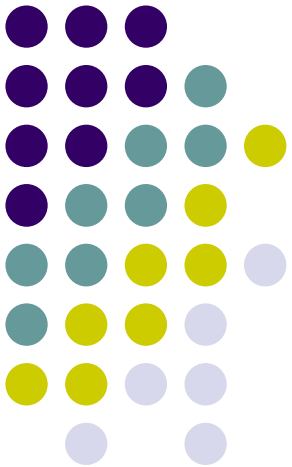
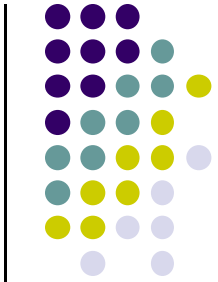
CS 528 Mobile and Ubiquitous Computing
Lecture 1: Introduction

Emmanuel Agu

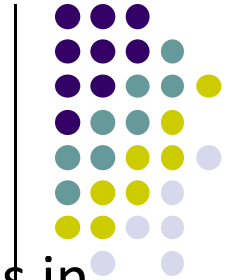


A Little about me



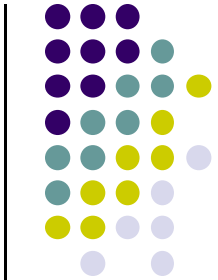
- Faculty in WPI Computer Science
- Research interests:
  - mobile computing especially mobile health, computer graphics
- How did I get into mobile and ubiquitous computing
  - 3 years in wireless LAN lab (*pre 802.11*)
  - Designed, simulated, implemented wireless protocols
  - Group built working wireless LAN prototype (*pre 802.11*)
- Computer Systems/Electrical/Computer Science background
  - Hardware + software
- Current active research: Mobile health apps

About this class (Administrivia)



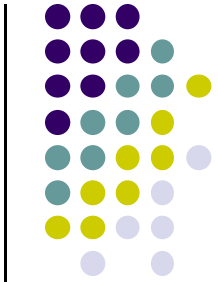
- **Class goal:** overview, insight into hot topics, ideas and issues in mobile and ubiquitous computing
- **Focus:** implement ideas on Android smartphone
- **Semester break:** March 10 (no class)
- **Website:** http://web.cs.wpi.edu/~emmanuel/courses/cs528/S13/
- Projects: 3 assigned, 1 big final project
- This area combines lots of other areas: (networking, OS, software, machine learning, programming, etc)
  - Most students don't have all the background!!
  - **Independent learning is crucial!**
  - **Final Projects:** Make sure your team has requisite skills

Administrivia: Schedule



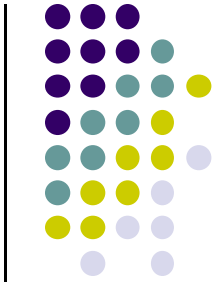
- ***Week 1-6:*** I will present (course introduction, Android programming)
- ***Weeks 7 – 8:*** Students will present papers
  - Goal: examine cutting edge research ideas
  - Student talks short and direct (~15 minutes)
  - Discussions
- ***Week 9:*** Students propose final project
- ***Weeks 10-13:*** Students present more papers
- ***Week 14:*** Students present + submit final projects
- Each week, 15-min break halfway

Formal Requirements



- *What do you have to do to get a grade?*
- **Seminar:** Come to class + Discuss + Do good projects!!
- Each student will present 1 or 2 papers
- Weeks 7-8,10-13: Submit summaries for any 2 of week's papers
- Do projects: assigned and final project(s)
- Final project: 5-phases (See website for deadlines)
  - Pick partner + decide project area
  - Brainstorm on ideas
  - Submit intro + related work + proposed project plan (week 9)
  - Build, evaluate, experiment, analyze results
  - Present results + submit final paper (in week 14)
- **Grading policy:** Presentation(s) 15%, Class participation 5%, Assigned Projects 25%, Final project: 40%, Summaries: 15%

Written Summaries



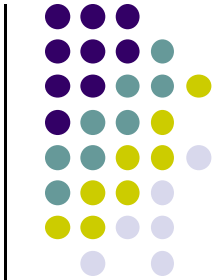
- Submit using turnin *before class*
- Summarize key points of any 2 of papers for week
  - Main ideas
  - Limitations of the work
  - What you like/not like about paper
  - Any project ideas?
- Half a page max per paper
- Summary should quickly refresh memory in even 1 year's time
  - Include main ideas/algorithms, results, etc.
- See handout for more details

Course Text

- **Text:** The Busy Coder's Guide to Android to Android Development by Mark Murphy version 6.3 (Covers Android version 5.0)

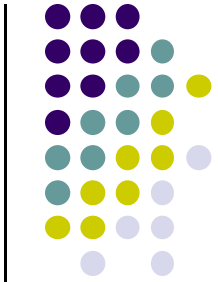
- Android API changes often, so book uses annual subscription
- U$45 annual subscription gives 1 year access to book updates
- **Free to all registered students in this class!!**
- Many different formats of book (pdf, apk file, kindle, etc)
- Lots of free working demo apps available on github
  - http://github.com/commonsguy/cw-omnibus
- Divided into **core sections** and **trails** (optional)
  - **Core sections:** must be followed in sequence
  - **Trails:** Can be read in any order

Poll Question



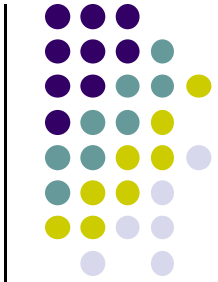
- How many students:
  - Own Android phones
  - Can borrow Android phones for projects (e.g. from friend/spouse)?
  - Do not own and cannot borrow Android phones for projects?

