COMP90018
Mobile Computing Systems Programming

Prof. Vassilis Kostakos

Logistics

- Lecturers
  - Vassilis Kostakos <vassilis.kostakos@unimelb.edu.au>
  - Tilman Dingler <tilman.dingler@unimelb.edu.au>
- Tutors
  - TBD

Location and Time

- Lectures
  - Tuesdays 12:00 pm - 2:00 pm
  - PAR-The Spot-B01 (Copland Theatre)
- Workshops
  - Multiple days (see official timetable)
  - Hands on tutorials
  - Help with your project from mid-semester

Schedule

Week	Lecture	Workshop
1	Introduction	Compile your first Android app
2	User Interfaces	Build a GUI for mobile & smartwatch
3	Processing sensor data	Threads, services, sensors. Build a compass GUI.
4	Location-based Services	Databases (SQLite and Firebase)
5	Context-aware computing	Connectivity
6	Introduction to the Project	GPS and cloud services
7	Guest Lecture	Help with project
8	Ethnography & evaluation	Help with project
9	Privacy	Help with project
10	Advanced topics	Help with project
11	Field studies	Help with project
12	Revision lecture	Help with project

Assessment

- Group assessment
  - Research: 20% - Due **week 5**
    - Report – 3000 words: 15%
    - Video (upload on youtube) – 5 minutes: 5%
  - Project: 20% - Due **week 12**
    - 30 hours work per student

- Individual assessment
  - Exam: 60% - During exam period

Research - Video

- Group presentation of 5 minutes
  - Choose cool new applications, concepts or ideas that can be shown in less than 5 minutes by a short video (of your own)
  - Has to be related to mobile computing
- Picking a topic
  - Up to your group
  - Confirm/check with tutors

Research - Report

- Structure (suggestion)
  - Title, authors, abstract, HTTP link to video (1 page)
  - Introduction (2 pages)
  - Topic and critical analysis (6-7 pages in two or three sections)
  - Conclusions and future directions (1-2 pages)
  - References and sources (1 page)
  - Itemised contributions by each member (1 page)
- Length
  - Maximum 14 pages
- Submission
  - PDF format
  - Via LMS

Project

- Programming environment
  - Android, Java
  - Hands on experience
- Topic: Develop a context-aware mobile application
  - Must run on phone
  - Must use multiple sensors
  - Must access information from the Internet/cloud
  - Must be usable
  - Must be innovative
  - Must not be trivial to build (not a weekend project)

Project examples

- You can build a game
  - Flappy Bird with Sensory input and a game server
- You can build a utility app
  - Smart City app: use publicly available APIs (e.g. public transport) and your GPS location, and do something interesting
- You can build a social app
  - Foursquare/friends nearby, sharing your context (e.g. my phone is set to silent), location, nearby Points-of-Interest.

What is Mobile Computing?

What is Mobile Computing?

- Who is mobile?
  - The user
- What is mobile?
  - The device
  - The software agent (or process)
- Assumption
  - Network access is availabe (not necessarily always and from anywhere)

What is Mobile Computing?

- Vijay Kumar
  - Information management platform free from spatial and temporal constraints
- Sandeep Jain
  - Technologies that enable people to access network services anyplace, anytime, and anywhere
- Lata Narayanan
  - Computing paradigm that permits users with portable computers to retain their (wireless) network connection even when mobile

Topics in Mobile Computing

- Information management
  - Location-based services
  - Mobile data management
- Technologies
  - WPANs (Bluetooth), WLANs, WANs, RFID, GPS, ...
  - Routing in MANETs, mobile IP, ...
- Computing paradigm
  - Developing software for mobile devices
  - UI design, thin/fat client, ...

4 Challenges of Mobile Computing

- George H. Forman and John Zahorjan
  - Communication
  - Mobility
  - Portability
  - Social impact

Source: Forman and Zahorjan (1994). The Challenges of Mobile Computing, *Computer*, 27(4)

Wireless Communication

- More frequent disconnections
- Lower bandwidth
- Higher latency
- Variation in available bandwidth
- Greater network heterogeneity
- Increased security risks

Mobility

- Address migration
  - Mobile devices used different (IP) addresses
  - Selective broadcast, central services, home base, forwarding pointers
- Location dependent information
  - Information request depends on the location of the device
- Migrating locality
  - Connections should be dynamically transferred to servers that are (geographically) closer

Portability I

- Energy usage
  - Batteries are the most important source of weight (e.g., iPad more than 20% (148g),
  - Power consumption is proportional to CV^2F
  - C: capacitance can be reduced by VLSI design
  - V: voltage can be reduced by smaller structures
  - F: clock frequency

Portability II

- Risk to data
  - Portable devices are more vulnerable to loss or damage
- Resource-poor relative to static devices
  - Given a certain cost and level of technology, processor speed, memory size, and storage capacity are always smaller compared to their static counterparts
- Small user interface

Social Impact

- Privacy
- Security
- Behaviour

Wired versus Wireless Networks

Wired Networks	Wireless Networks
High bandwidth	Low bandwidth
Low bandwidth variability	High bandwidth variability
Possibility to listen on wire	Hidden terminal problem
Physical access (security)	Requires proximity
Low delay	High delay
Connected operation	Disconnected operation

Milestones for Mobile Systems

- 1928
  - First mobile radio used by the Detroit Police Department
- 1940s
  - Limited cellular networks
  - Implementation of the first mobile radio system to connect with a fixed telephone number
- 1950s
  - Bell Labs tests cellular techniques
- 1970s
  - Cellular service proposed by AT&T

Mobile Computing and ...

