UbiComp

Future Computing Environments

Supporting the Construction of Context-Aware Applications

Anind K. Dey

Intel Berkeley Lab

Intel Research

How UbiComp is Different from Traditional GUI

- Not only direct interaction for input and output
- Need to use RANGE of explicit AND implicit interaction
- Different computing paradigm

Context and Context-Awareness

- Focused on input
- Context: *any information relevant to an interaction that can be used to characterize the situation of an entity*

- Context-awareness
  - General model of interactive computing
  - Addresses subset of ubicomp problems: input

Value of Context

- Potential for improved usability
  - Very important for mobile users with poor input devices
- "Smarter" applications
- Increased communications bandwidth

Design Space for Context-Aware Applications

- Toolkit allows exploration of design space
- Basic types of context:
  - Location, identity, time, activity
  - Simple/singular → complex/multiple
  - Combinations
- Uses of context:
  - **Present** to user
  - Automatically perform set of **services**
  - **Tag** captured information to ease retrieval

Example

- Tour guides, travel assistants, personalization software

- Reminder to buy milk
  - When to deliver: not time/location specific
  - How to deliver: appropriate modality

Outline

- Motivation
- Problems dealing with context
- Contribution: Context Toolkit
- Validation:
  - Design space and applications
  - Building more realistic applications
- Conclusions and future work

Building Applications

- M. Weiser: The whole point of ubiquitous computing, of course, is the applications.

Building Applications

- M. Weiser: The whole point of ubiquitous computing, of course, is the applications.
- But … what if the applications are hard to build? And, what if this inhibits our ability to build compelling applications?

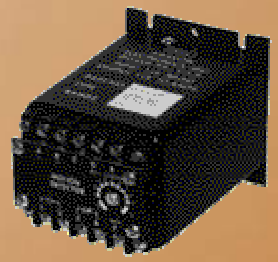
Issues in Context-Awareness

- What is context?
- Representation of context
- Application domains
- Which behaviors to support
- When to execute behaviors
- Privacy, Quality of Service, …
- Evaluation of applications

- make it easier to build ⏩ explore

Why Context is Hard to Use

- Acquired from sensors
  - Not just keyboards and mice – lots of heterogeneous devices
- Need to abstract data
- Distributed
- Dynamic



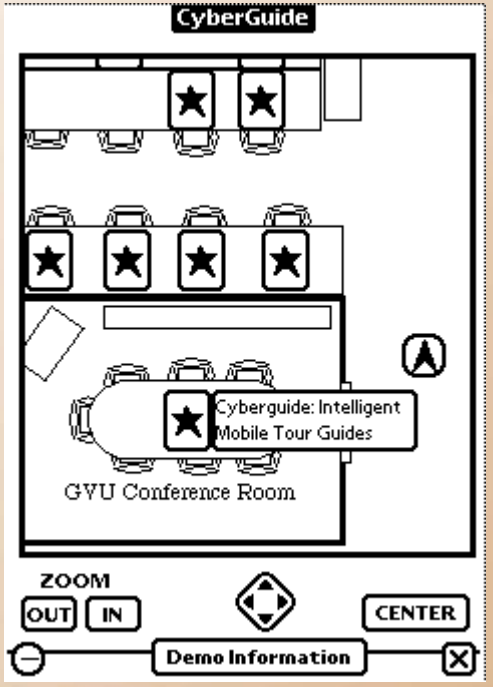
Results of Difficulties

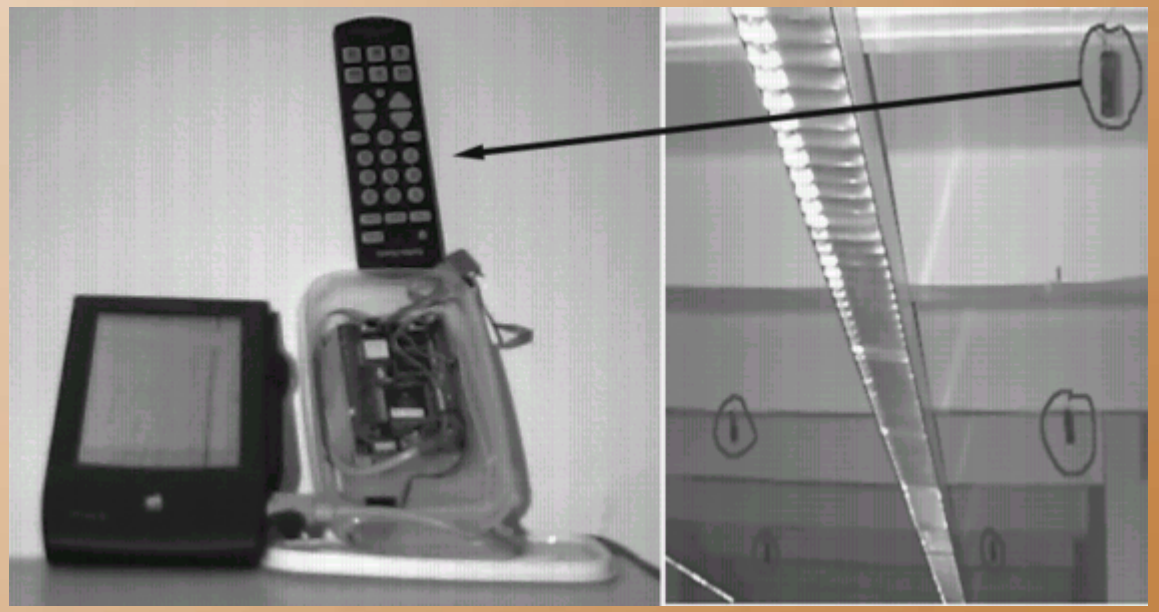
- *Ad hoc* application building
  - Difficult to build, reuse and evolve
- Small variety of sensors
- Small variety of context: mostly *location*
- Few applications, mostly simple: mostly *presenting context*

- Practical: difficult to prototype, test and evaluate

Why Applications are Hard to Build: A Case Study

- Cyberguide case study: no separation of concerns





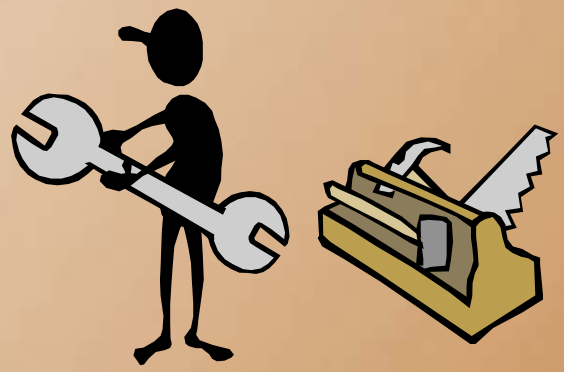
Outline

- Motivation
- Problems dealing with context
- Contribution: Context Toolkit
- Validation:
  - Design space and applications
  - Building more realistic applications
- Conclusions and future work

Need Programming Support

- Goal: make application development easier
- Identified number of requirements for architectural support and design process
- Examined existing support and determined how developers might think about building context-aware applications
- Developed Context Toolkit: architecture with supporting library of components

Related Work



- Existing Context-Aware Systems
  - Schilit (Columbia, 1995)
  - Stick-e notes (Pascoe, Kent, 1996)
  - CyberDesk (Dey, Georgia Tech, 1997)
  - CALAIS (Ward, Cambridge, 1998)
  - MUSE (Castro, UCLA, 2000)
  - Context-Awareness SDK (Tangis Corp., 2000)

- Proposed/Related Systems
  - Situated Computing Service (HP, 1997)
  - Contextual Information Service (Pascoe,Kent, 1998)
  - HIVE (Minar, MIT, 1998)
  - OAA (Cohen, OGI, 1996)

Research Contributions

- Conceptual framework requirements
  - Provide framework for designing apps more easily
  - Lower threshold to enable more designers
- Context Toolkit
  - Implementation and exploration of design space
- Support investigation of complex problems and more realistic apps
  - Raise ceiling
  - Privacy, uncertainty, security, end-user programming

Toolkit Requirements

- Context specification
- Discovery
- Separation of concerns
- Storage
- Constant availability
- Transparent communications
- Interpretation

Design Process

1. Specification
2. Acquisition
3. Delivery 
4. Reception
5. Action

Design Process

1. Specification
2. Acquisition
3. Delivery
4. Reception
5. Action

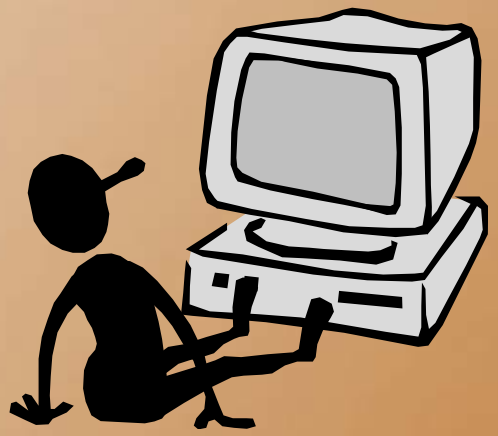
1. Specification
2. *Acquisition*
3. Action

Time for a Big Insight!

- Have a design process – complex and simplified
- Have a set of architecture requirements
- Need to figure out how to support these

Look to input handling

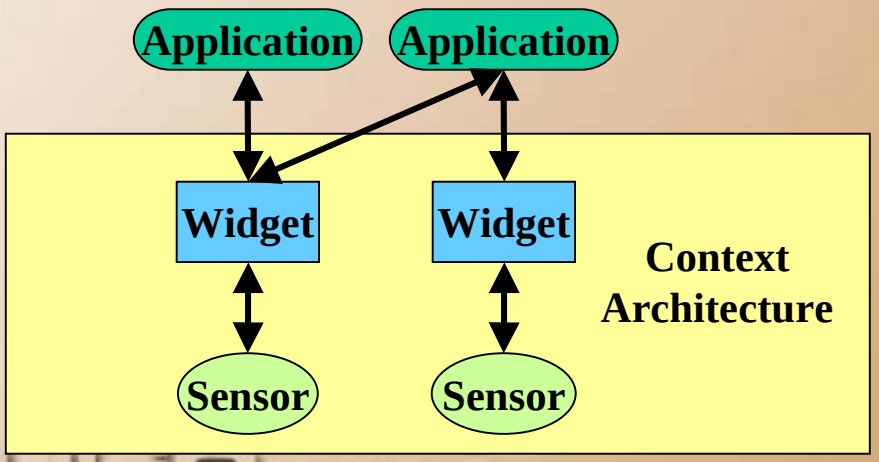
- Graphical User Interface (GUI) widgets
  - separation of concerns
  - callbacks and attributes
  - query/subscribe
  - common interface



  - e.g. button

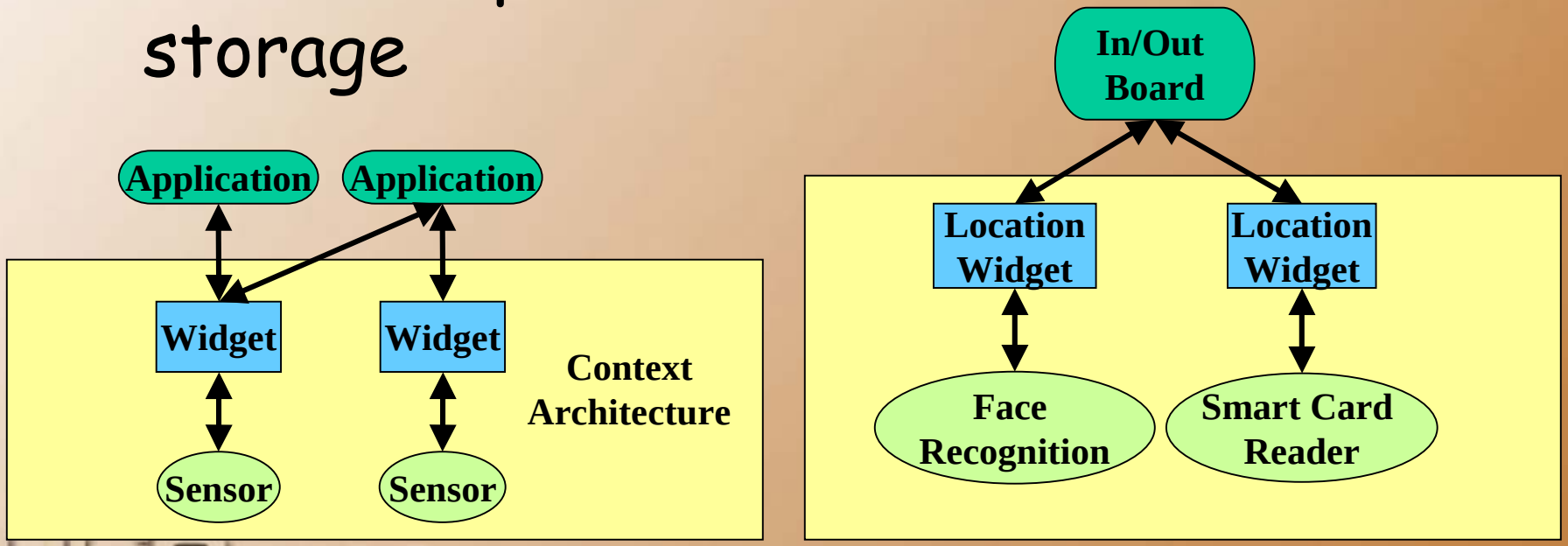
Context Widgets

- Responsible for acquiring and abstracting data from particular sensor, separation of concerns, storage



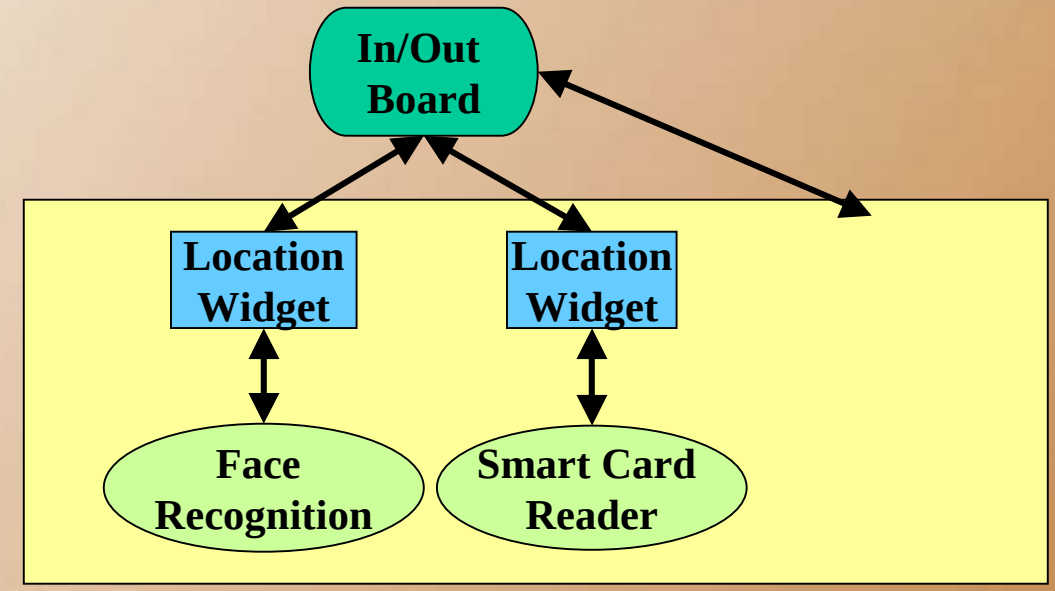
Context Widgets

- Responsible for acquiring and abstracting data from particular sensor, separation of concerns, storage