Mobile vs Ubiquitous Computing

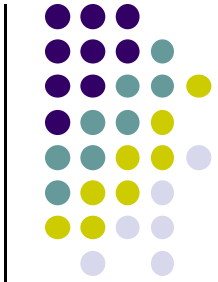


- Mobile computing
  - mostly *passive* network components
  - Human computes while moving, continuous network connectivity
  - **Note:** Human initiates all activity, clicks on apps!!
  - **Example:** Using *foursquare.com* on smart phone
- Ubiquitous computing
  - Collection of specialized assistants to assist human in tasks (reminders, personal assistant, staying healthy, school, etc)
  - Array of *active* elements, sensors, software agents, artificial intelligence
  - Builds on *mobile computing* and *distributed systems* (more later)
  - **Note:** System/app initiates activities, inference
  - **Example:** Google Now on smartphone

Ubicomp Sensing



- Sense what?
  - *Human:* motion, mood, identity, gesture
  - *Environment:* temperature, sound, humidity, location
  - *Computing Resources:* Hard disk space, memory, bandwidth
  - *Ubicomp example:*
    - *Assistant senses:* Temperature outside is 10F (environment sensing) + Human plans to go work (schedule)
    - *Ubicomp assistant advise:* Dress warm!
- Sensed **environment + Human + Computer resources** = ***Context***
- *Context-Aware* applications adapt their behavior to context



Sensing the Human

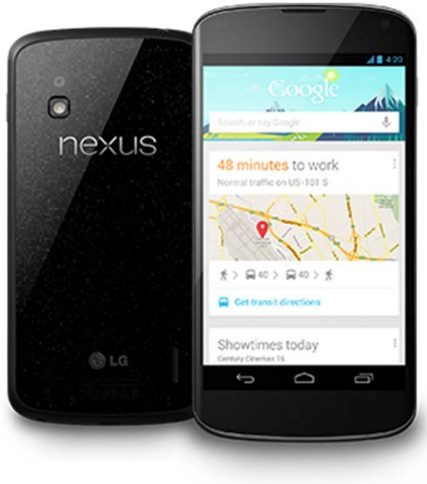
- Environmental sensing is relatively straight-forward
  - Use specialized sensors for temperature, humidity, pressure, etc
- Human sensing is a little harder (ranked easy to hard)
  - **When:** time (Easiest)
  - **Where:** location
  - **Who:** Identification
  - **How:** (Mood) happy, sad, bored (gesture recognition)
  - **What:** eating, cooking (meta task)
  - **Why:** reason for actions (extremely hard!)

5 W's + 1 H

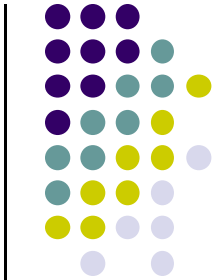
- Human sensing (gesture, mood, etc) easiest using cameras
- Research in ubiquitous computing integrates
  - location sensing, user identification, emotion sensing, gesture recognition, activity sensing, user intent

Mobile Devices

- Smart phones (Blackberry, iPhone, Android, etc)
- Tablets (iPad, etc)
- Laptops



SmartPhone Hardware



- Communication: Talk, text, Internet access, chat
- Computing: Java apps, JVM, apps
  - Powerful processors: Quad core CPUs, GPUs
- Sensors/Multimedia: Camera, video, accelerometer, etc
- Smartphone = Communication + Computing + Sensors
- Google Nexus 5 phone: Quad core 2.5 GHz CPU, Adreno 330 GPU

	Nexus 4	Galaxy S III	iPhone 5	Moto Droid
CPU	APQ8064	MSM8960	Apple A6	OMAP 3430
	1.7 GHz **Quad**-core	1.7 GHz **Dual**-core	1.3 GHz **Dual**-core	600 MHz
GPU	Adreno 320	Adreno 225	PowerVR SGX543MP3	PowerVR SGX 530
	OpenGL ES 3.0 **OpenCL 1.2** OpenVG 1.1	OpenGL ES 2.0 OpenVG 1.1	OpenGL ES 2.0 Shader Model 4.1	OpenGL ES 2.0 Shader Model 4.1
	NA **40-45 GFLOPS**	400 MHz **19.2 GFLOPS**	266 MHz (**Tri**-core) **25.5 GFLOPS**	200 MHz (1.6 GFLOPS)

GLOPS: floating-point operations per second

Comparison courtesy of Qian He (Steve)

Smartphone Sensors

- Typical smartphone sensors today
  - accelerometer, compass, GPS, microphone, camera, proximity