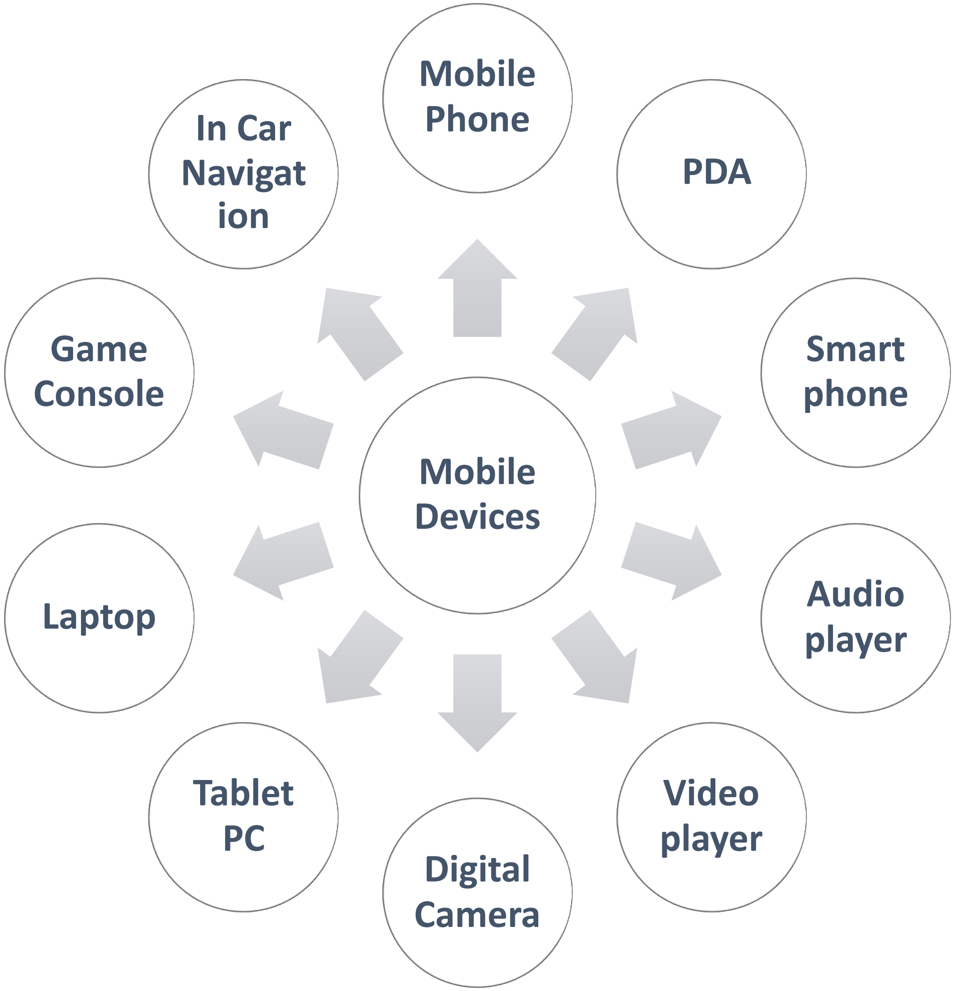
- Context-awareness
  - Obtain information from the environment and dynamically configure services
- Ubiquitous computing = ?
  - Mobile computing + context awareness
- Wearable computing
- Mark Weiser’s vision of ubiquitous computing
  - *The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.*

Are we there yet …

- Digital mobile phone networks
- Handheld devices
- GPS
- WLAN
- To come
  - RFID
  - WiMaX
  - UWB

