

Problem

TCP designed for **serial** operation



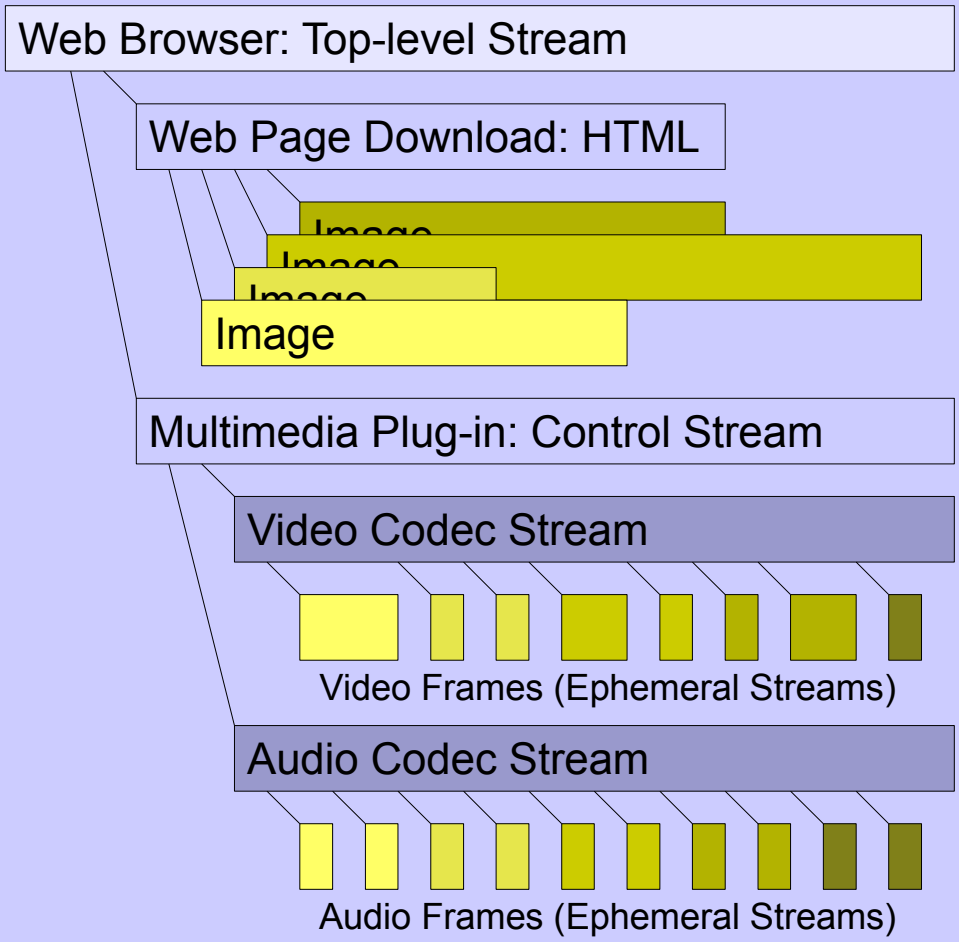
Modern interactive apps are **parallel**



Structured Stream Transport

Supports *efficient, short-lived streams*

- Stream “**fork**” operation
- No handshake, quick shutdown
- Subsumes **datagrams**



Benefits of SST

Ex. HTTP over SST: *more responsive*

- No unnecessary request serialization
- **Fork** provides out-of-band communication

⇨ ***Dynamically prioritize*** requests

(Demo)

Related Work

Dynamic DNS, Mobile IP, IPSEC VPNs

Decentralized security: SDSI/SPKI

Host identities: SFS, HIP, JXTA, i3

Naming/routing: DDNS, TRIAD, i3, CoDoNS

Optimistic replication: Ficus, Coda, Ivy

Mobile data: Rumor, P-Grid, Roma, Footloose

Social networking: Turtle, Sprout, F2F, Tribler

Conclusion

UIA delivers **new network abstractions** for **tomorrow's personal devices**

- **Personal Groups, Personal Names** [OSDI '06]
- **Structured Streams** [SIGCOMM '07]
- ..and more...

http://pdos.csail.mit.edu/uia/

Acknowledgments

UIA Team
Jacob Strauss, Chris Lesniewski-Laas, Sean Rhea

Thesis Committee
Frans Kaashoek, Robert Morris, Hari Balakrishnan

Naming, Routing [OSDI '06]
Franklin Reynolds, MyNet Team – Nokia Research
Martín Abadi, Tom Rodeheffer – Microsoft Research

Transport [SIGCOMM '07]
Craig Partridge, Chip Elliott, Lars Eggert

NAT Traversal [USENIX '05]
Pyda Srisuresh, Dan Kegel, Henrik Nordstrom,
Christian Huitema, Justin Uberti, Mema Roussopoulos

Funding: NSF (Project IRIS, UIA), MIT/Quanta T-Party