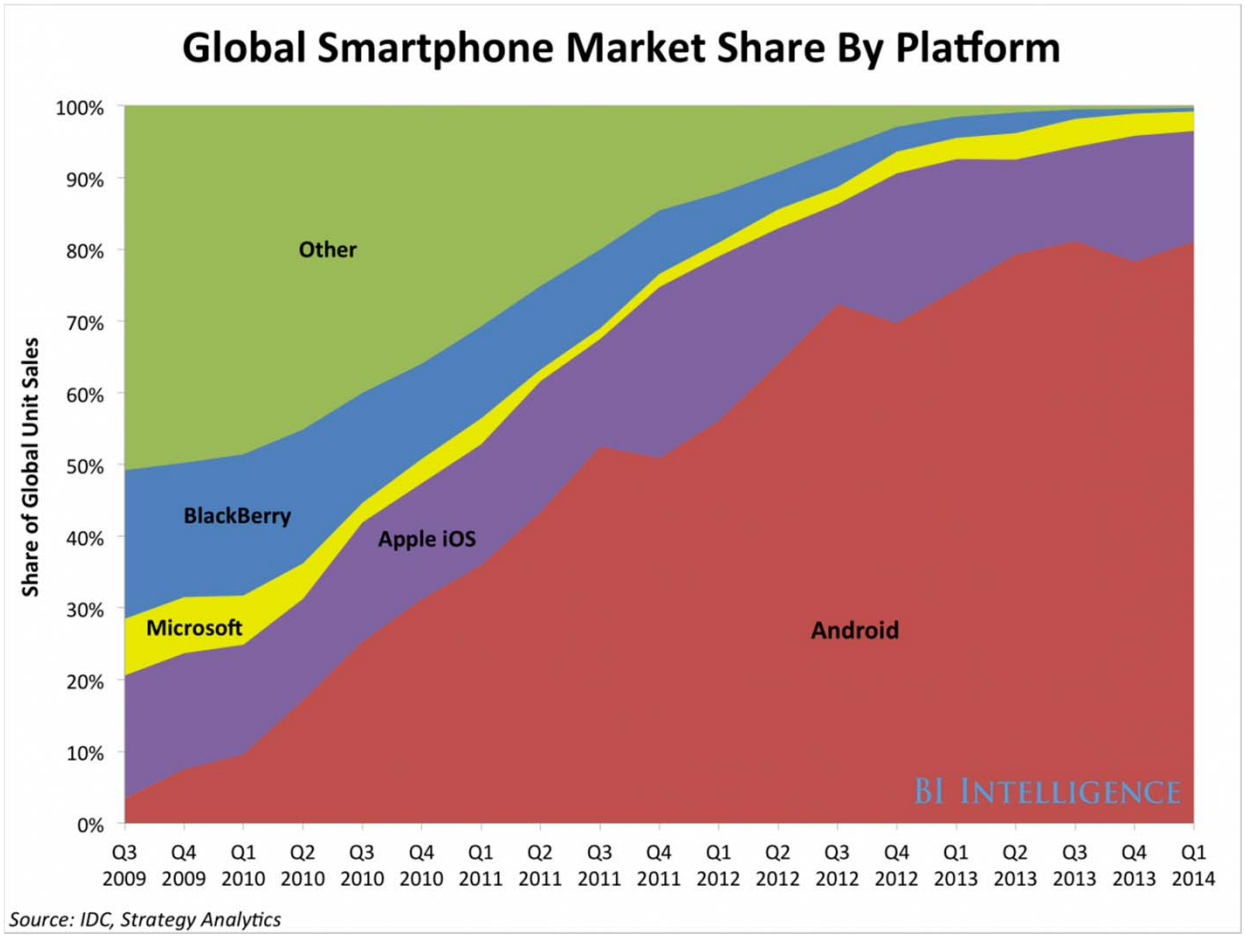
Future sensors?