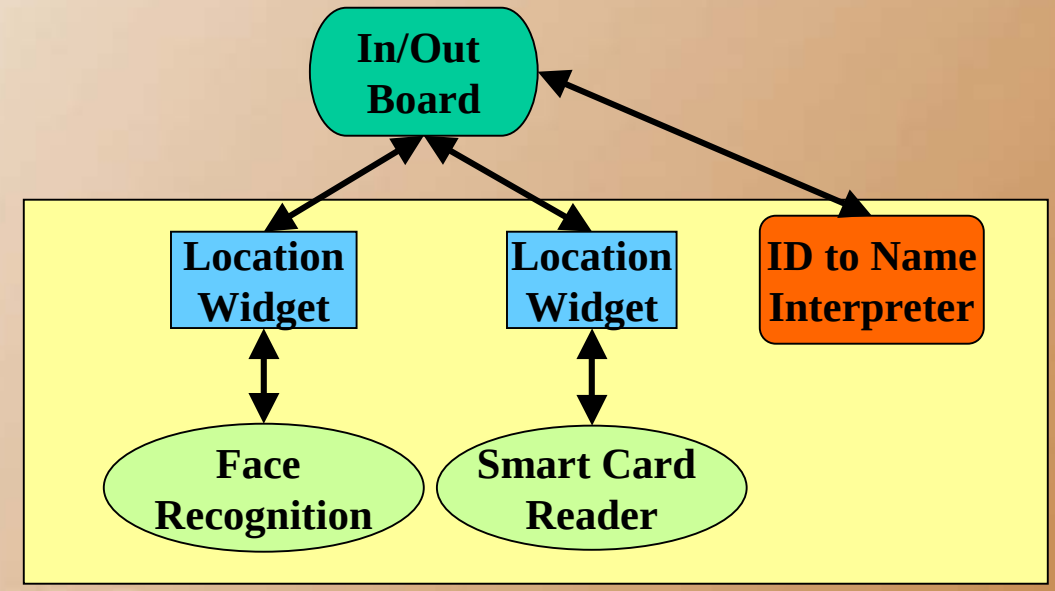
Context Interpreters

- Convert or interpret context to higher level information
- Context not available at appropriate level



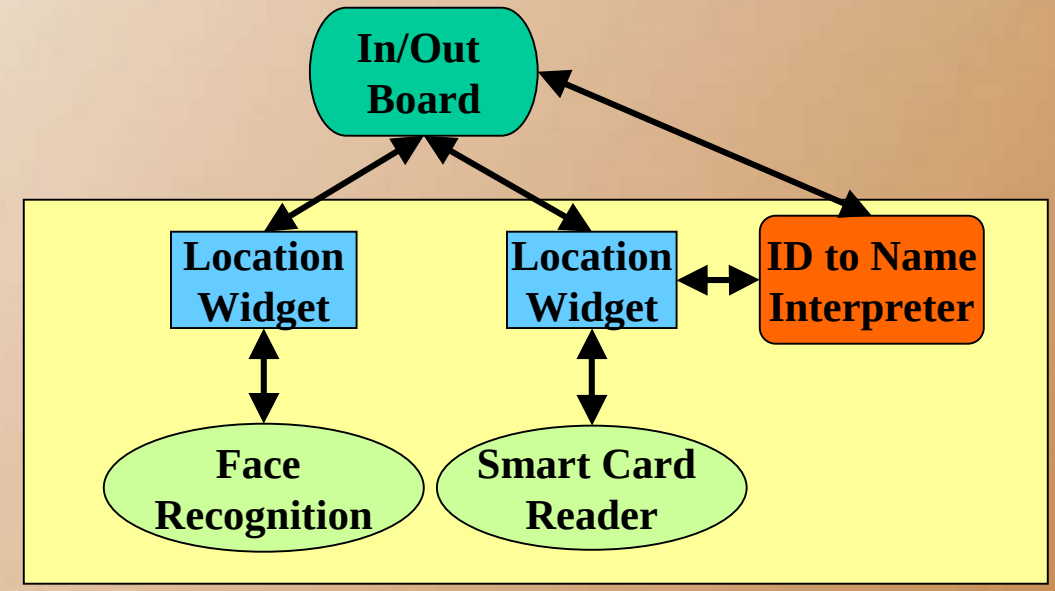
Context Interpreters

- Convert or interpret context to higher level information
- Context not available at appropriate level



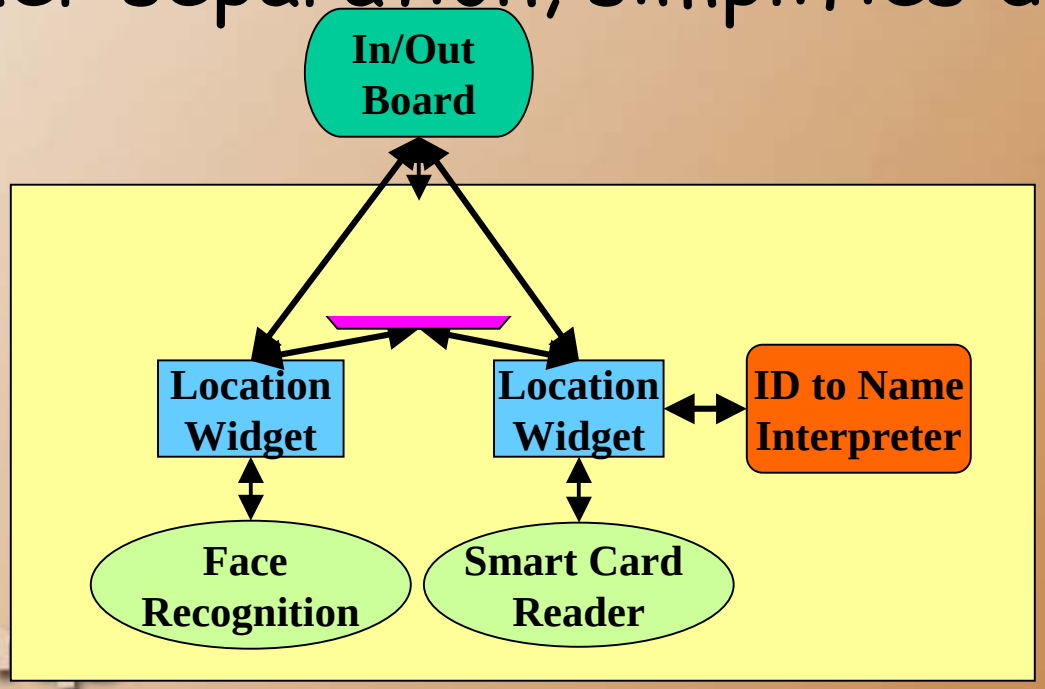
Context Interpreters

- Convert or interpret context to higher level information
- Context not available at appropriate level



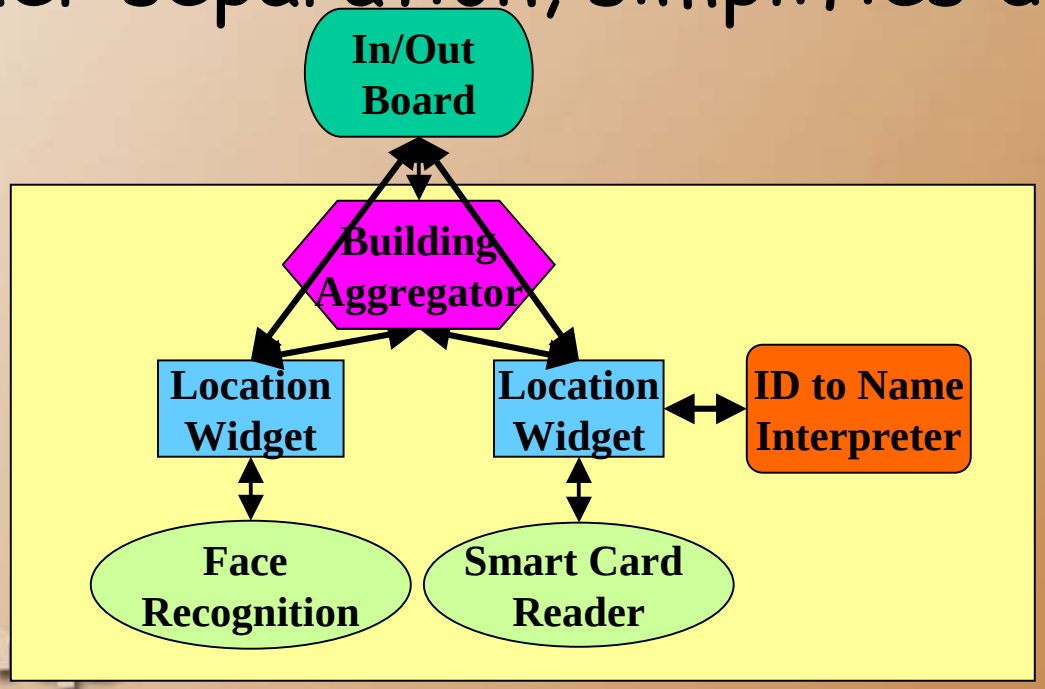
Context Aggregators

- Collect context relevant to particular entities (recall definition)
- Further separation, simplifies design



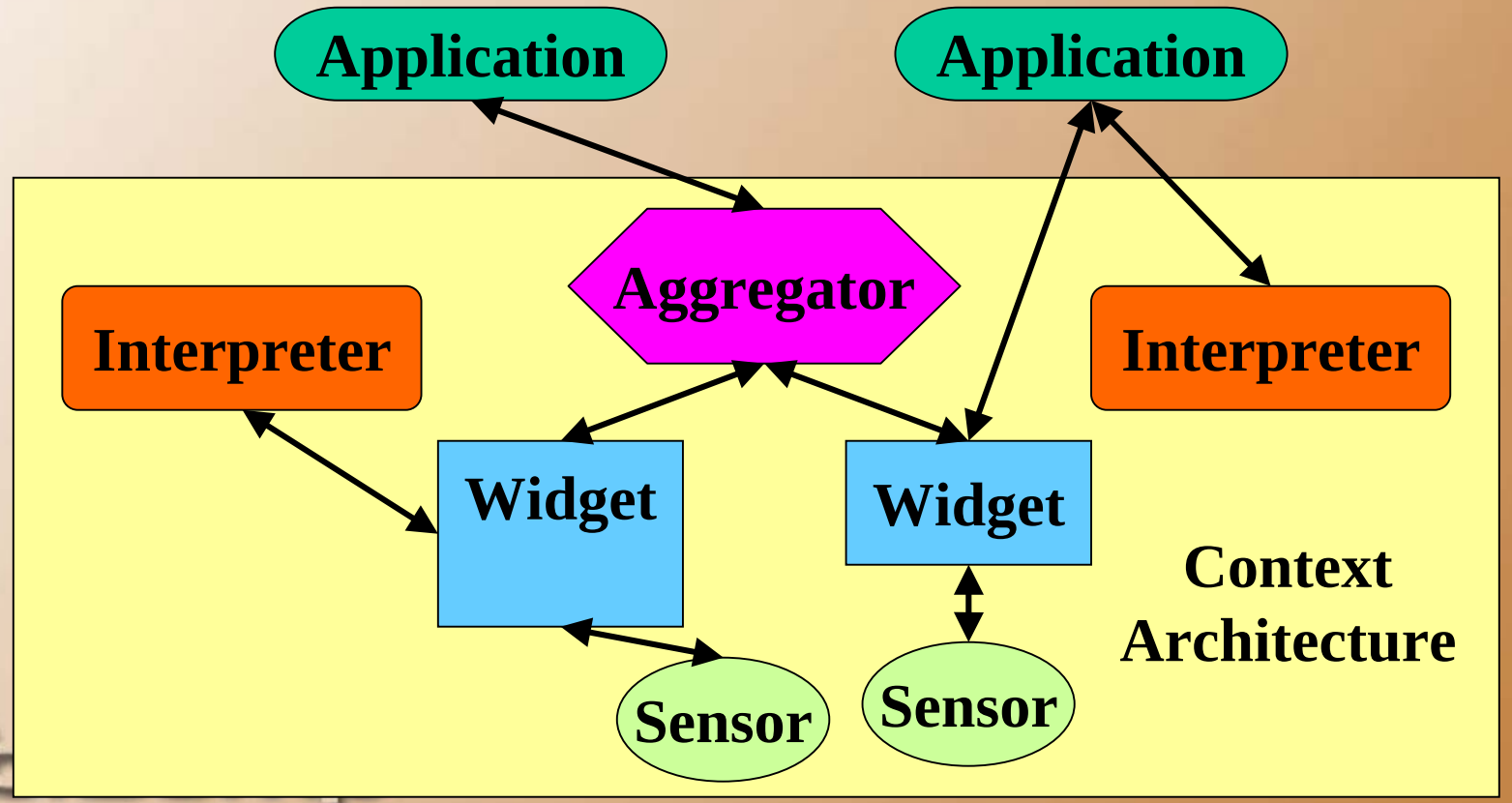
Context Aggregators

- Collect context relevant to particular entities (recall definition)
- Further separation, simplifies design