Smartphones

Mobile Computing Devices



Mobile Devices

Mobile Phone

PDA

Smart phone

Audio player

Video player

Digital Camera

Tablet PC

Laptop

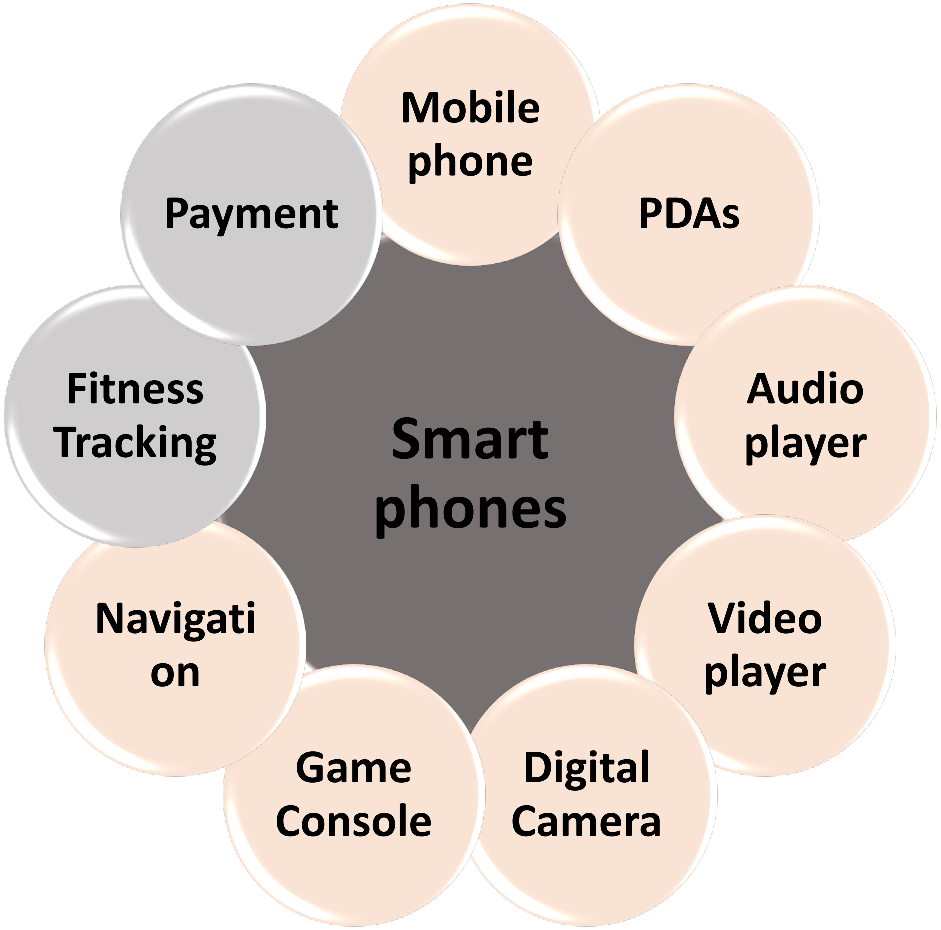
Game Console

In Car Navigat ion

Overview of Handheld Devices

- Handheld (device)
  - Portable device that supports wireless communications or information management (or both)
- Mobile phone
  - Portable telecommunication device
- PDA (Personal Digital Assistant)
  - Handheld for personal information management (PIM)
- Smartphone
  - Handheld combining mobile phone and PDA functions

Smartphones



Smart phones

Mobile phone

PDAs

Audio player

Video player

Digital Camera

Game Console

Navigati on

Fitness Tracking

Payment

Ubiquity of Mobile Phones

- Status quo
  - 4.8 billion mobile phone users*
  - Several million PDAs (no precise numbers)
- Number of mobile phones ≫ number of PDAs
- Number of mobile phones > number of PCs!
- Smartphones integrate PDA functionality
  - Are likely to serve as a convergence device

*http://www.statista.com/statistics/274774/forecast-of-mobile-phone-users-worldwide/

The First Smartphone

- Simon
  - Developed by IBM in 1992, distributed by BellSouth in 1993 (original price: USD 899)
  - Capabilities: address book, appointment calendar, calculator, sketchpad, world clock
  - Communication: email (no Internet), fax, phone
  - GUI: icons, background patterns, round rectangles
  - B&W display (160 x 293)
  - Quite large and heavy

Source http://cdecas.free.fr/computers/pocket/simon.php

Smartphone Applications I

- Digital purchases
  - Download and use products immediately
  - Ringtones, games, …
- Mobile shopping
  - Price comparison
  - Product information
- Mobile advertising
  - User profiles → personalized offers
  - But: unsolicited messages

Smartphone Applications II

- Information services
  - Ubiquitous access to information
  - Stock quotes, weather information, news, …
- Location-based services
  - Where am I, what is nearby, how do I get from A to B?
  - Context-aware applications
- Advanced wireless and 3G services
  - Access to multimedia files (podcasting)
  - Mobile TV

The Future of Smartphones I

- Mobile banking
  - Access bank accounts from everywhere (Internet banking on the go)
  - Using a mobile for payments (digital cash)
- Speech recognition
  - Voice dialing -> verification, text input, translation, …
- Barcode reader
  - Dedicated or integrated with the camera
  - Shopping, information, payment

The Future of Smartphones II

- Increase range of wireless services
  - WiMax
  - Peer to peer phones
- Integration with sensors
  - GPS, accelerometer, temperature, …
- Overcome limitations in screen size
  - Folding e-paper
  - Built-in projectors
  - Smart walls

Operating Systems

- Symbian
  - Nokia, Panasonic, Siemens AG, Sony Ericsson
  - Predecessor: EPOC
  - Pre-emptive multitasking, multithreading, memory protection
  - Support for C++, Java, Python, …
- Windows Mobile
  - Compact OS based on Microsoft’s Win32 API
  - Pre-emptive multitasking, multithreading, memory protection
  - Support for C#, Visual Basic, C++,…

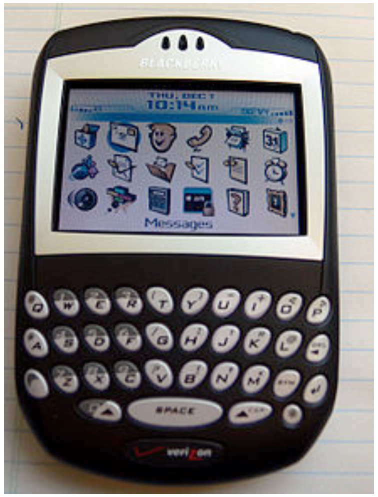
Operating Systems

- Linux
  - Major player (Android)
  - Pre-emptive multitasking, multithreading, memory protection
  - Many programming languages

- Palm OS
  - Palmsource
  - No (per-process) memory protection, no multithreading, no pre-emptive multitasking
  - Palm OS Cobalt supports multitasking and multithreading
  - Support for C++, Java, Visual Basic, …
  - Essentially dead now …

Operating Systems

- RIM
  - Proprietary multi-tasking OS for BlackBerry
  - Specializes on input devices for mobile phones (keyboard, scroll wheel, track ball)
  - Good support for Java MIDP