- Heart rate monitor,
- Activity sensor,
- Pollution sensor,
- etc

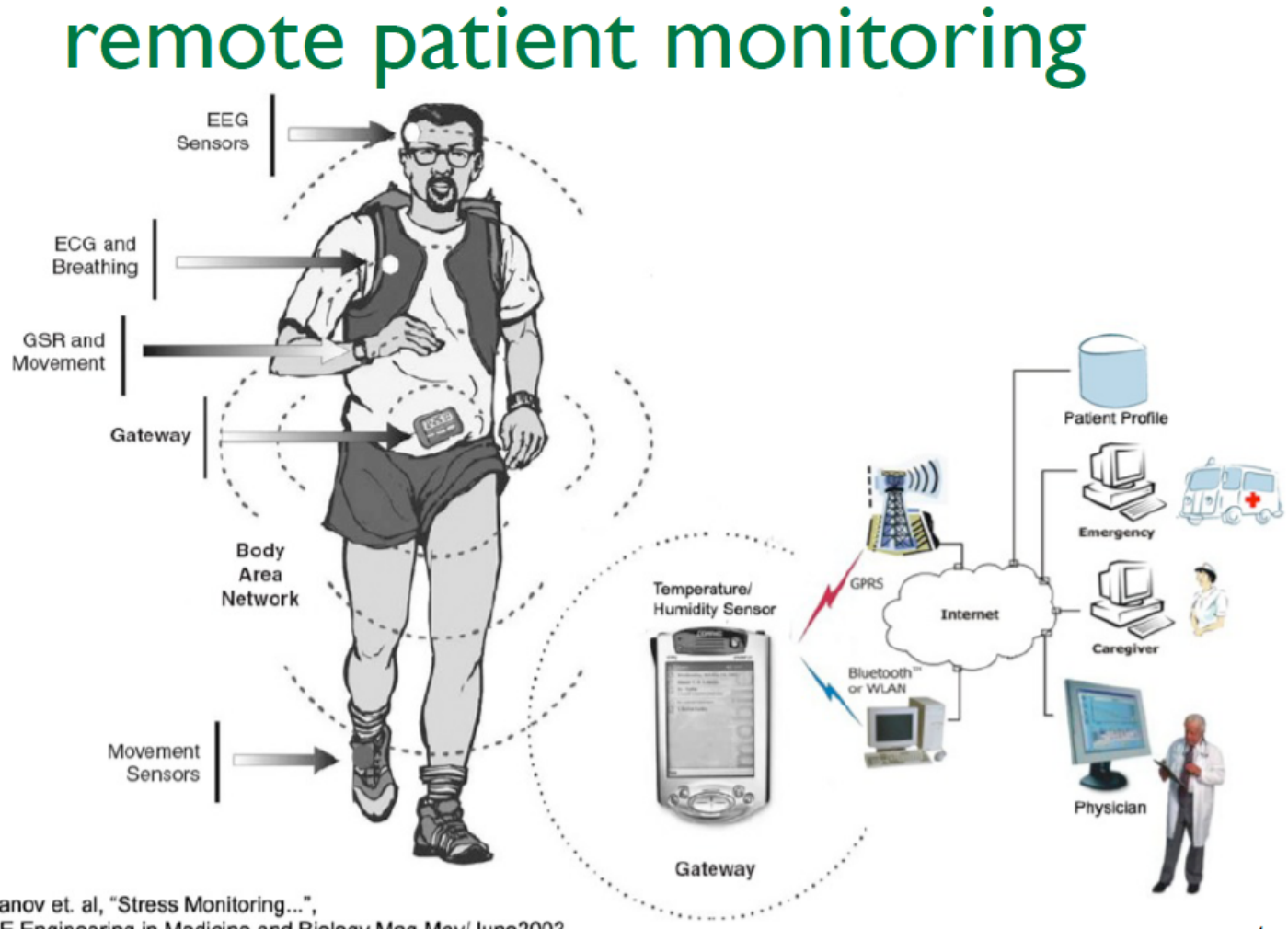
SmartPhone OS

- Over 80% of all phones sold are smartphones
- Android share 78% worldwide
- iOS 18%

Source: IDC, Strategy Analytics



Ubiquitous Computing: Wearable sensors for Health



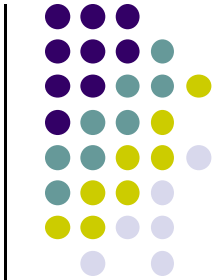
Jovanov et. al, "Stress Monitoring...",
IEEE Engineering in Medicine and Biology Mag.May/June2003

External Sources of Data

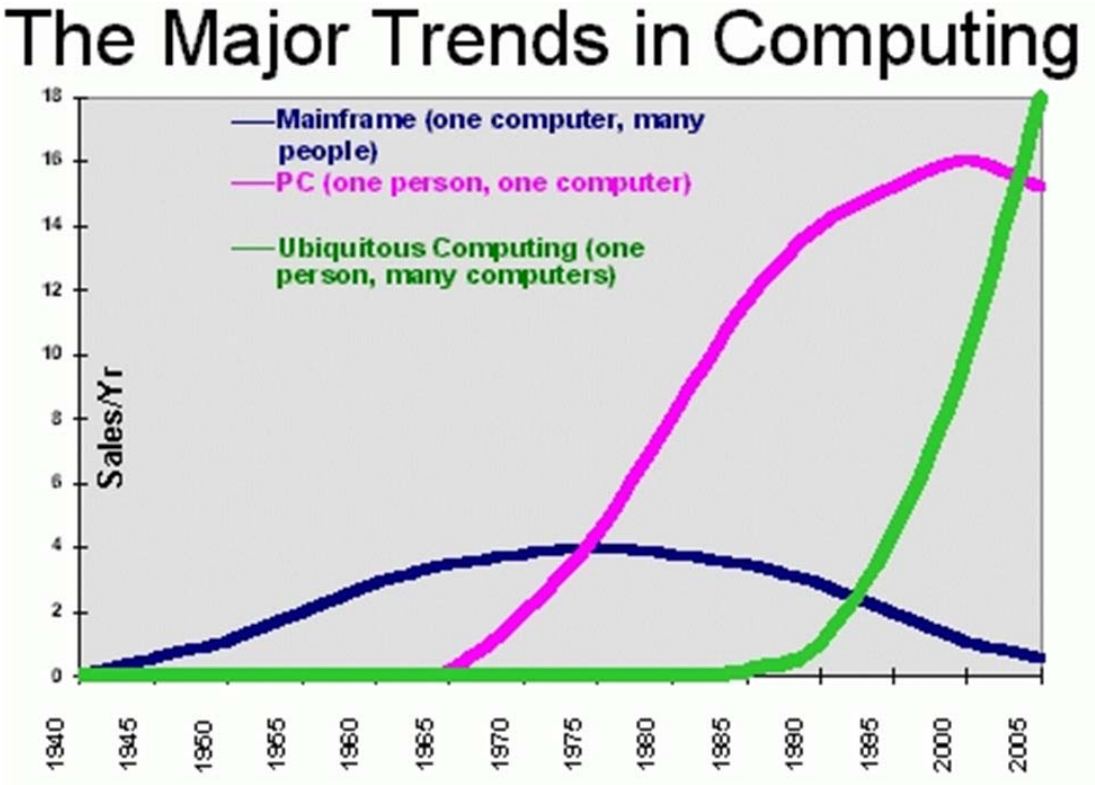
Body Worn Activity Trackers

Bluetooth Wellness Devices

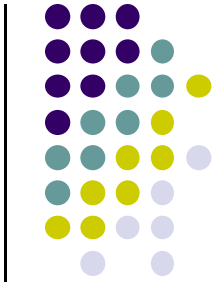
Explosion of Devices



- *Recent Nokia quote:* More cell phones than tooth brushes
- Many more sensors envisaged
- ***Ubiquitous computing:*** Many computers per person