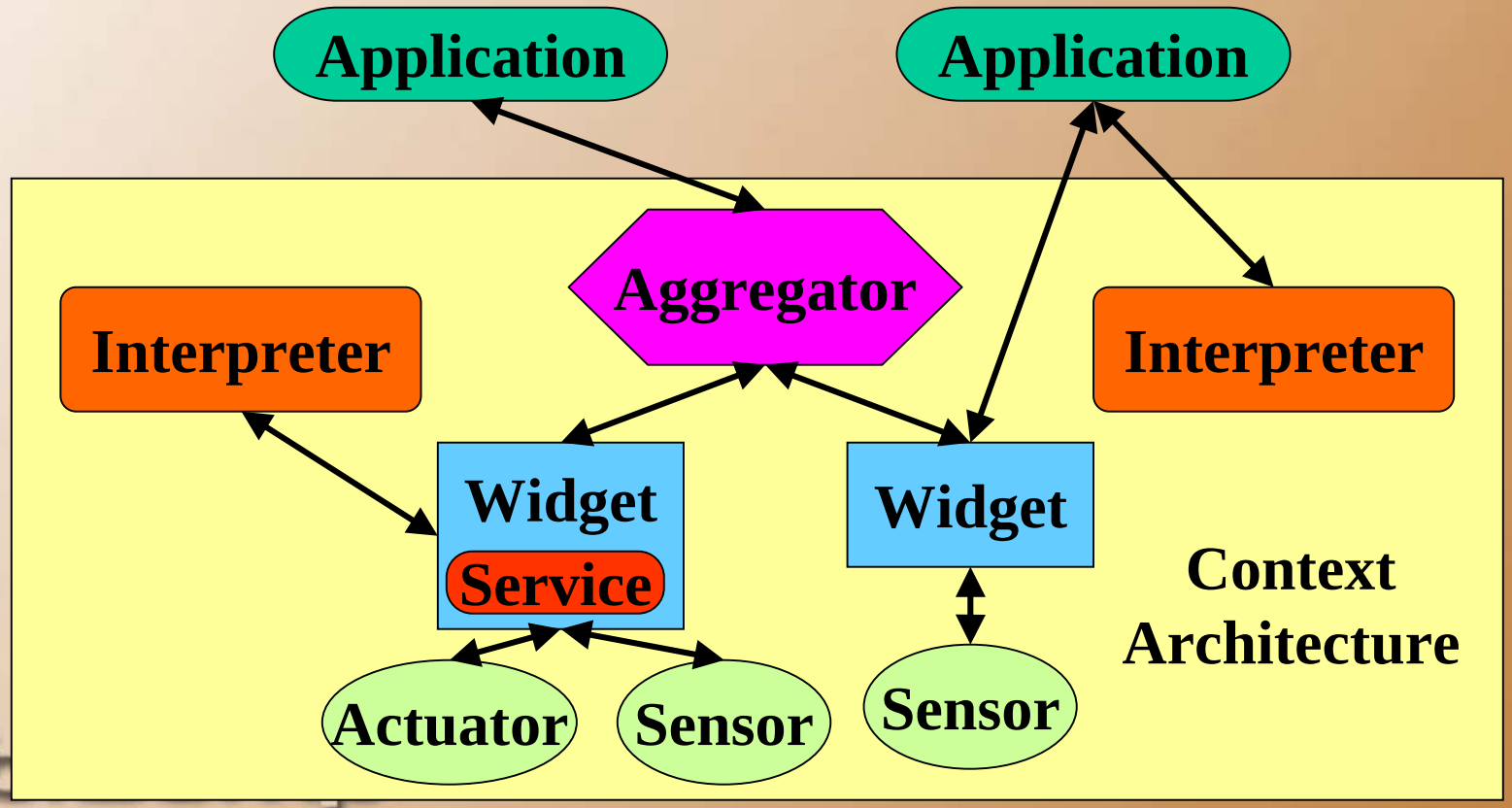
Context Toolkit Framework

- Supports real-world model/methodology and provides library (distributed: XML/HTTP, input-focused)
- Component model: facilitates building of applications



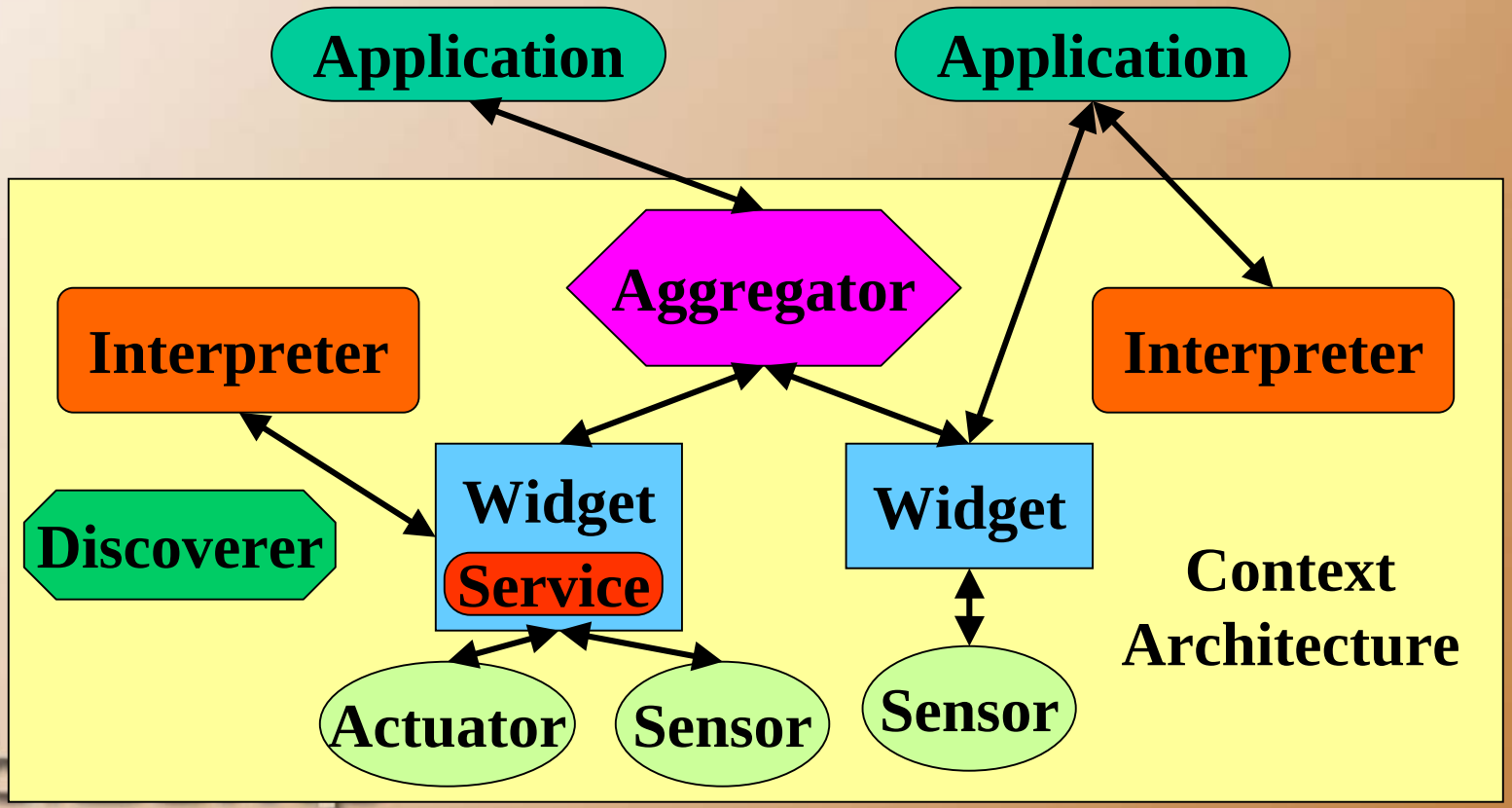
Context Toolkit Framework

- Supports real-world model/methodology and provides library (distributed: XML/HTTP, input-focused)
- Component model: facilitates building of applications



Context Toolkit Framework

- Supports real-world model/methodology and provides library (distributed: XML/HTTP, input-focused)
- Component model: facilitates building of applications



Experiences: Benefits

- Provides separation of concerns
- Lightweight integration and re-use of components
- Easy to create and evolve apps, allowing exploration of the design space
  - Add context to context-less apps
  - Add more context to context-aware apps

Outline

- Motivation
- Problems dealing with context
- Contribution: Context Toolkit
- Validation:
  - Design space and applications
  - Building more realistic applications
- Conclusions and future work

Validation

- Used to build existing applications
- Used to explore the design space
- Used to build more complex and realistic applications

Additional Validation

- Facilitating larger community outside of Georgia Tech, including:

- Arch:
  - CMU (mobile agents)
  - Motorola (arch/mobile user apps)
  - Autonama de Madrid (arch/smart spaces apps)

- Apps
  - Keele (desktop apps)
  - Novator (apps for mobile workers)
  - Technical University Munich/CMU (informal meeting support)

- British Telecom Labs
- MIT
- Trinity College
- PLAY Research Group

- Stuttgart
- SICS, Sweden
- ETH
- Philips
- Telenor
- Nokia

Aware Home (MANSE '99)

- Great testbed for context-aware computing
- 3 goals: elderly, infants, everyone
- Context Toolkit is the s/w infrastructure in the Aware Home

Aware Home (MANSE '99)

- Gr
  co
- 3 g
- Co
  inf

Design Space for Context-Aware Applications

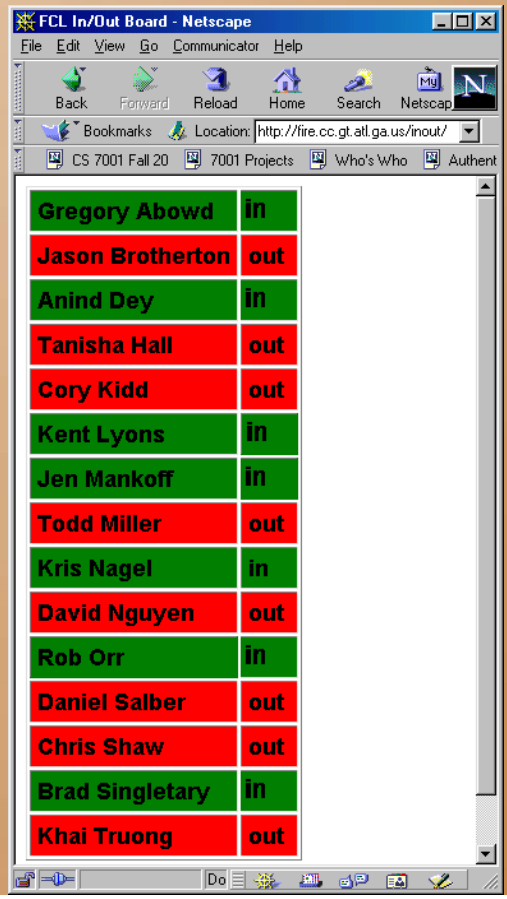
- Toolkit allows exploration of design space
- Types of context:
  - Location, identity, time, activity
  - Simple/singular → complex/multiple
  - Combinations
- Uses of context:
  - **Present** to user
  - Automatically perform set of **services**
  - **Tag** captured information to ease retrieval

Applications Built

- Simple use of location:
  - Turn lights on and off (perform service)
- Location and id (perform service)
  - Information Guide: present info about user's group (CHI '99)
  - Context-Aware Mailing List

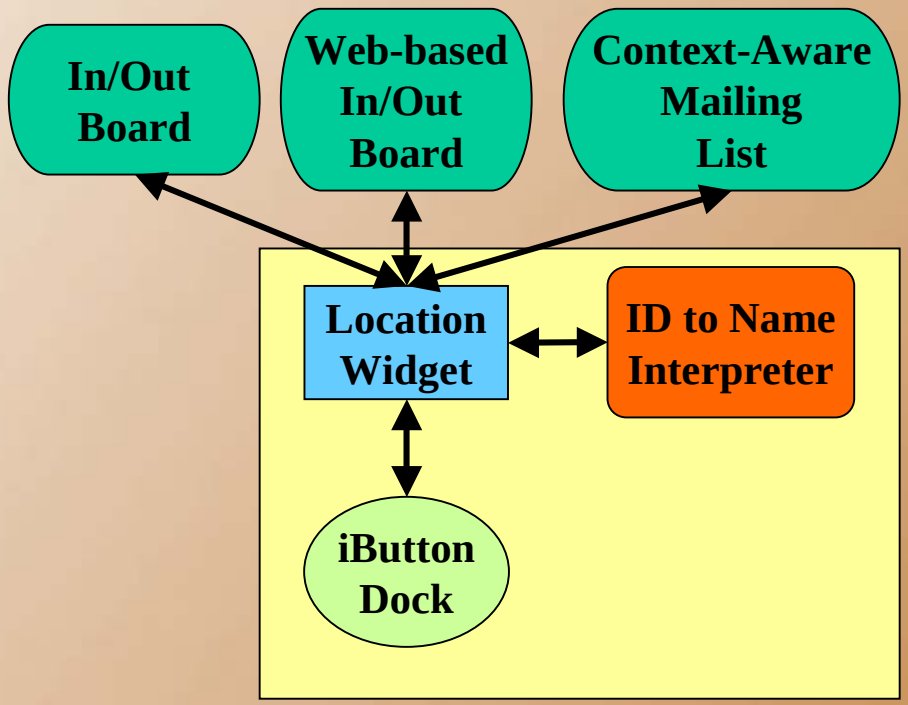
In/Out Board – 3 versions
(CHI '99)

- Context used: location, identity, time
- How used: present context



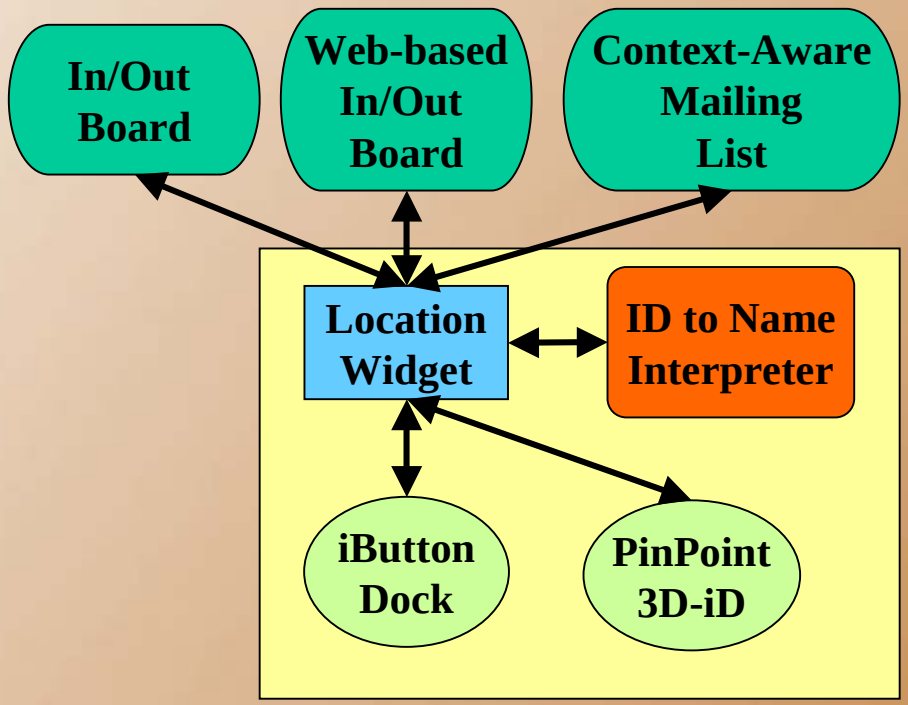
In/Out Board Architecture

- Simple apps demonstrates support for reusability (don't have to re-build infrastructure on per-application basis) and evolving applications



