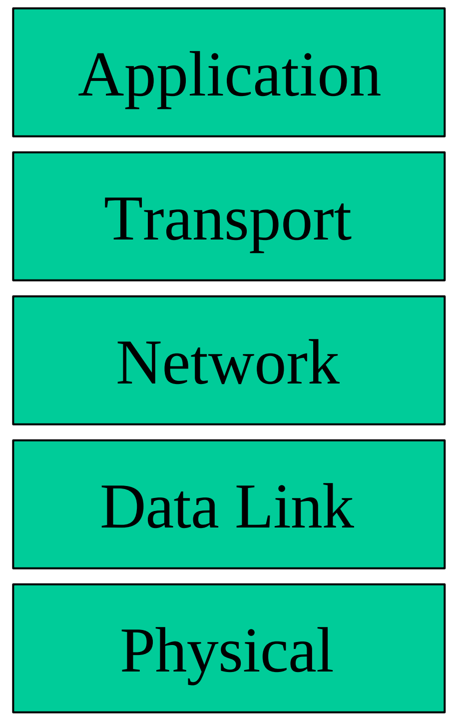
CS 525M – Mobile and Ubiquitous Computing Seminar

Emmanuel Agu

So Far..

- Last week:
  - Overview of course
  - Defined mobile, nomadic, ubiquitous computing and terms
  - Explained vision (Weiser's vision)
  - Sample of issues we will discuss
  - Most mobile/wireless issues due to
    - Mobile device: limited resources
    - Wireless channel: error, low BW
    - User: disconnection, mobility
- Adopted 5-layer networking model
- CS-approach: top down
- Today, start with mobile/wireless applications



Wireless Application

- Wireless/mobile applications:
  - Wireless messaging: SMS, etc
  - Wireless web: iMode, Wireless Access Protocol (WAP)
  - Experiences with application aware adaptation in Odyssey
  - MPEG-4
- Others:
  - Wireless graphics: Scalable Vector Graphics (SVG)

Wireless Messaging

- Quick word on wireless messaging:
  - Email is still killer application on the Internet
  - Instant messaging also very huge growth
  - Messaging available on certain wireless phones
- Short Messaging Service (SMS) was part of original GSM 2G cellular network in Europe
- Most 2G and 2.5G phones can send some form of SMS
- SMS is sometimes hooked up to AOL, MSN, Yahoo messenger
- Popularity of SMS led to other messaging standards:
  - CBS (broadcast messages)
  - USSD (connection-oriented, can reply immediately)
  - Enhanced or Smart messaging (fonts, concatenate msgs, etc)
  - Multimedia messaging (graphics, multimedia)

Wireless Web

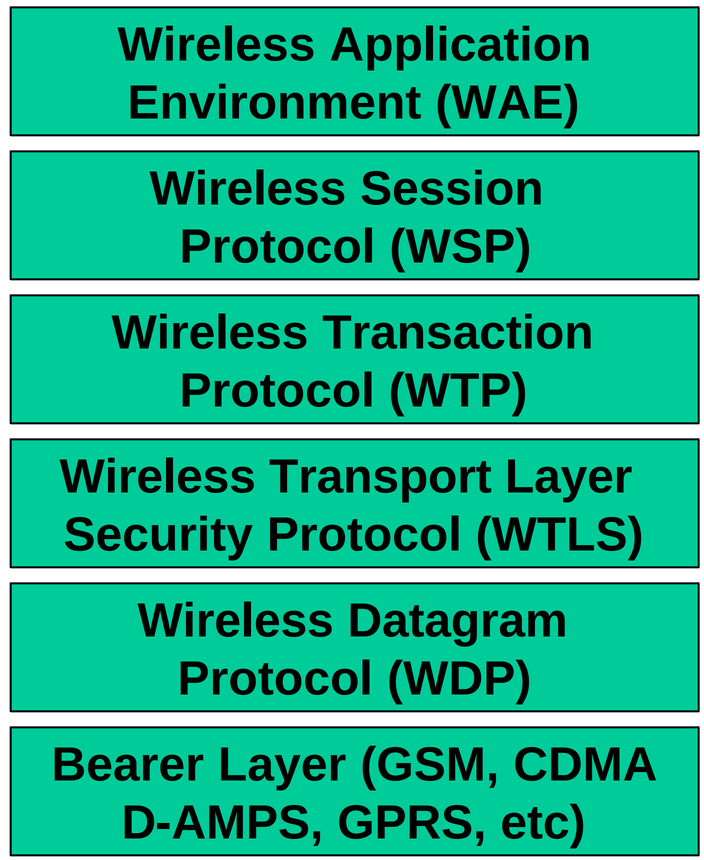
- Reference: *Computer Networks by Tanenbaum (4th edition)*
- Today's web model
  - You click on a page, HTML page and linked elements (images, are retrieved)
  - Page is retrieved in network packets (packet switched)
  - Success of web made people want to access it wirelessly
- Wireless Application Protocol (WAP) 1.0
  - Application protocol stack for wireless web
  - Standard proposed by consortium which included Nokia, Ericsson, Motorola, and Phone.com (previously Unwired planet)
  - WAP device may be mobile phone, PDA, notebook, etc
  - WAP optimized for mobile device (low CPU, memory, screen), low-bandwidth wireless links