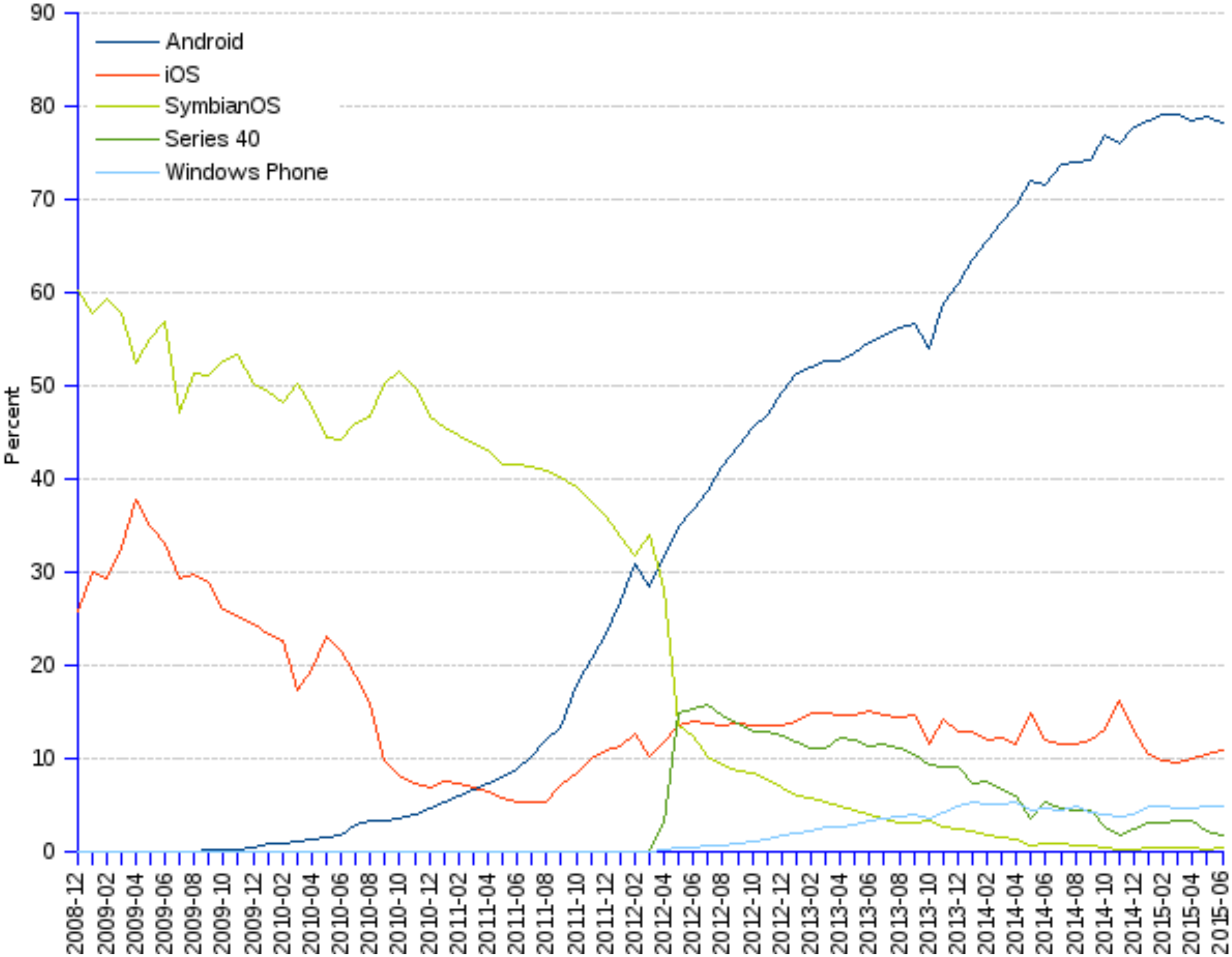


Operating Systems

- Android
  - Not an OS but a software stack that uses Linux
  - Java (Dalvik virtual machine)
  - Integrated browser based on open source (WebKit)
  - See http://www.netmite.com/android/ to run any J2ME or MIDP application

Operating Systems

- iPhone OS (Apple)
  - Unix-like OS, kernel with memory protection, multithreading
  - Multitasking was limited before iOS 4 to Apple apps (iPod, Mail)


Android
iOS
SymbianOS
Series 40
Windows Phone
Percent
90
80
70
60
50
40
30
20
10
0
2008-12
2009-02
2009-04
2009-06
2009-08
2009-10
2009-12
2010-02
2010-04
2010-06
2010-08
2010-10
2010-12
2011-02
2011-04
2011-06
2011-08
2011-10
2011-12
2012-02
2012-04
2012-06
2012-08
2012-10
2012-12
2013-02
2013-04
2013-06
2013-08
2013-10
2013-12
2014-02
2014-04
2014-06
2014-08
2014-10
2014-12
2015-02
2015-04
2015-06

Development Comparison I

	Symbian	Java ME	Python	.Net Compact	Micro-browser Based	iPhone OS	Android
Language	C++	Java	Python	C#, VB	XHTML, WML	Objective C	Java
Learning Curve	Steep	Average	Excellent	Average	Varies	Steep	Average
Debugger	Good	Excellent	Average	Excellent	Good	Excellent	Excellent
Emulator available	Free Emulator	Free Emulator	Add-on to Nokia Emulator	Included in Windows Mobile 7	Many	Yes (Xcode)	Free Emulator
IDE	Many choices	Eclipse, WTK, NetBeans	Several & Eclipse	Visual Studio 2010	Many	Xcode	Eclipse, Undroid (NetBeans)
Develop-ment Cost	Varies	Free	Free	Free for basic tools	Free	Free/$99	Free/$25