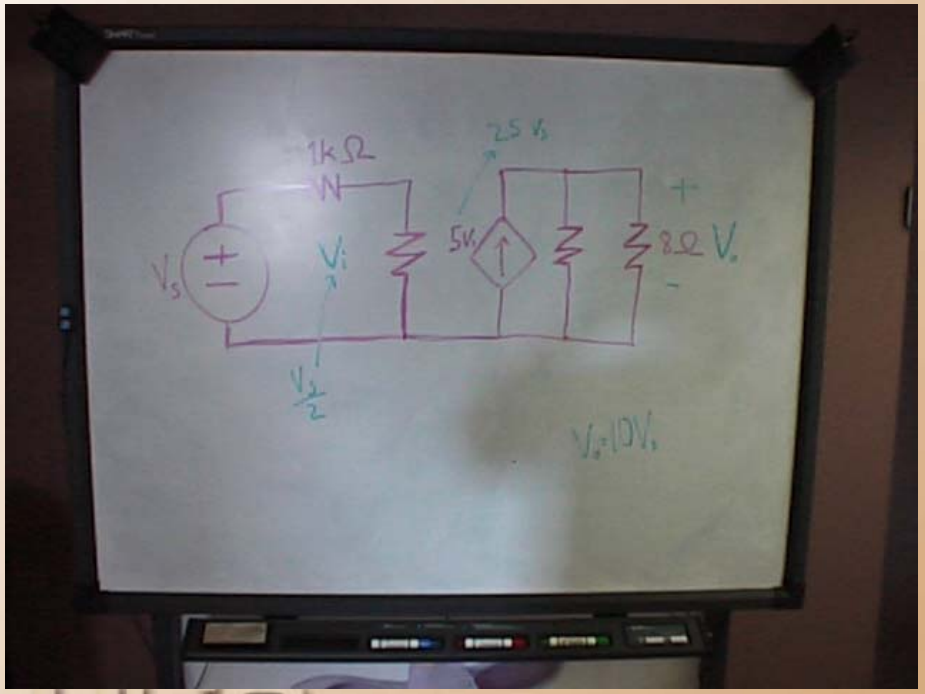
In/Out Board Architecture

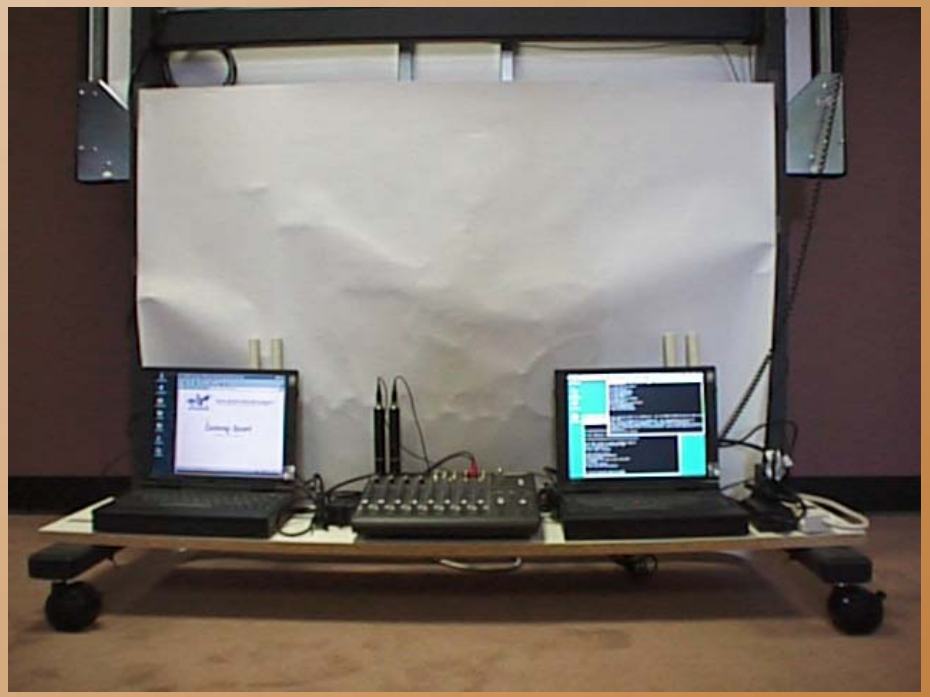
- Simple apps demonstrates support for reusability (don't have to re-build infrastructure on per-application basis) and evolving applications



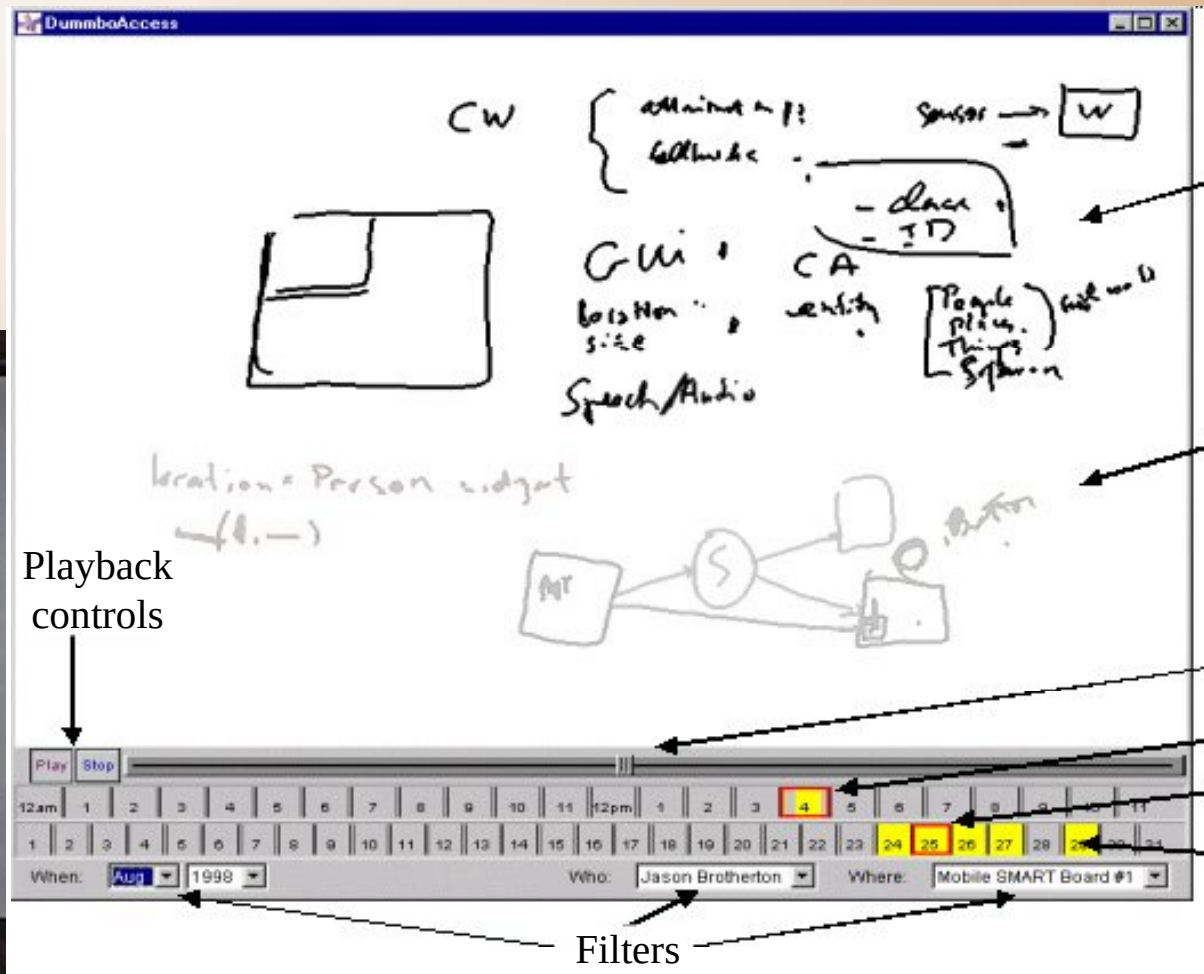
Serendipitous Meetings

- Context used: location, id, time, activity
- How used: present, perform service, tag
- Work done by others in research group



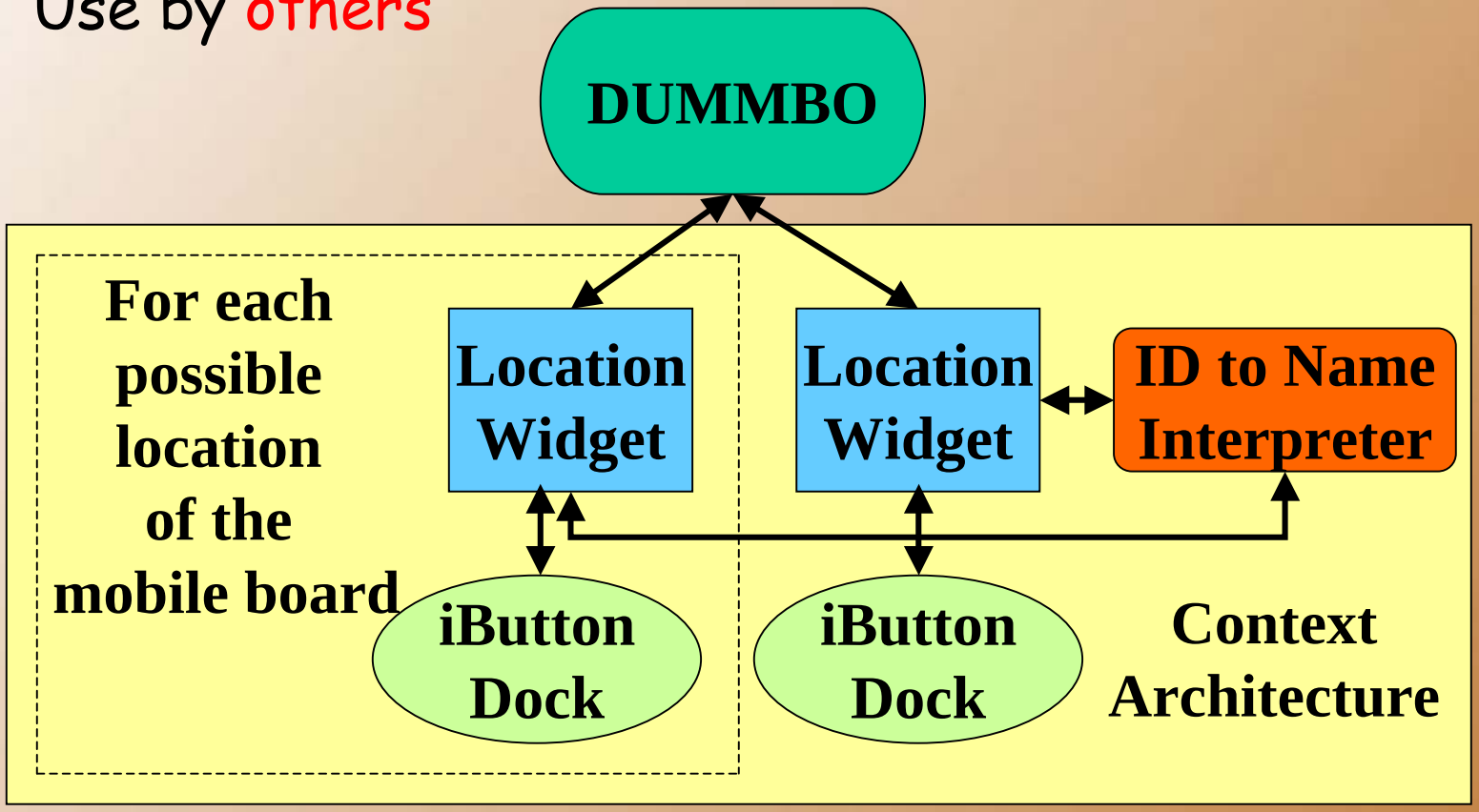


Serendipitous Meetings



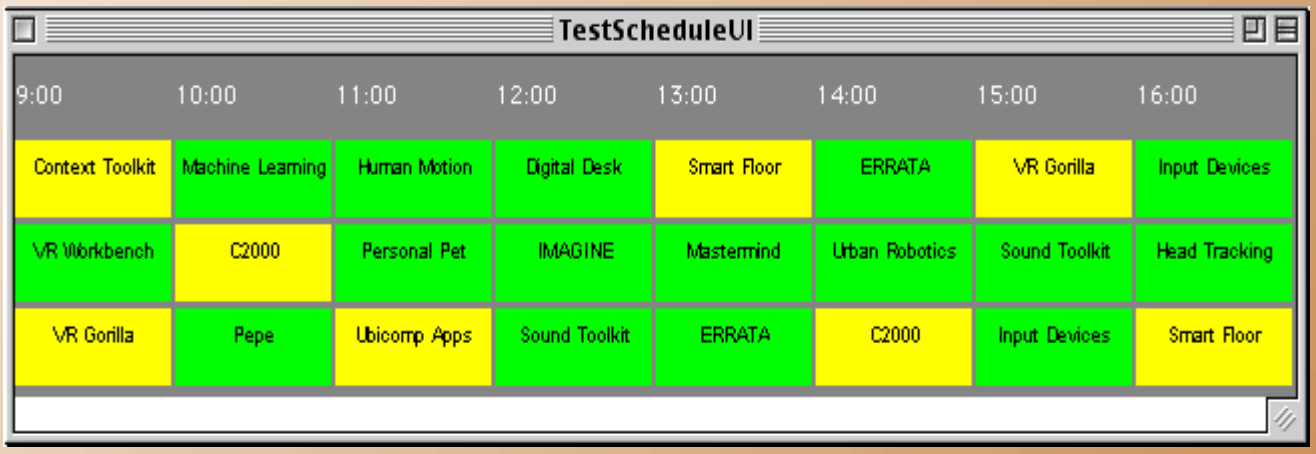
Meeting Architecture

- Demonstrates support for evolution
- Use by others



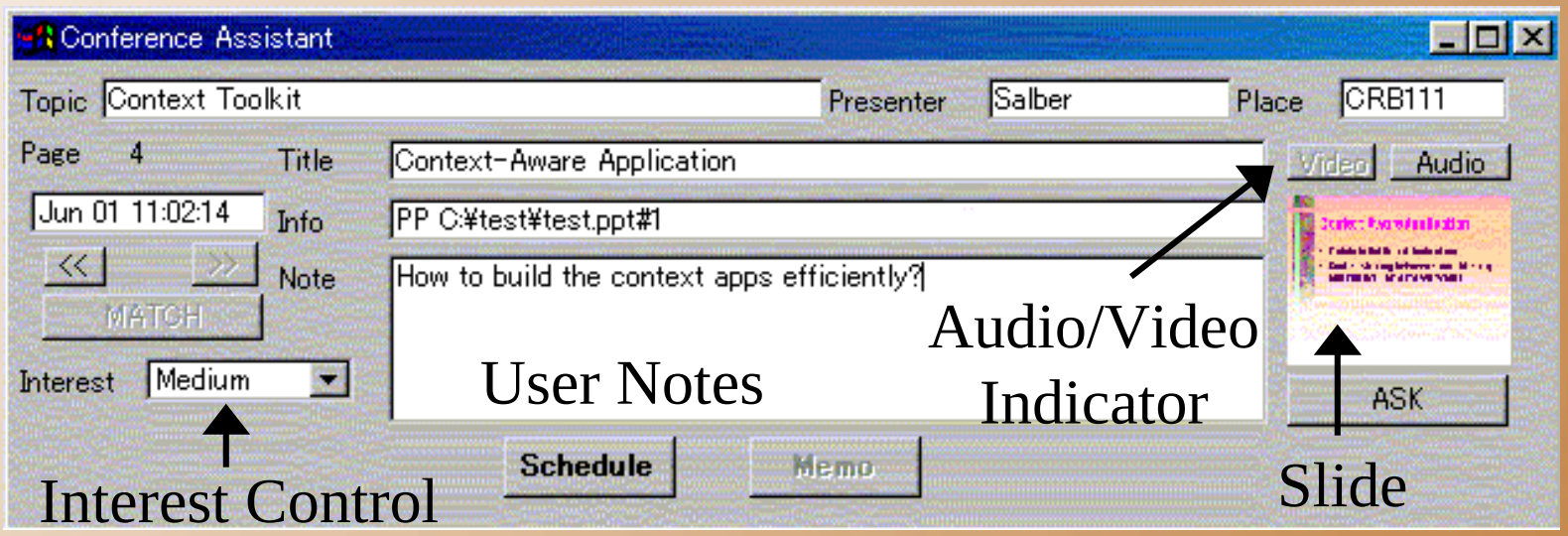
Conference Assistant (ISWC '99)