Definitions: Portable, mobile & ubiquitous computing



- ***Distributed computing:*** system is physically distributed. User can access system/network from various points. E.g. Unix, WWW. (huge 70's revolution)

- ***Portable (nomadic) computing:*** user intermittently changes point of attachment, disrupts or shuts down network activities

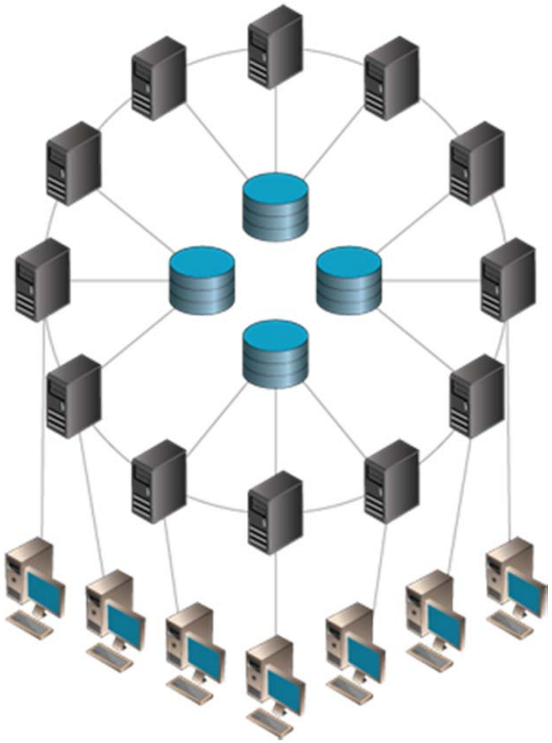
- ***Mobile computing:*** continuous access, automatic reconnection

- ***Ubiquitous (or pervasive) computing:*** computing environment including sensors, cameras and integrated active elements that cooperate to help user

- Class concerned mostly with mobile and ubiquitous computing

Distributed Computing

- ***Distributed computing example:*** You, logging in and web surfing from different terminals on campus (library, your dorm room, etc). Each web page consists of hypertext, pictures, movies anywhere on the internet.
- Note: network is fixed, Human moves



- Issues:
  - Remote communication (RPC),
  - Fault tolerance,
  - Availability (mirrored servers, etc)
  - Caching (for performance)
  - Distributed file systems (e.g. Network File System (NFS)
  - Security (Password control, authentication, encryption)

Portable (Nomadic) Computing

- ***Portable (nomadic) computing example:*** I own a laptop. Plugs into my home network, surf web while watching TV. In the morning, bring laptop to school, plug into WPI network, start up!
- Note: Network is fixed, device moves and changes point of attachment, no computing while moving

- Issues:
  - File/data pre-fetching
  - Caching (to simulate availability)
  - Update policies
  - Re-integration and consistency models
  - Operation queuing (e.g. emails while disconnected)
  - Resource discovery (closest printer while at home is not closest printer while at WPI)

Mobile Computing Example

- ***Mobile computing:*** John owns SPRINT PCS phone with web access, voice, SMS messaging. He runs apps like facebook and foursquare and remains connected while walking around Boston

- Note: Network topology changes, because sarah and mobile users move. Network deals with changing node location
- Issues
  - Mobile networking (mobile IP, TCP performance)
  - Mobile information access (bandwidth adaptive)
  - System-level energy savings (variable CPU speed, hard disk spin-down, voltage scaling)
  - Adaptive applications: (transcoding proxies, adaptive resource resource management)
  - Location sensing
  - Resource discovery (e.g. print to closest printer)

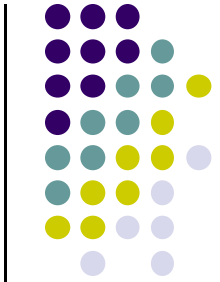
Ubiquitous Computing Example

- ***Ubiquitous computing:*** John is leaving home to go and meet his friends. While passing the fridge, the fridge sends a message to his shoe that milk is almost finished. When John is passing grocery store, shoe sends message to glasses which displays “BUY milk” message. John buys milk, goes home.
- **Core idea:** ubiquitous computing assistants **actively** help John
- Issues:
  - Sensor design (miniaturization, low cost)
  - Smart spaces
  - Invisibility (room million sensors, minimal user distraction)
  - Localized scalability (more distant, less communication)
  - Uneven conditioning
  - Context-awareness (assist user based on current situation)
  - Cyber-foraging (servers augment mobile device)
  - Self-configuring networks

Location-Aware App: Yelp

- **Example search:** Find Indian restaurant
- App checks user's location
- Indian restaurants **close to user's location** are returned

Context-Aware Search



- *[Hapori: Context-based Local Search for Mobile Phones using Community Behavioral Modeling and Similarity,* Nicholas D. Lane, Dartmouth College]

- Goal: Improves mobile search results using context, such as weather, age, profile of user, time, location and profile of other users to improve search.

- **Example:** a teenager gets a completely different set of recommendations from and elder.