Source: http://en.wikipedia.org/wiki/Mobile_development

Development Comparison II

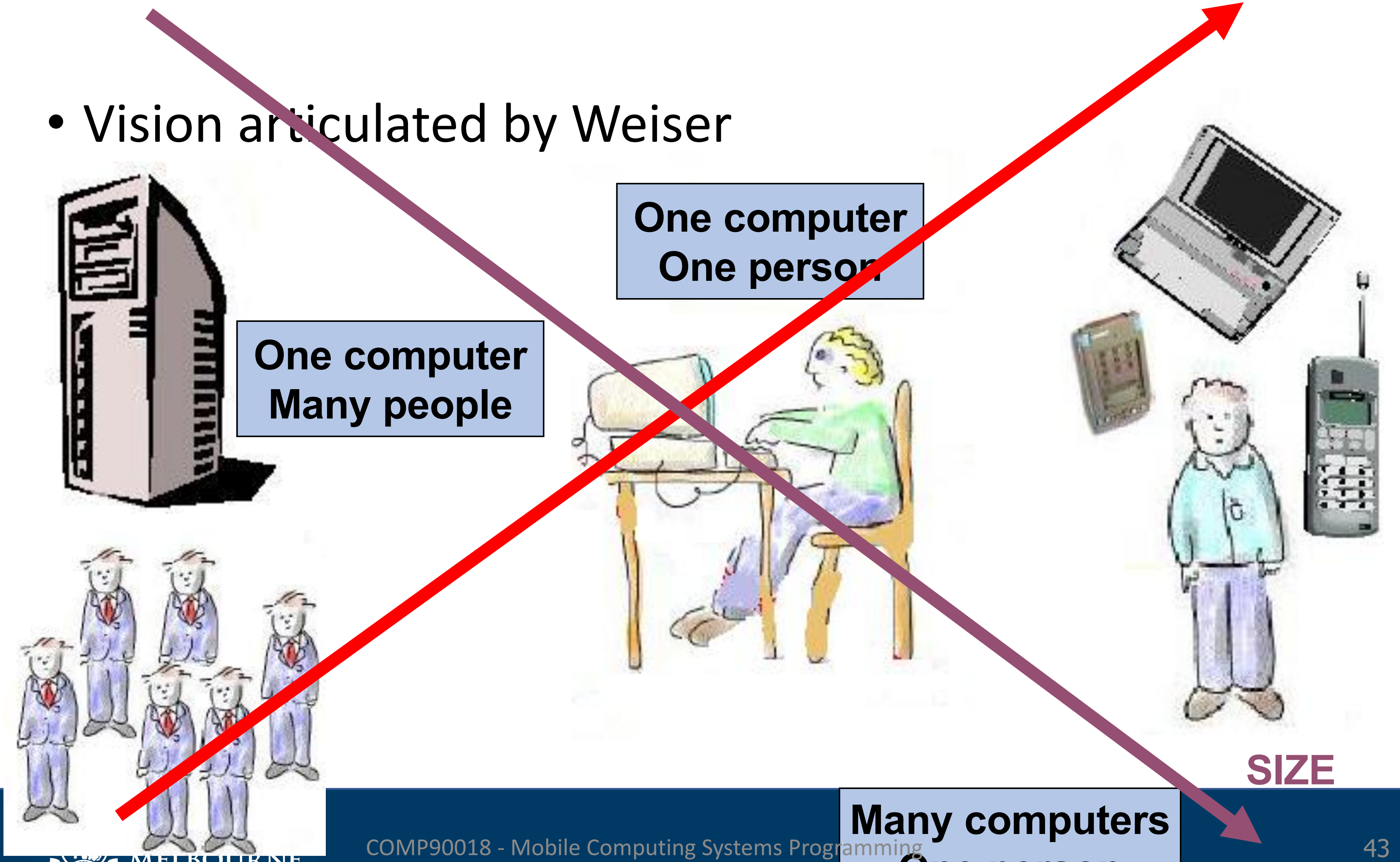
	Symbian	Java ME	Python	.Net Compact	Micro-browser Based	iPhone OS	Android
Cross-Platform Deploym.	Compile per target	Excellent: Bytecode	Nokia Series60	Windows Pocket PC	Excellent	iPhone, iPod, iPad	Android handsets only
Graphical Interface	2D, 3D, many widgets, Visual GUI Builder	2D, 3D, many widgets, Visual GUI Builder	2D access, some widgets	2D, many widgets, Visual GUI Builder	N/A	2D, 3D, widgets, Visual GUI builder	2D, 3D, widgets, Visual GUI builder
Runtime Speed	Best	Average (Bytecode)	Average (Interpreted)	Good	Average	Best	Good
Market	Extensive	Extensive	Average	Average	Extensive	Extensive	Extensive