- Context used: location, multiple levels of identity, activity, time
- How used: present, service, tag



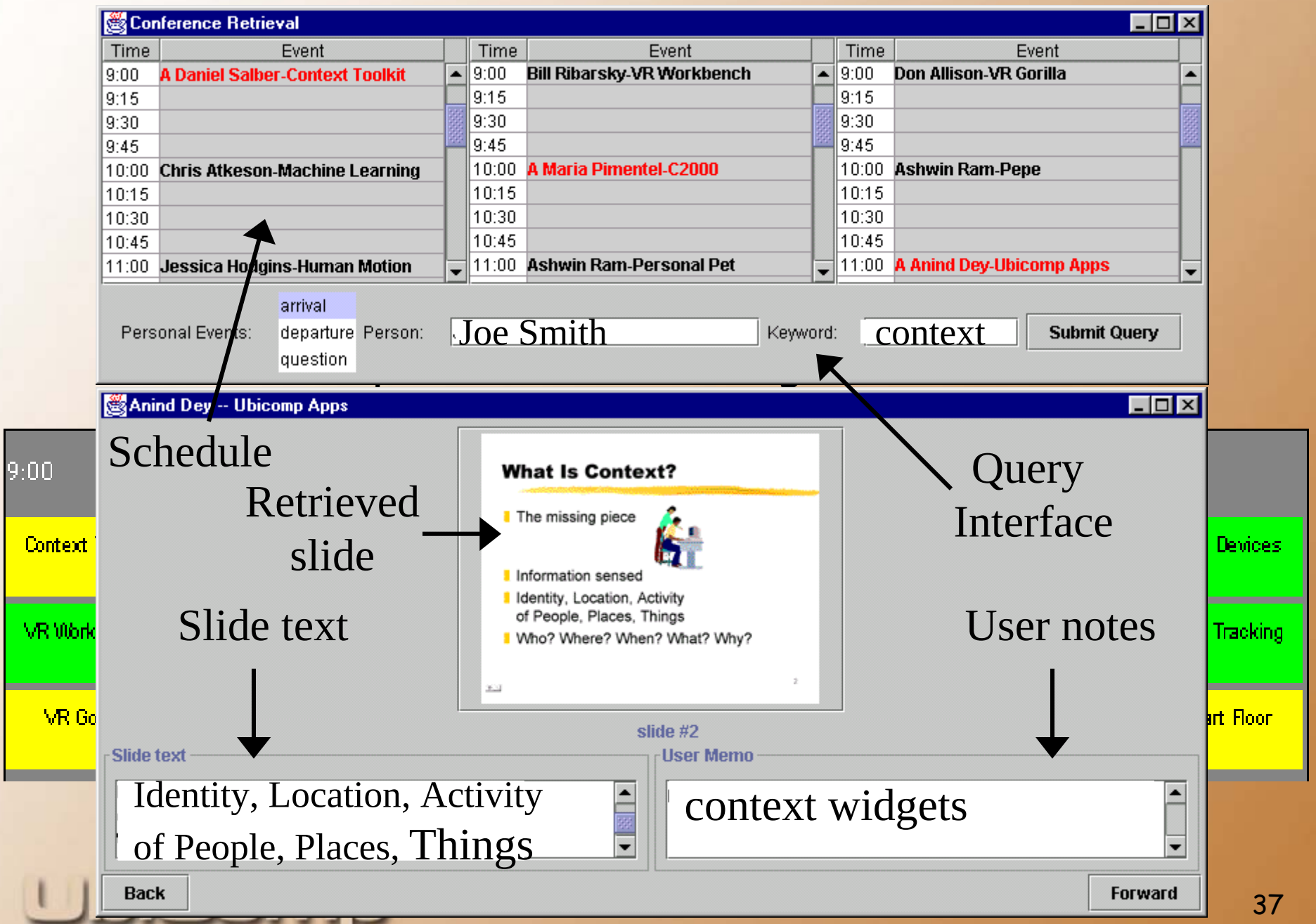
Conference Assistant (ISWC '99)

- Context used: location, multiple levels of identity, activity, time
- How used: present, service, tag



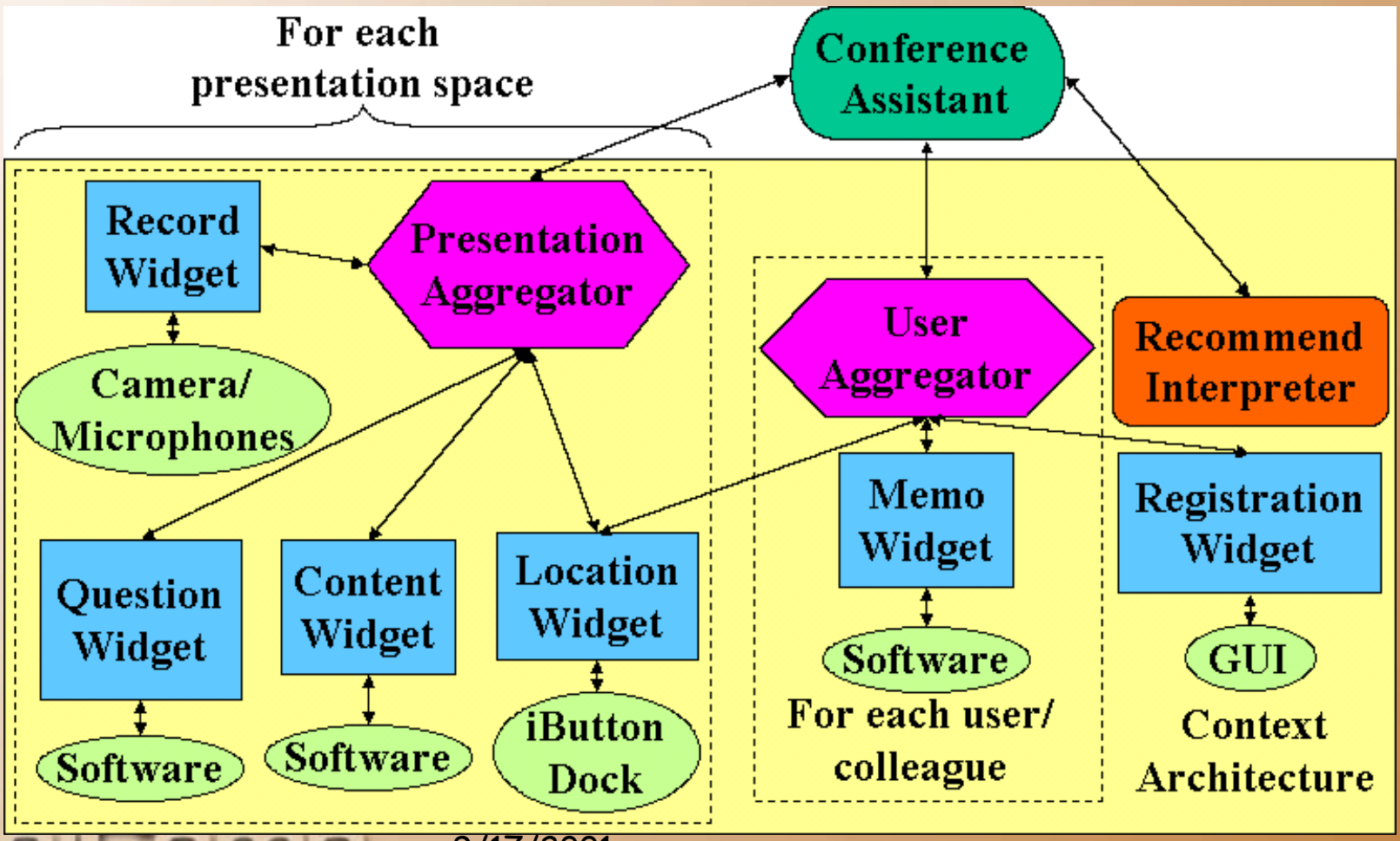
Conference Assistant (ISWC '99)

- Context used: location, multiple levels of identity, activity, time
- How used: present, service, tag


Conference Retrieval
Time
Event
9:00 A Daniel Salber-Context Toolkit
9:15
9:30
9:45
10:00 Chris Atkeson-Machine Learning
10:15
10:30
10:45
11:00 Jessica Hodgins-Human Motion
9:00 Bill Ribarsky-VR Workbench
9:15
9:30
9:45
10:00 A Maria Pimentel-C2000
10:15
10:30
10:45
11:00 Ashwin Ram-Personal Pet
9:00 Don Allison-VR Gorilla
9:15
9:30
9:45
10:00 Ashwin Ram-Pepe
10:15
10:30
10:45
11:00 A Anind Dey-Ubicomp Apps
Personal Events:
arrival
departure
question
Person:
Joe Smith
Keyword:
context
Submit Query
Anind Dey -- Ubicomp Apps
Schedule
Retrieved slide
Slide text
Query Interface
User notes
What Is Context?
The missing piece
Information sensed
Identity, Location, Activity of People, Places, Things
Who? Where? When? What? Why?
slide #2
Slide text
Identity, Location, Activity of People, Places, Things
User Memo
context widgets
Back
Forward
9:00
Context
VR Work
VR Go
Devices
Tracking
art Floor

Conference Assistant Arch.

- Complex application: reuse, evolution



Outline

- Motivation
- Problems dealing with context
- Contribution: Context Toolkit
- Validation:
  - Design space and applications
  - Building more realistic applications
- Conclusions and future work

Complex and Realistic Applications

- Privacy: Dynamic Door Displays
- Ambiguity: In/Out Board extension
- Security: Service/Context Access (SACMAT 2001)
- End-user programming: CybreMinder (HUC 2000)

Component Abstraction

Features	Abstractions	
	None	Component
Context acquisition	X	P
Distributed communications	X	V
Query/subscribe	X	V
Storage	X	V
Multiple Apps	X	V
Sensor addition	X	P
Sensor failure	X	P
Evolution	X	P
Context specification	X	P
Situation realization	X	X

Level of Support
X – none
P – partial
V – complete

Component Abstraction

Features	Abstractions	
	None	Component
Context acquisition	X	P
Distributed communications	X	V
Query/subscribe	X	V
Storage	X	V
Multiple Apps	X	V
Sensor addition	X	P
Sensor failure	X	P
Evolution	X	P
Context specification	X	P
Situation realization	X	X

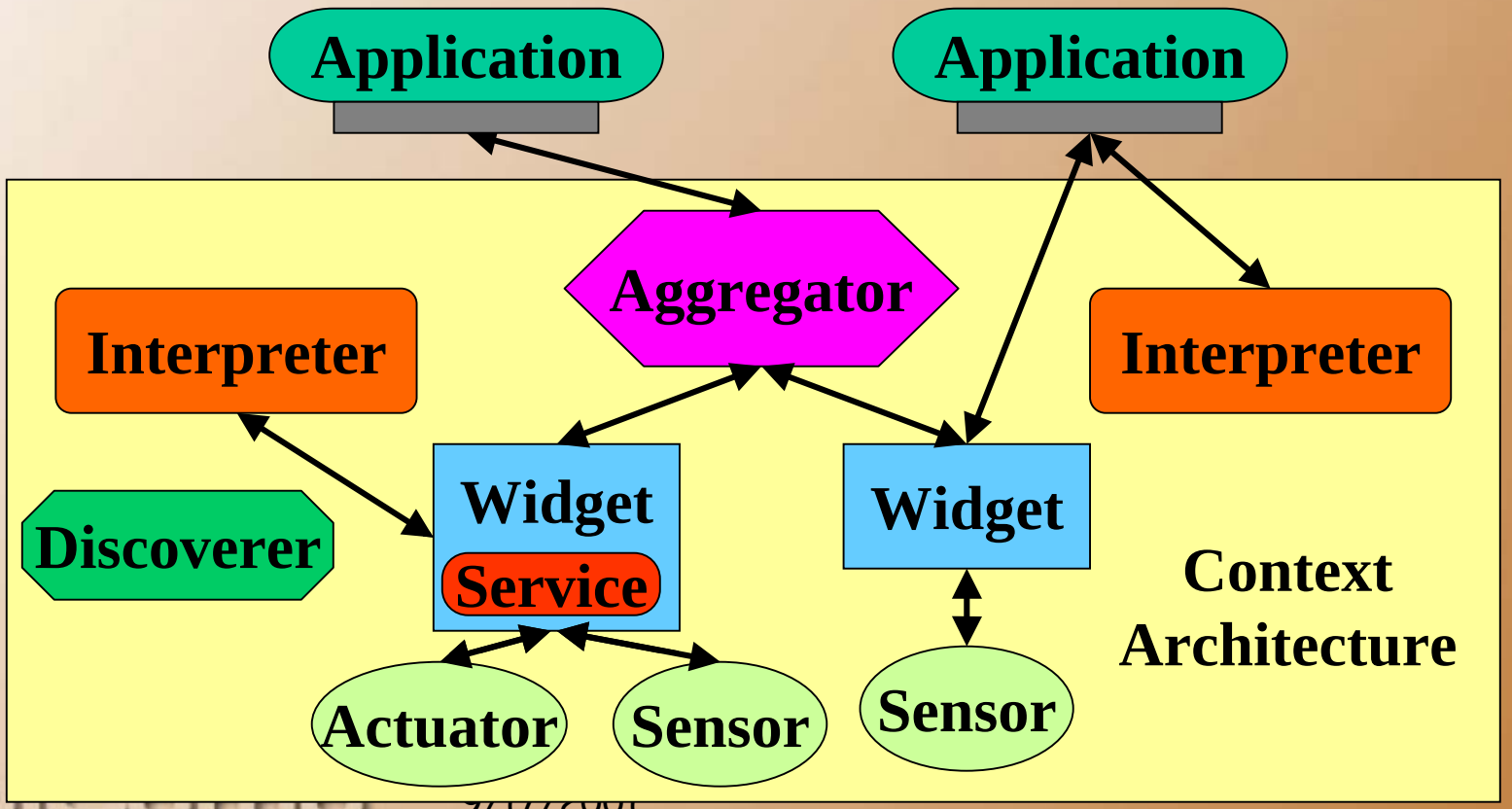
Level of Support
X – none
P – partial
V – complete

Situation Abstraction: Declarative Style

- Revisit context definition
  - Allow programmer to define a situation (real-world callbacks)
- *Declare* what context you want, not how to obtain it
- Architecture's responsibility to deliver it
- Makes *specification* in design process simpler, more robust, easier to evolve