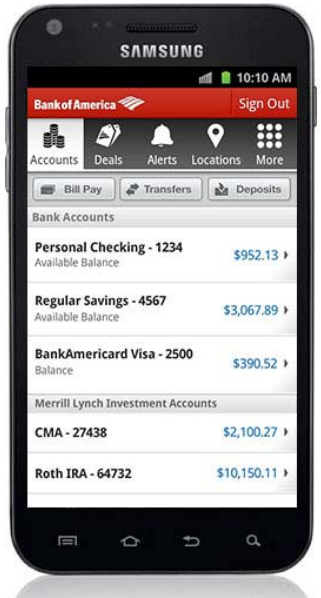
Location-Dependent App: Word Lens

- Translates signs in foreign Language

- Location-dependent because sign location varies

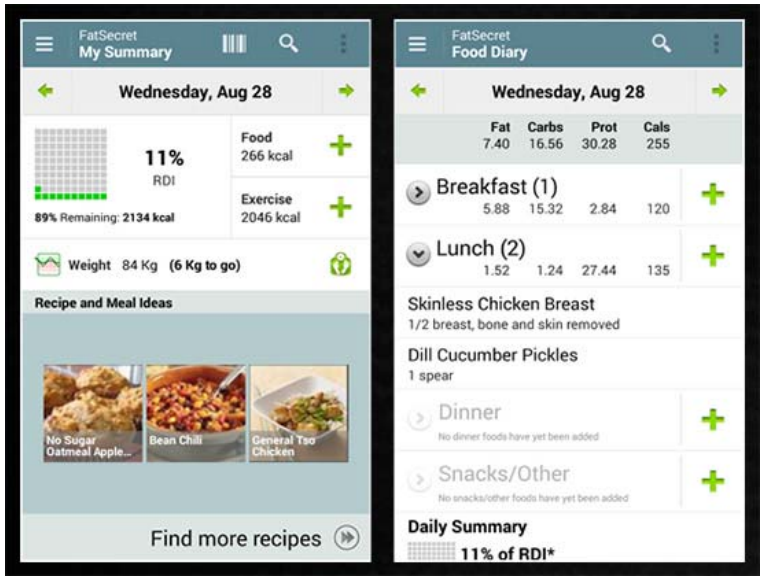
Desktop or Internet App on Mobile
NOT Really Mobile Computing

- Some apps run on mobile phone **just for convenience**
- No location-dependent, context-dependent inputs.
- Not really mobile computing apps
- **Examples:**



Mobile banking app

Internet Retailer app



Diet recording app

Energy Efficiency

- Most resources increasing exponentially *except* battery energy (ref. Starner, IEEE Pervasive Computing, Dec 2003)

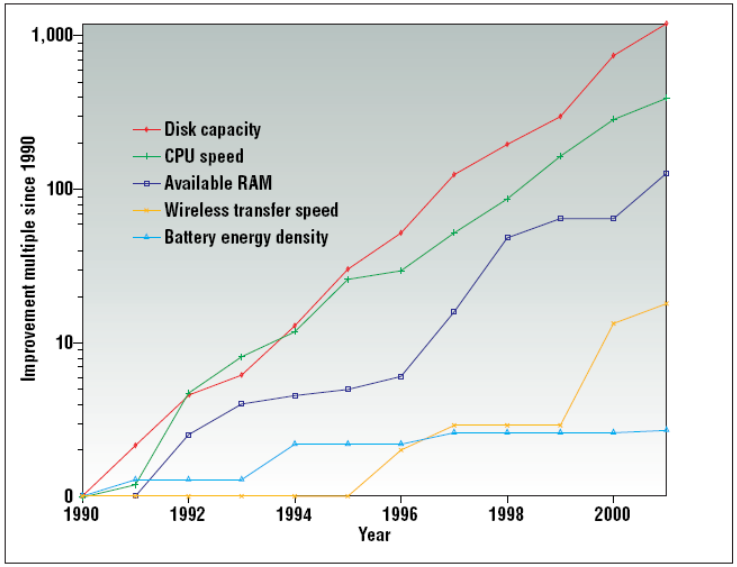
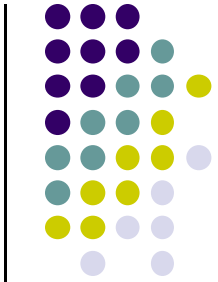


Figure 1. Improvements in laptop technology from 1990–2001.

- Some Strategies:
  - **Energy harvesting:** Energy from vibrations, moving humans
  - **Scale content:** Reduce image, video resolutions to save energy
  - **Better user interface:** Estimate and inform user how long each potential task will take
    - E.g: At current battery level, you can either type your paper for 45 mins, watch video for 20 mins, etc

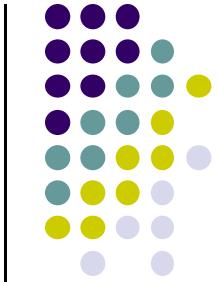
Mobile CrowdSensing



- **Internet of things:** Sensing data from consumer-centric devices including
  - Smartphones (iPhone, Google Nexus,)
  - Music players (iPods)
  - Sensor embedded gaming systems (Wii, Xbox, kinect)
  - In-vehicle sensors (GPS)
  - Body-worn sensors (e.g. fitbit, Nike+)

- **Mobile crowdsensing:** sense these devices
  - personal, community- and Internet-wide
- Sensing applications at community scale possible