Source: http://en.wikipedia.org/wiki/Mobile_development

Ubiquitous Computing

Overview

- Vision articulated by Weiser



NUMBER

One computer
Many people

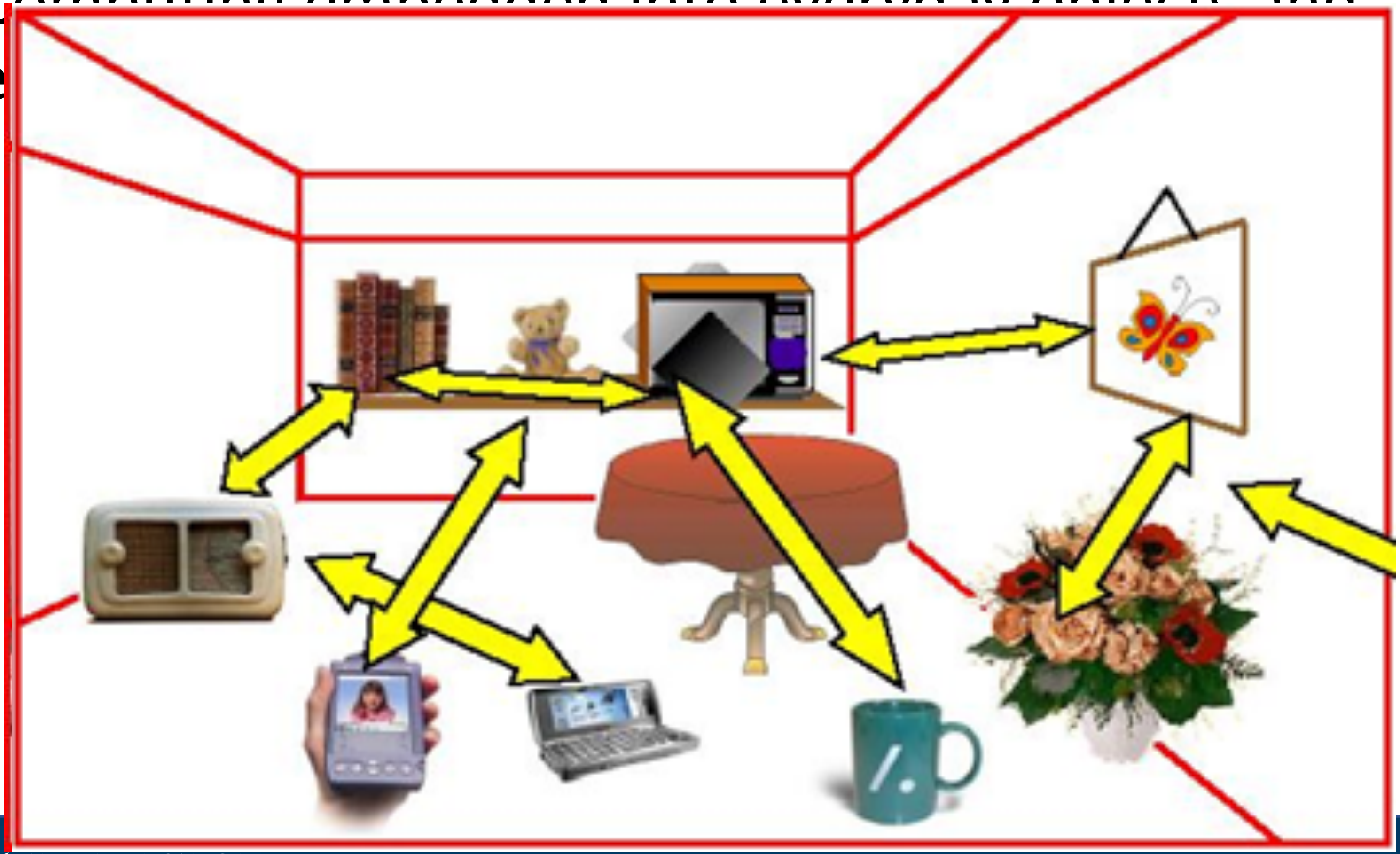
One computer
One person

Many computers
One person

SIZE

Overview

- Computing embedded into everyday objects and e



Context-Aware Computing

- Context: situational elements relevant to interaction between user, application, environment
- Context-awareness: situationally appropriate; apps adapting to context, increasing value to users
  - Using sensors and actuators to improve human-computer interaction and (computer-mediated) human-human and human-environment interaction
- Examples: tour guide, reminders, diary retrieval
- Difficulties in building:
  - Lack of abstractions for acquiring/using context
  - Lack of design principles
  - Distributed context sensing

30 Years of Ubicomp Research

- Systems-level contributions mostly
- Lots of people working on tools for building these types of applications
- No doubt that ubicomp will become a reality:
  - Dystopian a la Minority Report
  - More utopian
- Challenge for ubicomp

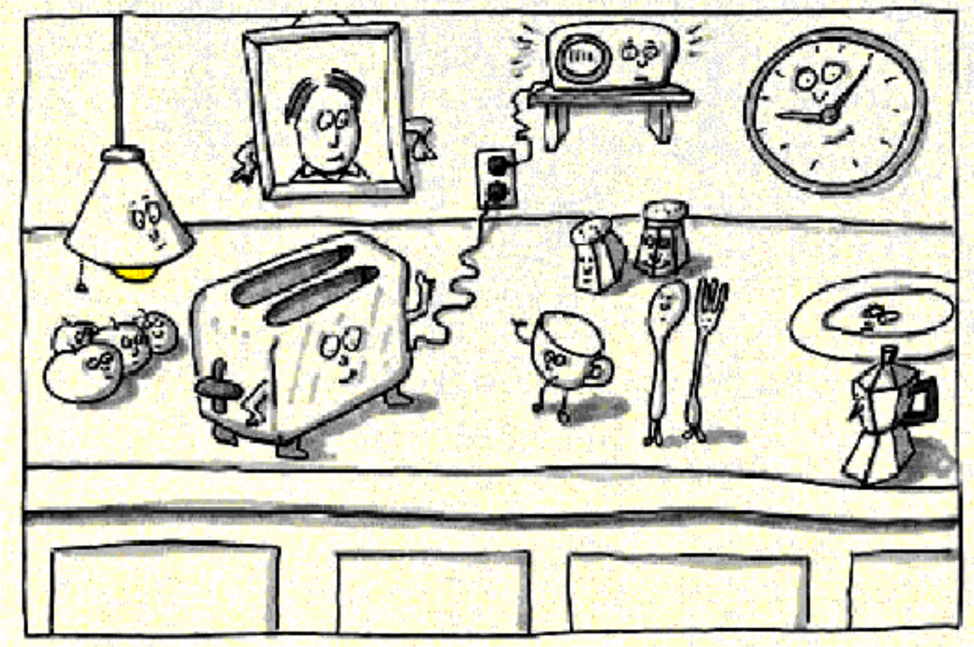
Not the Interesting Problem!