Revised Framework

- Supports blackboard/box model of the world
- Different than component model



Situation Abstraction

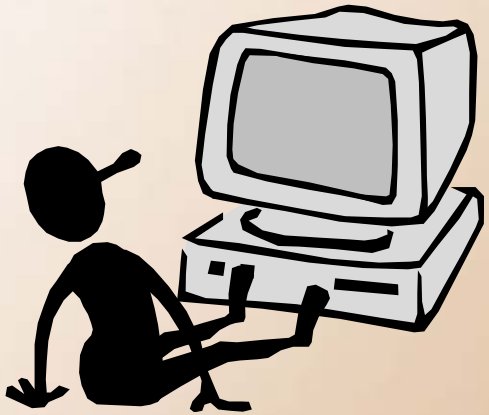
Features	Abstractions	
	None	Component
Context acquisition	X	P
Distributed communications	X	V
Query/subscribe	X	V
Storage	X	V
Multiple Apps	X	V
Sensor addition	X	P
Sensor failure	X	P
Evolution	X	P
Context specification	X	P
Situation realization	X	X

Situation Abstraction

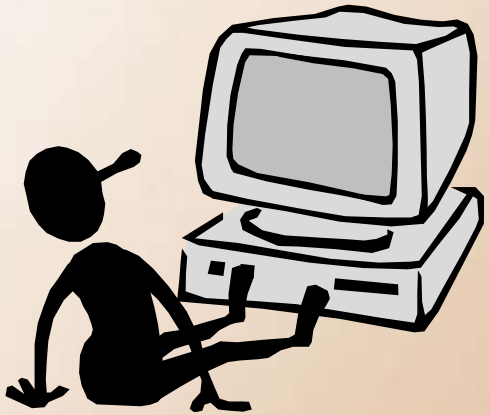
Features	Abstractions		
	None	Component	Situation
Context acquisition	X	P	P
Distributed communications	X	V	V
Query/subscribe	X	V	V
Storage	X	V	V
Multiple Apps	X	V	V
Sensor addition	X	P	V
Sensor failure	X	P	V
Evolution	X	P	V
Context specification	X	P	V
Situation realization	X	X	V

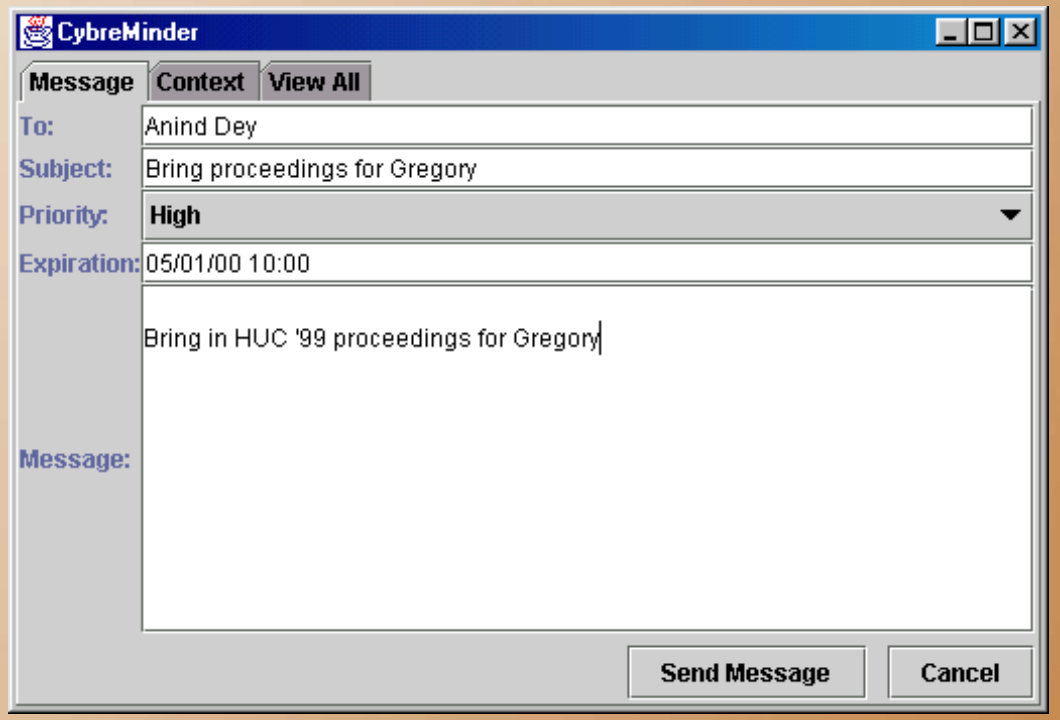
CybreMinder (HUC '00)

CybreMinder (HUC '00)



CybreMinder (HUC '00)



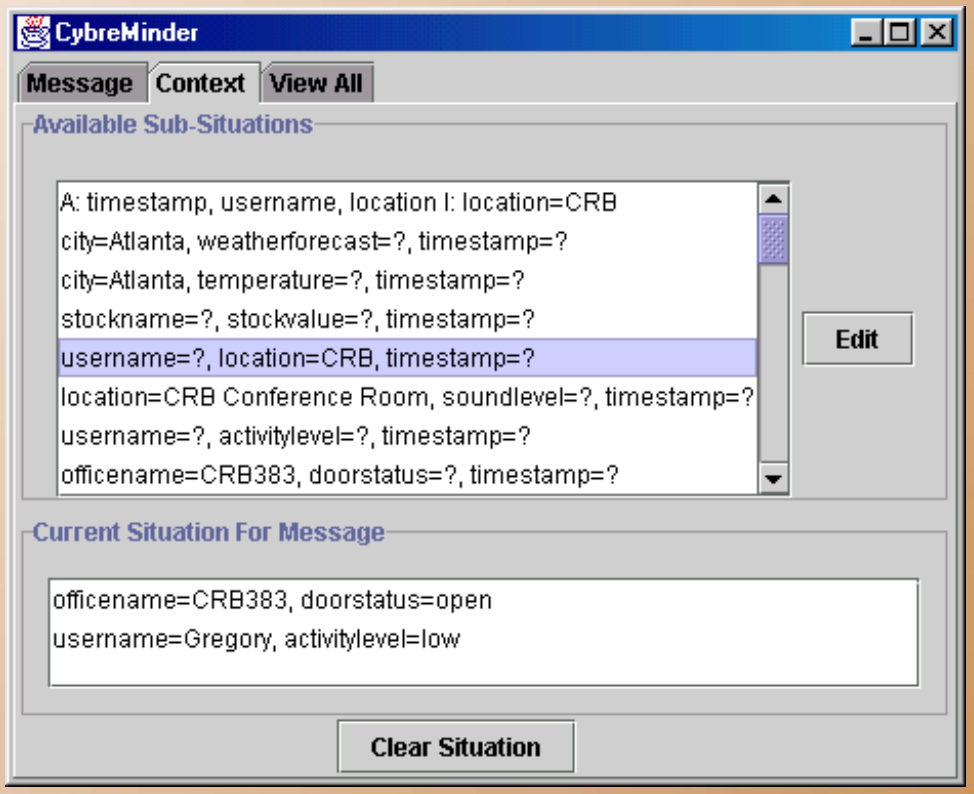


CybreMinder (HUC '00)

CybreMinder (HUC '00)

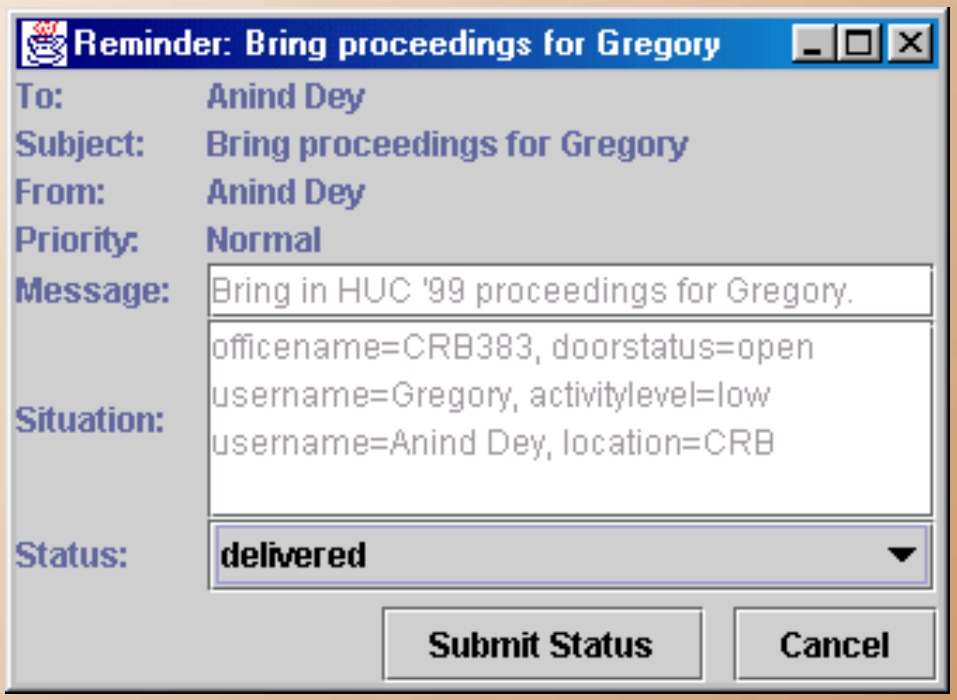
CybreMinder (HUC '00)

CybreMinder (HUC '00)

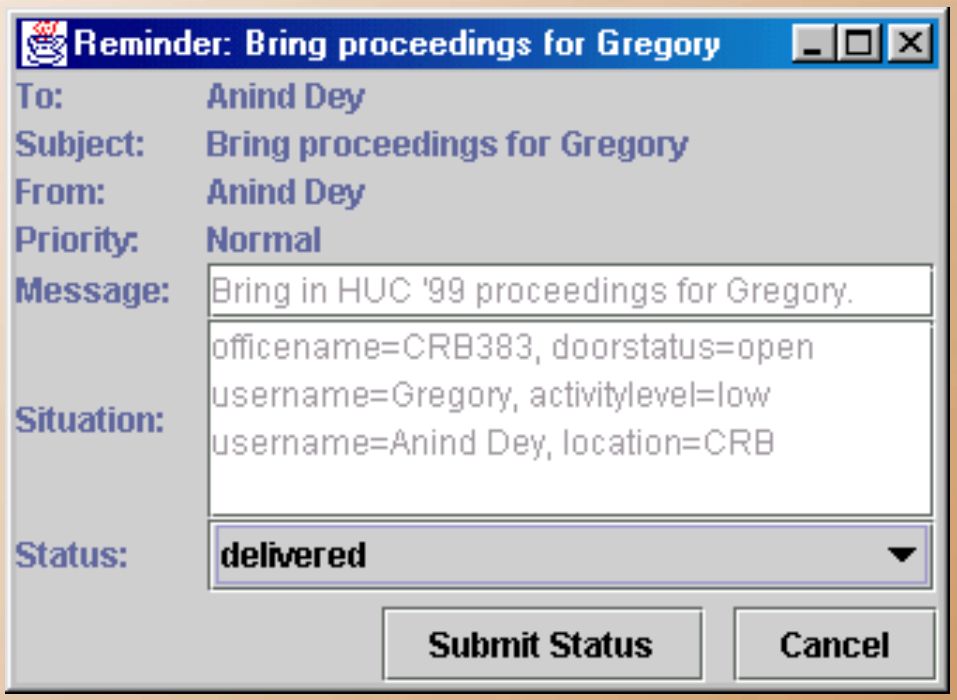


CybreMinder (HUC '00)

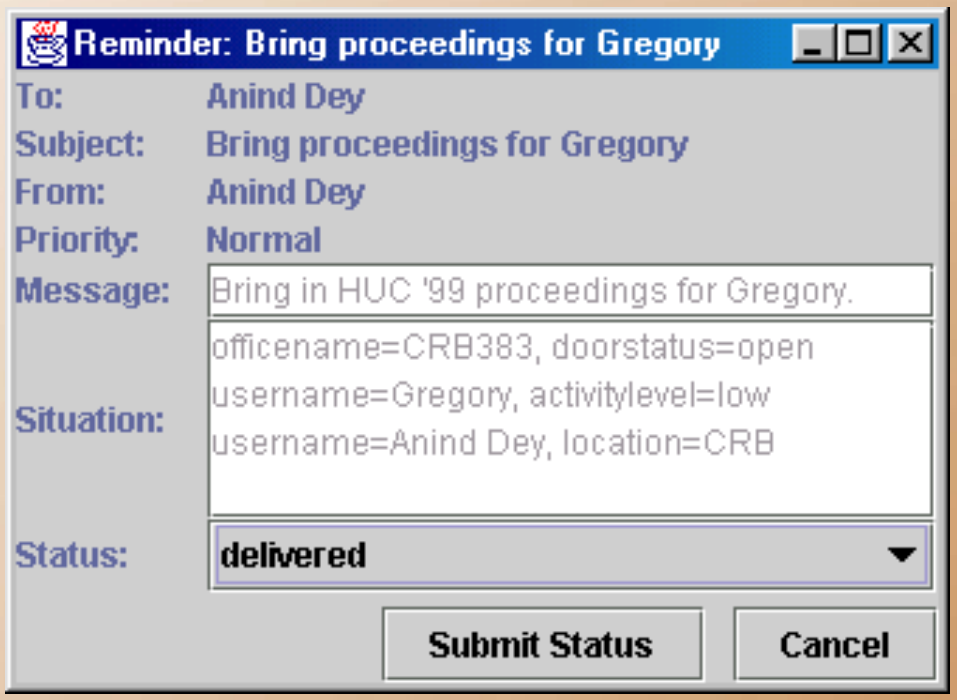
CybreMinder (HUC '00)



CybreMinder (HUC '00)



CybreMinder (HUC '00)



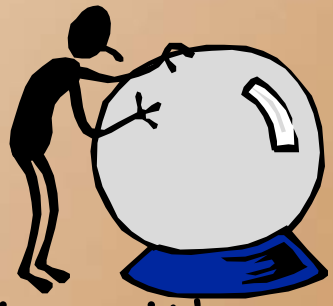
Outline

- Motivation
- Problems dealing with context
- Contribution: Context Toolkit
- Validation:
  - Design space and applications
  - Building more realistic applications
- Conclusions and future work

Research Contributions

- Conceptual framework requirements
  - Provide framework for designing apps more easily
  - Lower threshold to enable more designers
- Context Toolkit
  - Implementation and exploration of design space
- Support investigation of complex problems and more realistic apps
  - Raise ceiling
  - Privacy, uncertainty, security, end-user programming

What's Next?

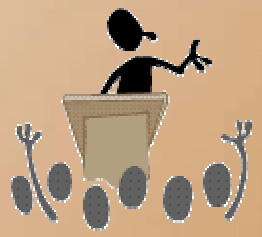


- Complex interpretation, sensor fusion, dealing with ambiguity → how to infer what a user really wants
- Ontology, QoS, privacy, security
- Data models
- Development environment
- Evaluation of context-awareness
- What to do, and when and why
- Overload/how interruptible is the user
- End-user control of what happens
- Broaden scope of framework to be a general model of interactive and ubiquitous computing: look at implicit output

Acknowledgements

- Gregory D. Abowd & FCE
- Motorola & NSF

- Contact info:
  - anind@cc.gatech.edu
  - http://www.cc.gatech.edu/~anind
  - http://www.cc.gatech.edu/fce/ctk
- Questions?