Mobile CrowdSensing



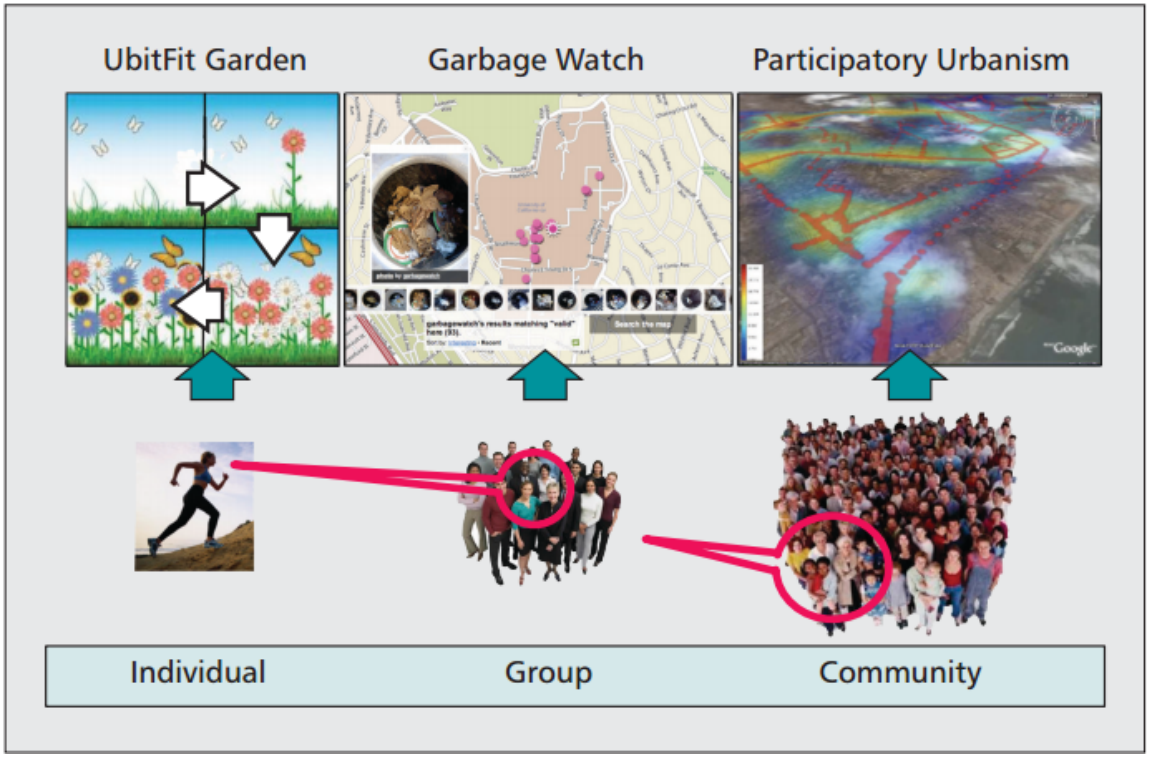
- **Personal sensing:** phenomena pertain to individual
  - E.g: activity detection and logging for health monitoring
- **Group:** friends, co-workers, neighborhood
  - GarbageWatch to improve recycling, neighborhood surveillance
- **Community sensing (mobile crowdsensing):**
  - Large-scale phenomena monitoring
  - Many people contribute their individual readings
  - **Examples:** Traffic congestion, air pollution, spread of disease, migration pattern of birds, city noise maps

Mobile CrowdSensing Types

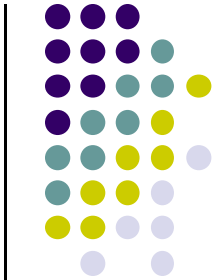
- **Participatory sensing: active** involvement of individuals (e.g taking a picture, reporting potholes)

- **Opportunistic sensing: passive** user involvement

(continuous location sampling without explicit user action)

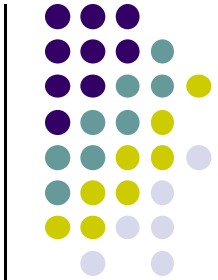


Mobile Crowd Sensing



- ***Classic example:*** Comparative shopping
- At CVS, ready to buy toothpaste. Is CVS price the best locally?
- Phone has software to query other members of my network
- People at other local stores (Walmart, Walgreens, etc) respond with prices