- Despite all these people building systems, no one using them or liking them
    - Not a technology issue
    - Not a development issue
    - Not a cost issue

    - So, what is the issue → needs of the user

Feeling in Control

- Dourish, Abowd and Mynatt, Bellotti and others: lack of control in these environments
- Information collected, synthesized and used implicitly
- How do I know what's going on? (feedback)
- How do I change what's going on? (control)
- Who gets this information? (privacy)
- Is this another way to SPAM me? (overload)



Rich Gold "Dancing Toaster"

Types of context-aware apps

- Various ways of classifying CA software:

- Proactive Triggering
  - e.g. performing some interaction based on environmental perceptions
- Streamlining Interaction
  - Reducing irrelevant information - e.g. travel guide for current location
- Memories for past events
  - Contextual retrieval, e.g. based on spatial or feature-based cues
- Reminders for future contexts
  - e.g. tagging details regarding current context for future access
- Optimising patterns of behaviours
  - e.g. changing interface based on situation (from screen to text-to-voice)
- Sharing Experiences
  - Social networking based on shared contexts

Types of context-aware systems

- Passive context-aware systems
  - New context is presented to the user, to inform them of change
  - User can then explicitly determine if the use of an application should change
  - Examples
    - Changing information regarding environment (weather meters)
    - Context-based tagging (e.g. by camera)
- Active context-aware system
  - Behaviour of the applications change automatically
  - Examples
    - Task filtering (e.g. information filtering, based on current wireless network speed)
    - Context-based task activation (e.g. routing)
    - Content adaptation (e.g. for people with disability)

Deriving context (low->high)

- New contexts can be created based on sensor data (captured by the device or nearby sensors)
- Lower-level raw contexts may need to be processed into higher level contexts
- Raw data may need to be scaled or transformed
  - e.g. electrical signal on a temperature gauge should be converted into a Celsius value...
  - ...or absolute geo-location position should be converted into an address identification of a building
- Often this abstraction is more useful
  - e.g. “this photo was taken at my parents home last christmas”

Deriving context (c1->c2)

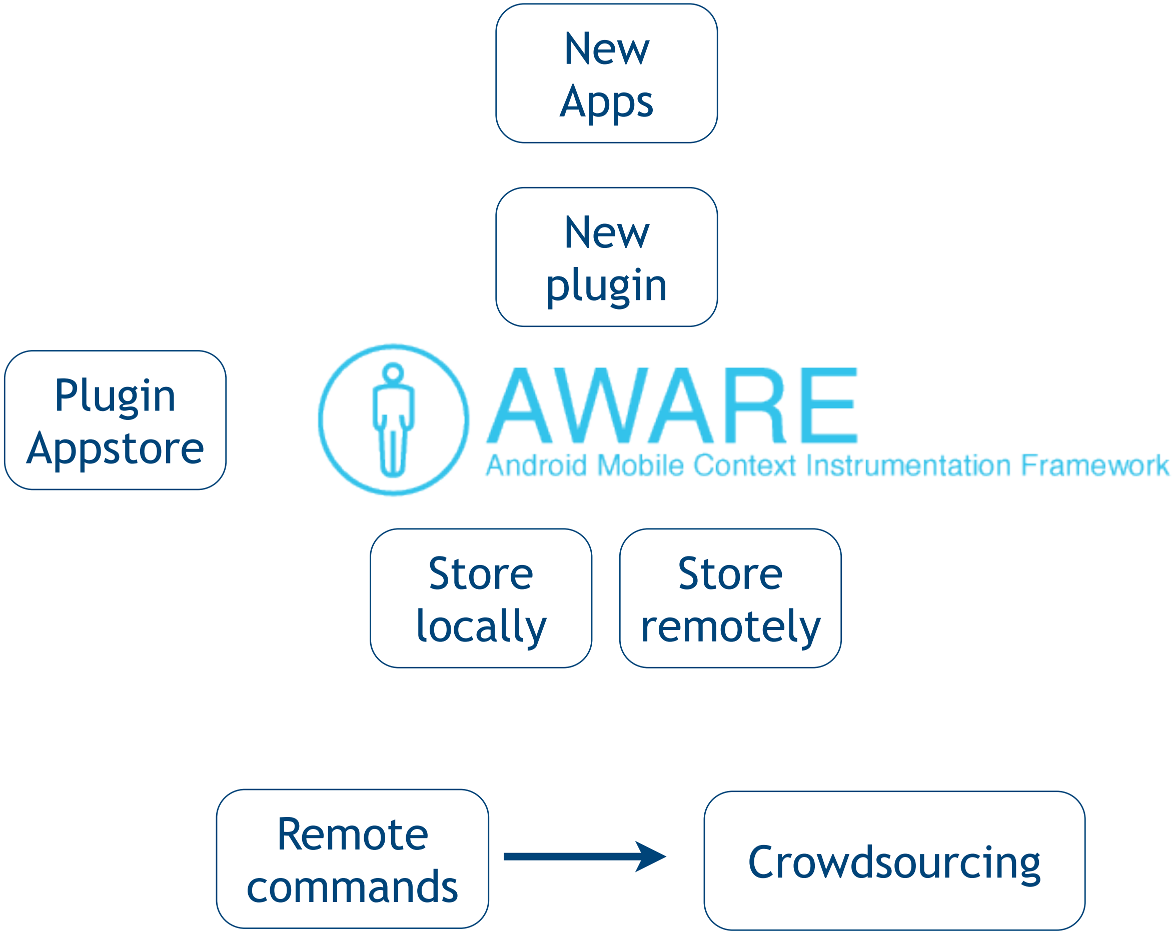
- Some contexts can be derived from other contexts
  - By combining several contextual cues, a better understanding of the context can be determined, than considering the cues in isolation
- The same cues might mean different things
  - A user in front of a building may be admiring the architecture, or waiting for a friend
- Good for implicit HCI - iHCI systems
  - e.g. knowing the current location, current time, and having access to the user’s calendar, an app can infer the user’s context
  - i.e. waiting for a flight, in a meeting, or even en-route, but running late.
- Cues may be heterogenous in format
  - Currently over 100 location coordinate systems are in use
  - Need mechanisms to resolve heterogeneity within open environments