Intercom System

- Facilitate communications between family members:
  - In the same house
  - Between houses
  - While mobile
- Uses 4 types of context in combination (primarily present, service, but potential for tagging when learning)
- Leverage social mediation skills

Design Process

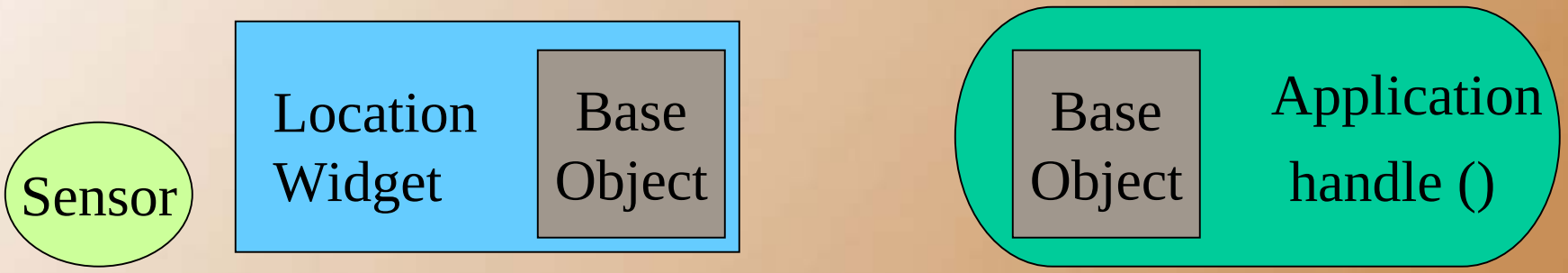
1. Specification – context and behaviors
2. Acquisition – install, API, query/notify, store, interpret
3. Delivery – deliver context to multiple, remote applications
4. Reception – locate relevant sensors, request context, interpret
5. Action – analysis and action

How to simplify?

- Brooks 87: "No Silver Bullet: Essence and Accidents of Software Engineering"
  - essential problems
    - inherent problems
    - specific to the task at hand
  - accidental problems
    - problems induced by design tools
    - not specific to the task