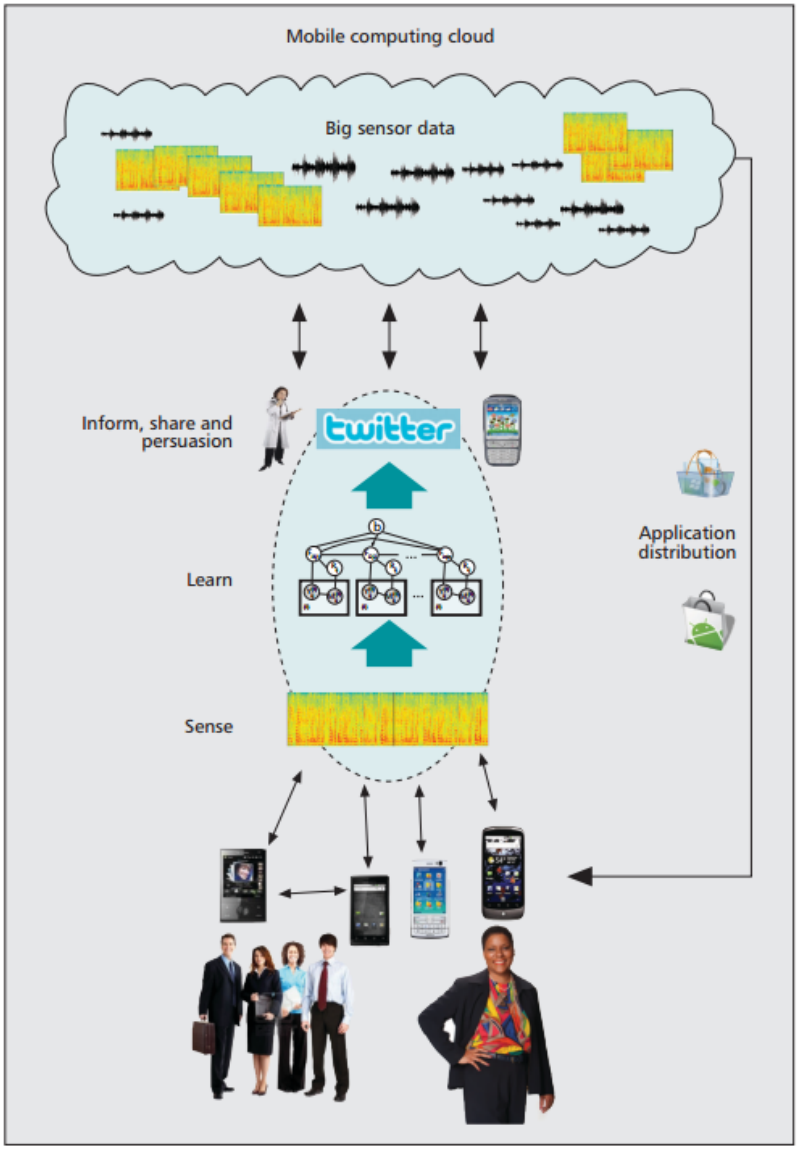
Sense What?



- **Environmental:** pollution, water levels in a creek
- **Transportation:** traffic/road conditions, available parking
- **City infrastructure:** malfunctioning hydrants and traffic signs
- **Social:** photoblogging, share bike route quality, petrol price watch
- **Health and well-being:**
  - Share exercise data (amount, frequency, schedule),
  - share eating habits and pictures of food

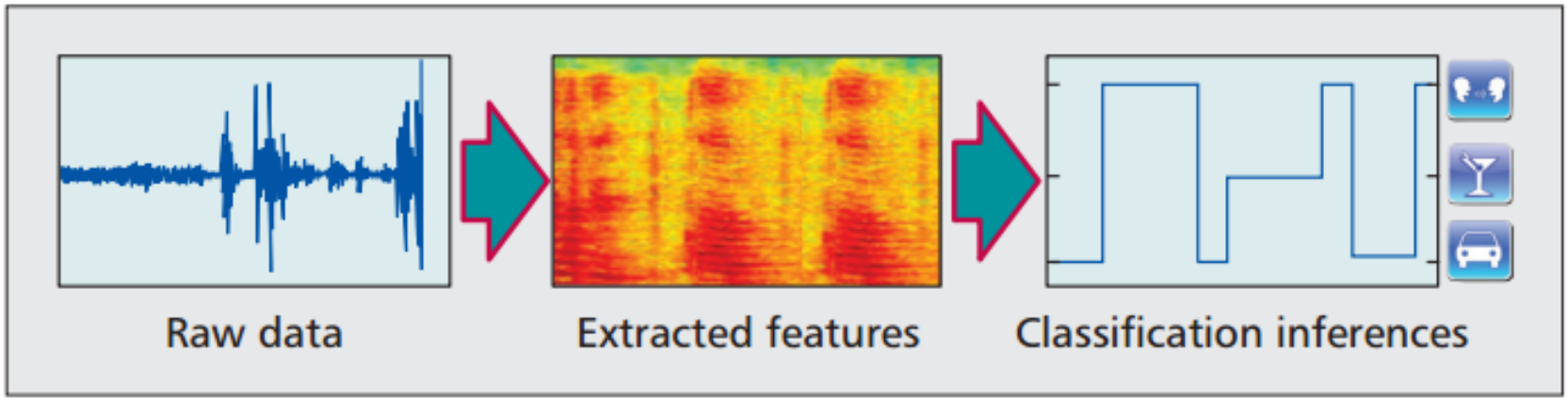
Mobile Phone Sensing Architecture

- **Sense:** Phones collect sensor data
- **Learn:** Information is extracted from sensor data by applying machine learning and data mining techniques
- **Inform, share and persuasion:** inform user of results, share with group/community or persuade them to change their behavior

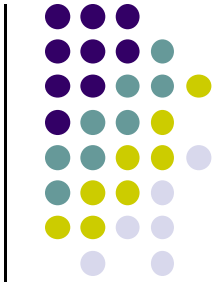


Sensor Processing

- **Machine learning** commonly used to process sensor data
  - Action to be inferred is hand-labelled to generate training data
  - Actual data is mined for combinations of sensor readings corresponding to action



Sensing Human Behavior



- *[Social Sensing for Epidimiological Behavior Change,* Anmol Madan et al, MIT Media Lab]

- **Goal:** infer how falling sick affects the [mobile/network] behaviors of human beings.
- **Examples:** Changes in call rates or visiting low entropy places more could mean person is sick
- Statistics of number of calls, co-location, proximity, WLAN and bluetooth entropy found to be good predictors of illness.
- Findings could be used as an early warning tool.
- If strong inference, then nurse could call the person
- **This work was basis for Venture funded company Ginger.io**

Mobile Computing: Measurement Studies

- How, when, where existing apps, mobile web, are being used
- Example: Where users engage in mobile commerce in UK