Deriving context (c1+c2)

- Combining several simpler contexts - Context Composition
- The joint inference of simpler contexts can imply other contexts
  - May require interoperation and mediation
- Combining homogeneous contexts
  - Getting several independent values about the same thing
- Combining heterogeneous contexts
  - Additional cues may disambiguate contexts suggested by other cues
- Deriving high-level context from lower-level ones
  - e.g. by knowing someone’s weight and location, an identity may be derived
- Deriving low-level contexts from higher-level ones
  - e.g. a physical location may be derived from more general information such as the street name and building name given by a third party

Further approaches

- Consulting a user profile or preferences
  - Checking the user’s calendar - not always reliable as the user may deviate...
- Using historic data
  - Observing user behaviour and building an implicit user profile
- Explicitly ask the user
  - “Training”
  - May be annoying to users



New Apps

New plugin

Plugin Appstore

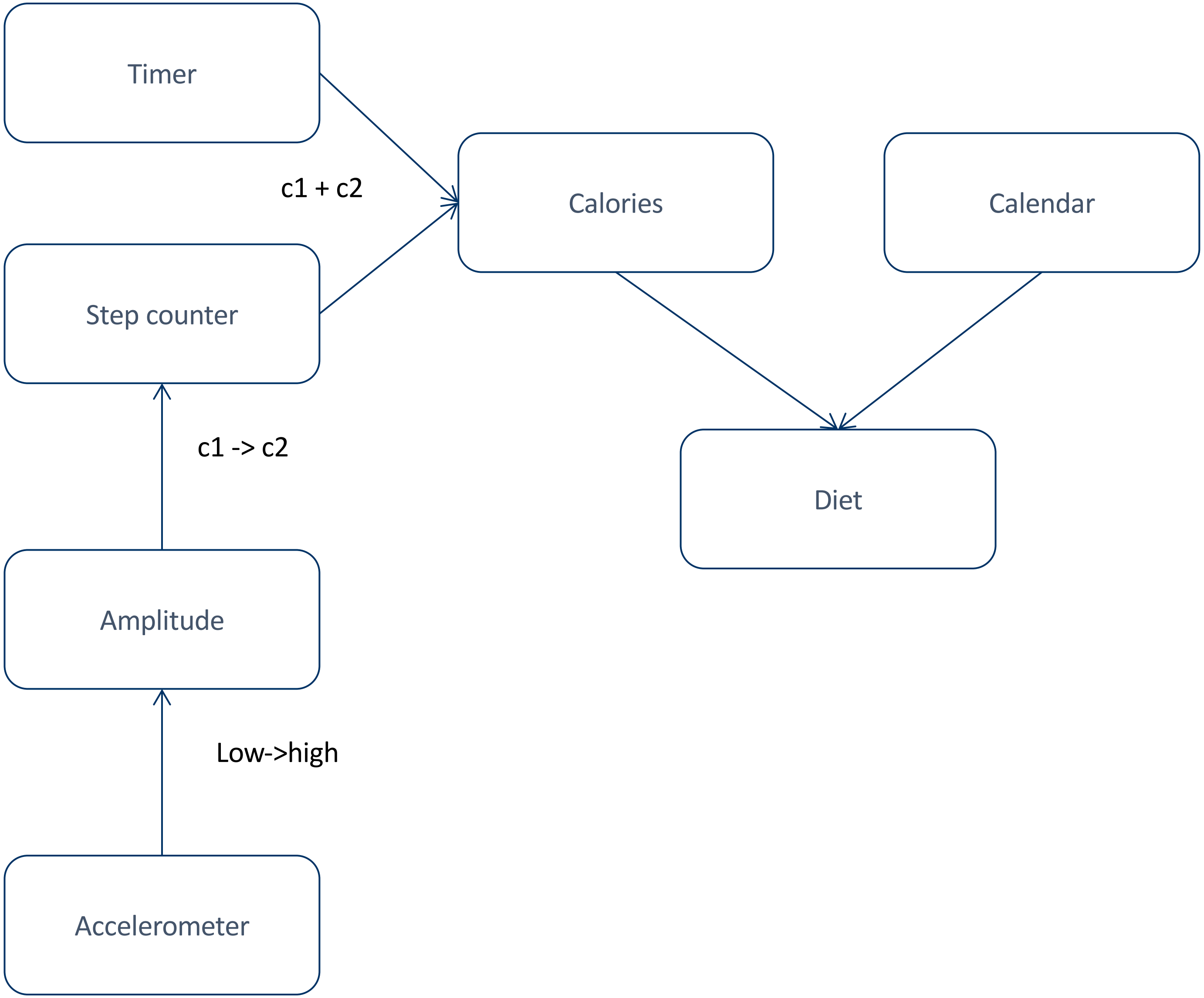
AWARE

Android Mobile Context Instrumentation Framework

Store locally

Store remotely

Remote commands → Crowdsourcing


Timer
c1 + c2
Calories
Calendar
Step counter
c1 -> c2
Diet
Amplitude
Low->high
Accelerometer

Activity

- Make a similar diagram
- Objective: the smartphone should detect if you are drunk or not

Summary

- Ubiquitous Computing: Interaction design beyond the desktop/mobile
    - Technology is “everywhere”
    - The environment becomes smart
- Computers can capture context
- Computers can infer/calculate context
- As a designer, how would you use context in your application.