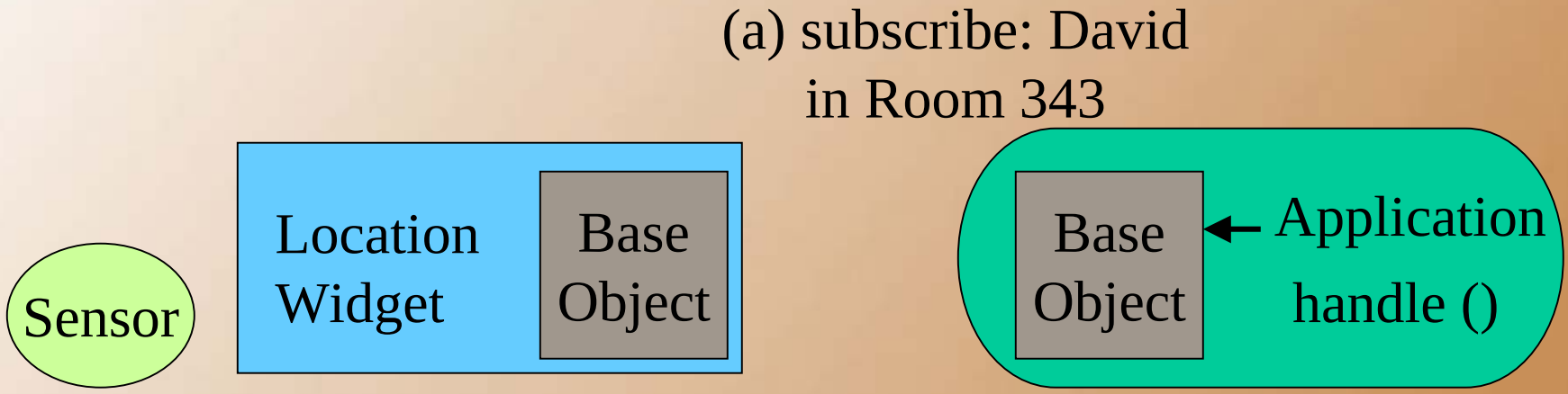
Callback Model

- Transparent communications, always available
- Similar to GUI architectures



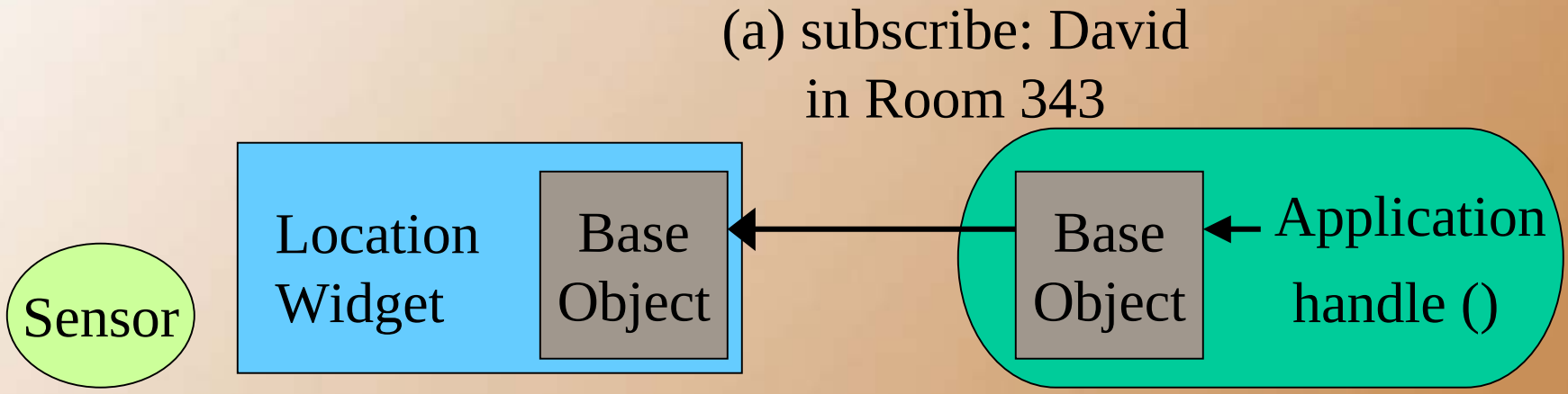
Callback Model

- Transparent communications, always available
- Similar to GUI architectures



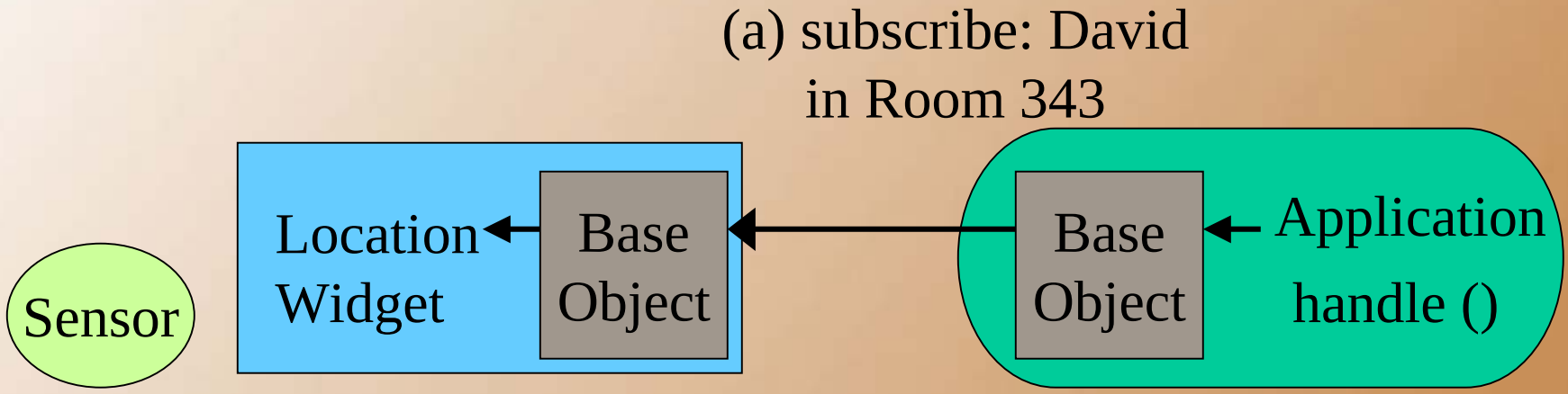
Callback Model

- Transparent communications, always available
- Similar to GUI architectures



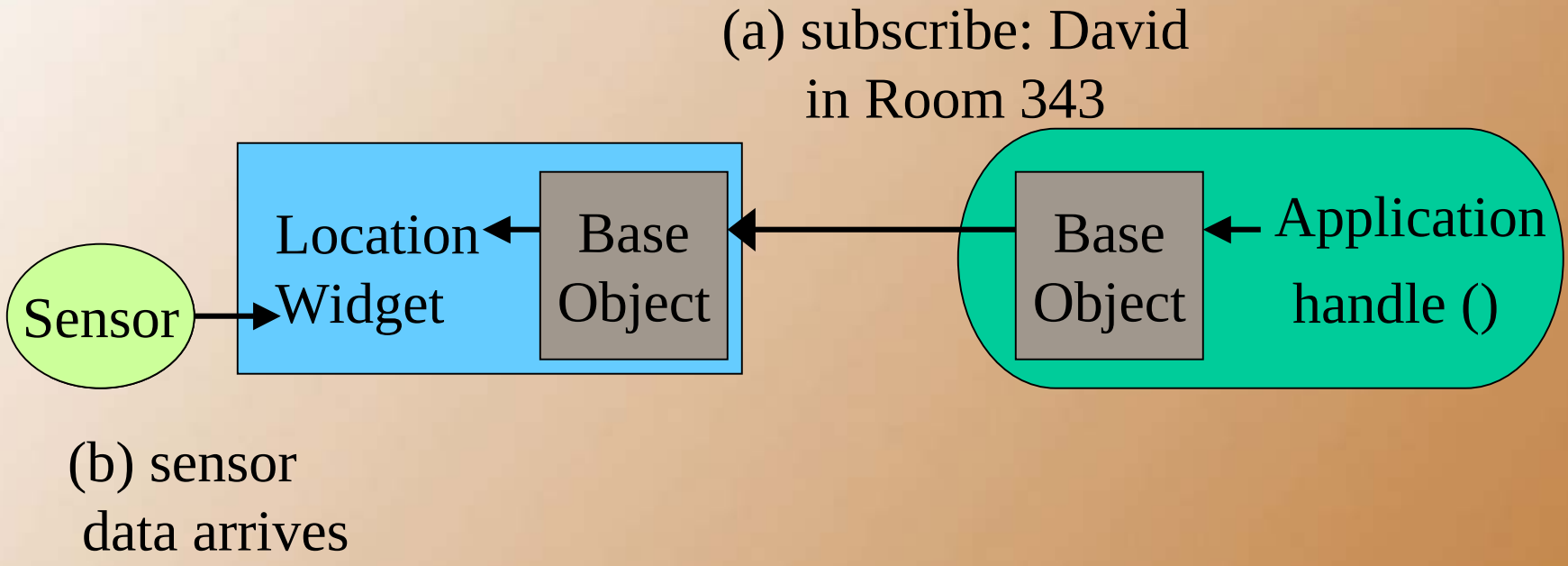
Callback Model

- Transparent communications, always available
- Similar to GUI architectures



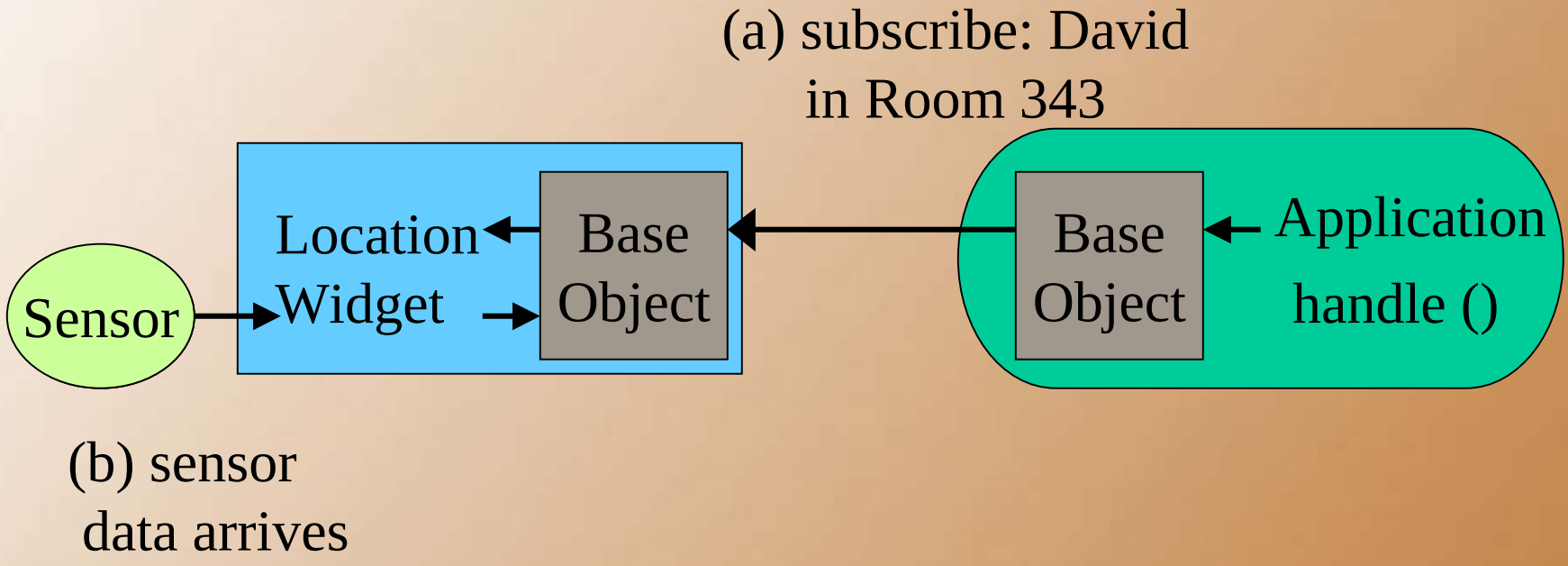
Callback Model

- Transparent communications, always available
- Similar to GUI architectures



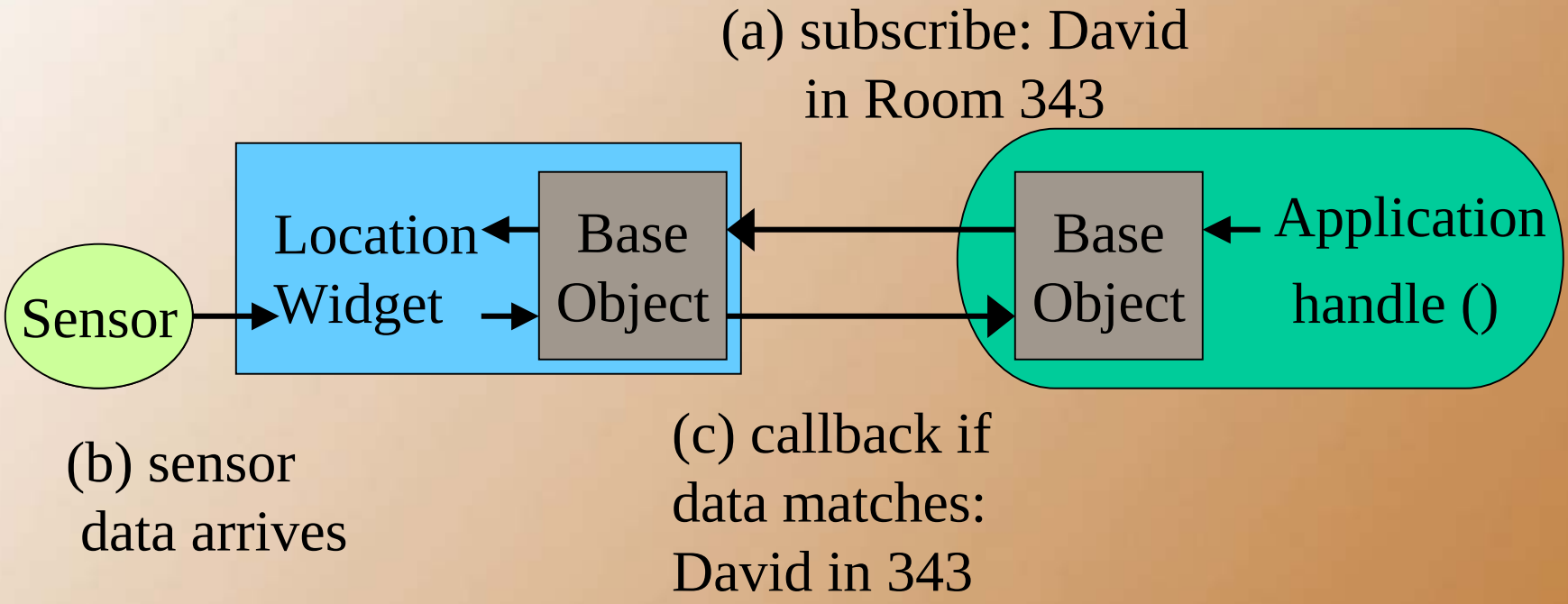
Callback Model

- Transparent communications, always available
- Similar to GUI architectures



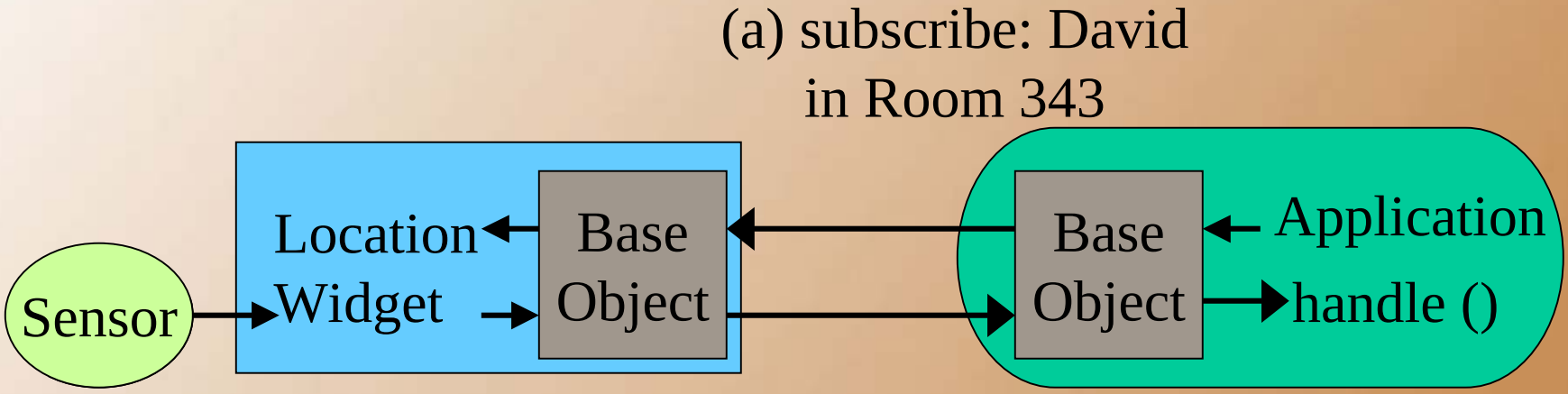
Callback Model

- Transparent communications, always available
- Similar to GUI architectures



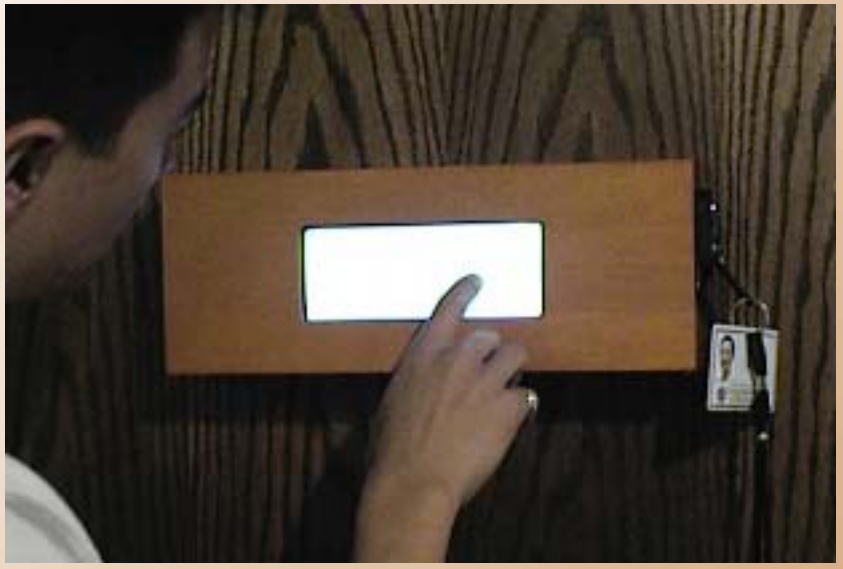
Callback Model

- Transparent communications, always available
- Similar to GUI architectures



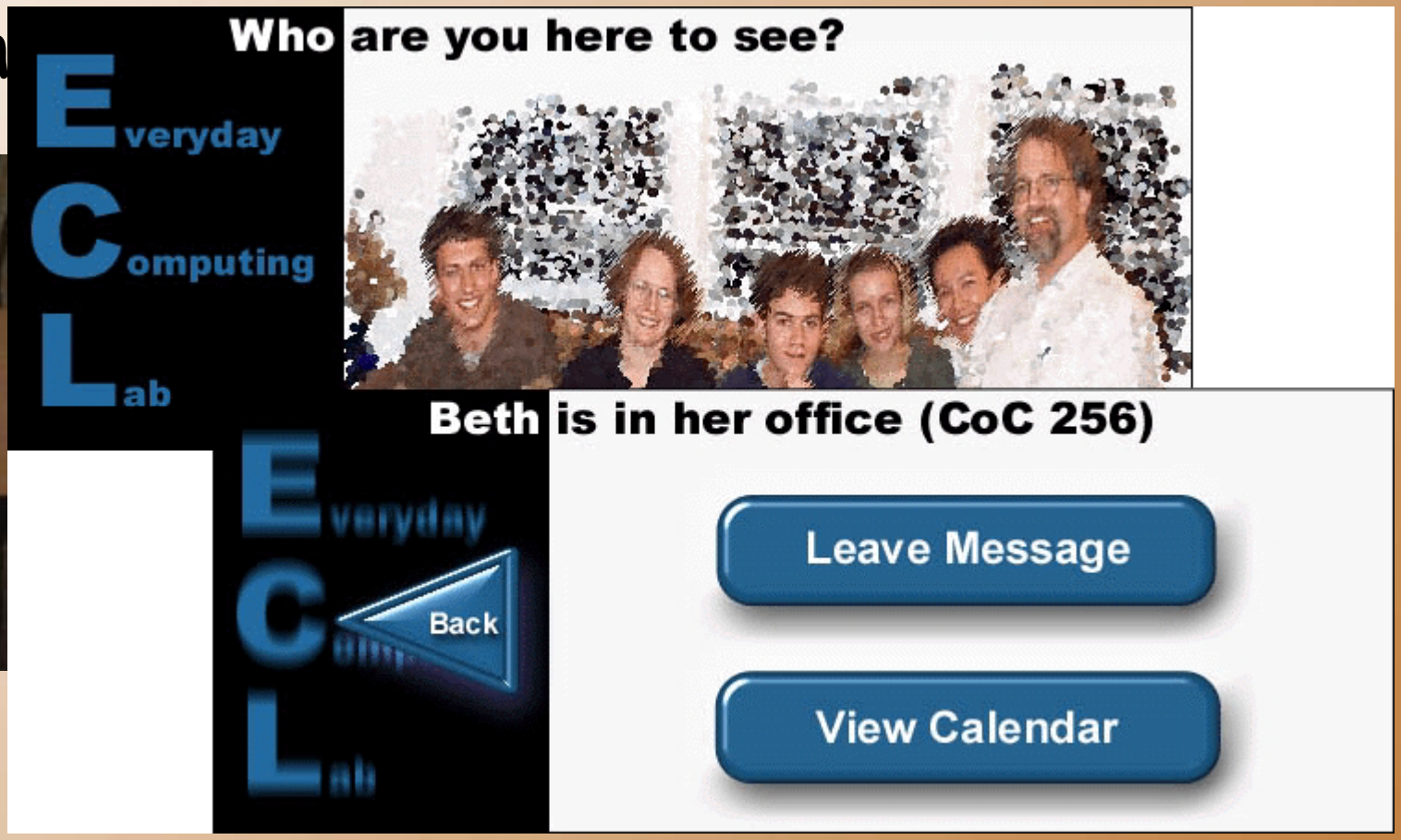
Access Control

- Dynamic Door Displays

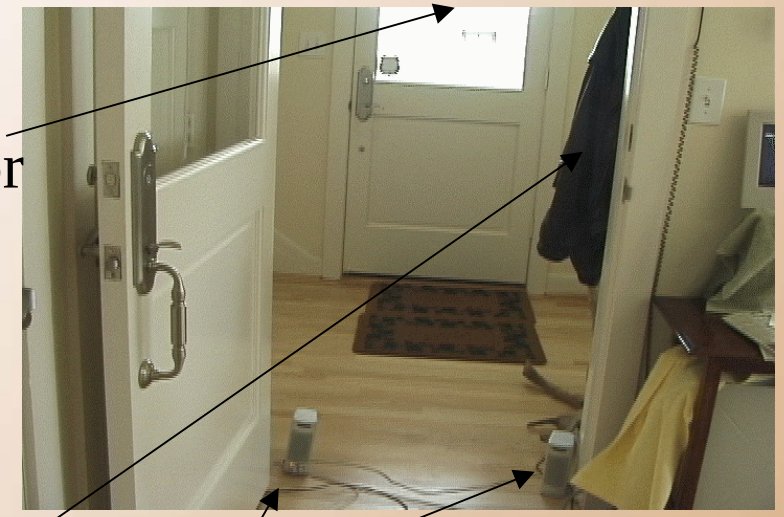


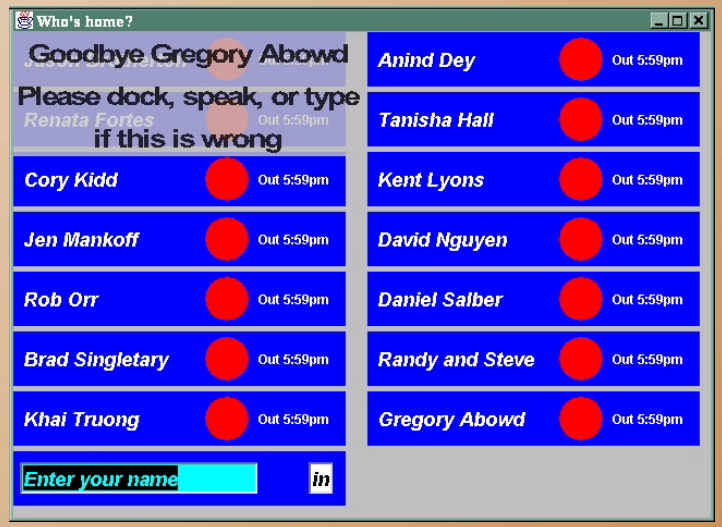
Access Control

- Dyn



Ambiguous Context





motion detector

microphone

speakers

display

dock

keyboard

UbiComp

Experiences: Limitations

- Continuously changing context
- Dealing with unreliable context and other quality of service issues
- Component failure
- Privacy

Important Distinction

Important Distinction

- Behavior that "looks easy" but is not.
  - Star Trek's doors
  - Real-time classroom control
  - Incremental speech recognition improvement

Important Distinction

- Behavior that "looks easy" but is not.
  - Star Trek's doors
  - Real-time classroom control
  - Incremental speech recognition improvement
- Behavior that "looks hard" but is not.
  - Mobile tour guide (GPS, IR beacons)
  - Temporal synching

Distribution of Sensing

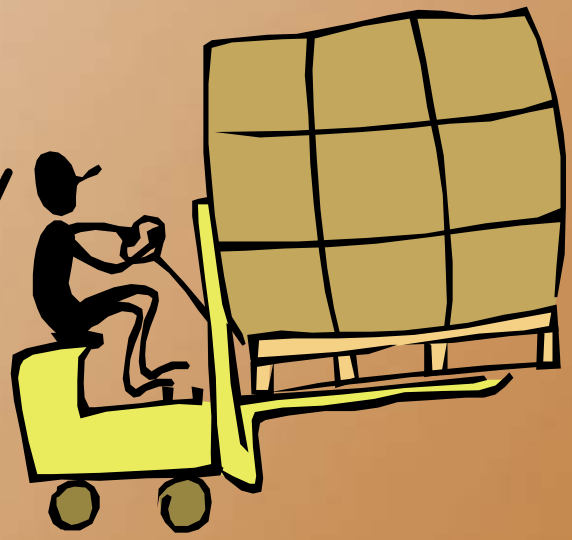
- Heterogeneity of platforms and languages
  - No guarantees on what sensors require
  - No guarantees on what what's available
  - No guarantees on what developers prefer

Abstraction: Interpretation

- Provide meaning to sensed data
- Simple converters
- Complex inferences

Abstraction: Aggregation

- Eases interpretation
- Maps to notion of an entity
- Efficiency mechanism



Component Persistence

- Not like GUI widgets
- Execute autonomously
- Always running

Context History

- Not like GUI widgets
  - Don't want to leave to apps
- Components always running
  - Store data for future apps

Situations: Declarative Style

- Say what you want, not how you want it done – framework figures it out
- Allow programmer to define a situation (complex real-world callbacks)
- *Specification* in design process simpler
- More robust w.r.t. component failures and easier to evolve