WAP 1.0

- WAP 1.0
  - Brute force approach
    - Make phone call to web gateway
    - Send URL to gateway
    - If available, gateway returns page
  - Issues:
    - Connection-oriented (circuit-switched, per-minute billing), charged while reading web page
    - WAP pages written in Wireless Markup Language (WML) (major drawback: No HTML)
    - WML is XML-based
    - Sometimes a WAP filter (server) can automatically convert HTML pages to WML
  - Result: failed, but laid groundwork for iMode and WAP 2.0

WAP Protocol Stack

- Six layers (including actual wireless network)
- WDP is datagram protocol, similar to UDP
- WTLS is security layer, subset of Secure Socket Layer by Netscape
- WTP is similar to TCP, concerned with requests responses
- WSP is similar to HTTP/1.1
- WAE is microbrowser



I-Mode

- Sometimes in telecom, single organization or person beats consortium E.g. Jon Postel developed RFCs for TCP, SMTP, etc
- In parallel to WAP effort, Japanese woman Mari Matsunaga created different approach called I-Mode (Information Mode)
- Mari convinced Japanese telco monopoly, NTT DoCoMo to deploy service
- I-Mode deployed in Feb. 1999
- I-Mode subscription exploded!!
- 35 million Japanese subscribers in 3 years, access to 40,000 I-Mode pages
- Major financial success!
- Interesting case study: features, why it succeeded?

I-Mode

- To make I-Mode work, 3 new components:
  - New transmission system (partnership with Fujitsu)
  - New handset (partnered with NEC, Matsushita)
  - New web page language (cHTML)
- Transmission system:
  - 2 separate networks:
  - Voice mode:
    - old 2G digital phone network, PDC
    - (circuit-switched),
    - billed per connected minute
  - I-Mode:
    - New packet-switched network for I-Mode, always on
    - Internet connection, users unaware of this!
    - No connection charge, billed per packet sent
    - Uses CDMA, 128-byte packets at 9600 bps
  - Both networks cannot be used simultaneously

I-Mode

- I-Mode handsets:
  - Enhanced features with CPU power of PC in 1995
  - small screen
  - IP-capable communications
- Handset specifications
  - 100 MHz CPU
  - Memory: Several MB flash memory, 1MB RAM
  - Dimensions: smaller than pack of cigarettes, 70 grams
  - Screen:
    - Resolution: min. 72 x 94 pixels, 120 x 160 high end
    - Color: 256 colors initially, good for line drawings, cartoons, no photographs. New: 65,000 colors
  - Navigation: no mouse, use arrow keys, “i” key takes you to I-mode services menu

I-Mode

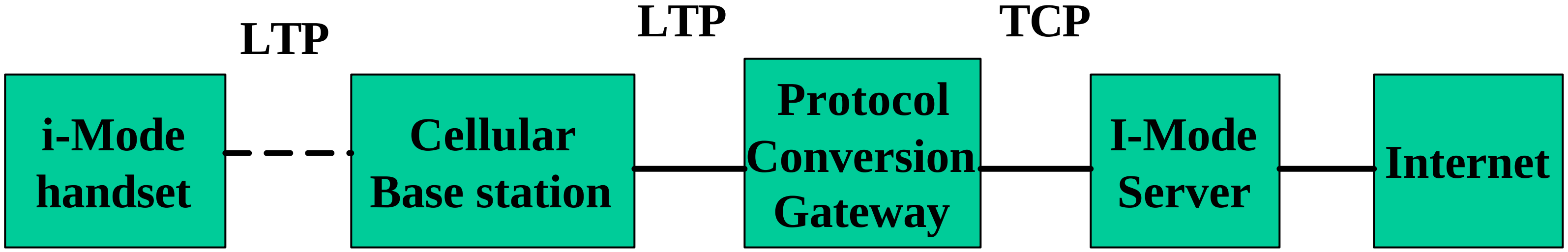
- I-Mode handsets:
  - When user hits “i” key on handset, user presented with list of Categories: email, news, weather, sports, etc (a portal)
  - over 1000 “services” in about 20 categories
  - Lots of services targetted at teenagers, young people
  - Each service is I-Mode website run by independent company
  - May type in service URL directly also
  - Users subscribe to services ($1-$2 per service)
  - > 1,000,000 subscriber makes service official
  - Official services billed through phone bill
  - 1500 official services, 39,000 unofficial circa 2001

I-Mode

- I-Mode handsets:
  - Most popular application is email: limit of 500 bytes (SMS on GSM limit is 160 bytes)
  - I-Mode phone number doubles as email address (e.g. 0345671234@docomo.co.jp)
  - Rich in graphics content, Japanese have high visual sensibility
  - Invented new cute pictograms like smileys called **emoji**
  - US company, Funmail has patented text-to-graphics. E.g. word Hawaii in email may be converted to animated cartoon image of *“beach with swaying palm trees”*
  - Funmail is multi-platform technology:
    - cell phones receive animations scaled for power, screen size.
    - Desktops receive full-blown animation

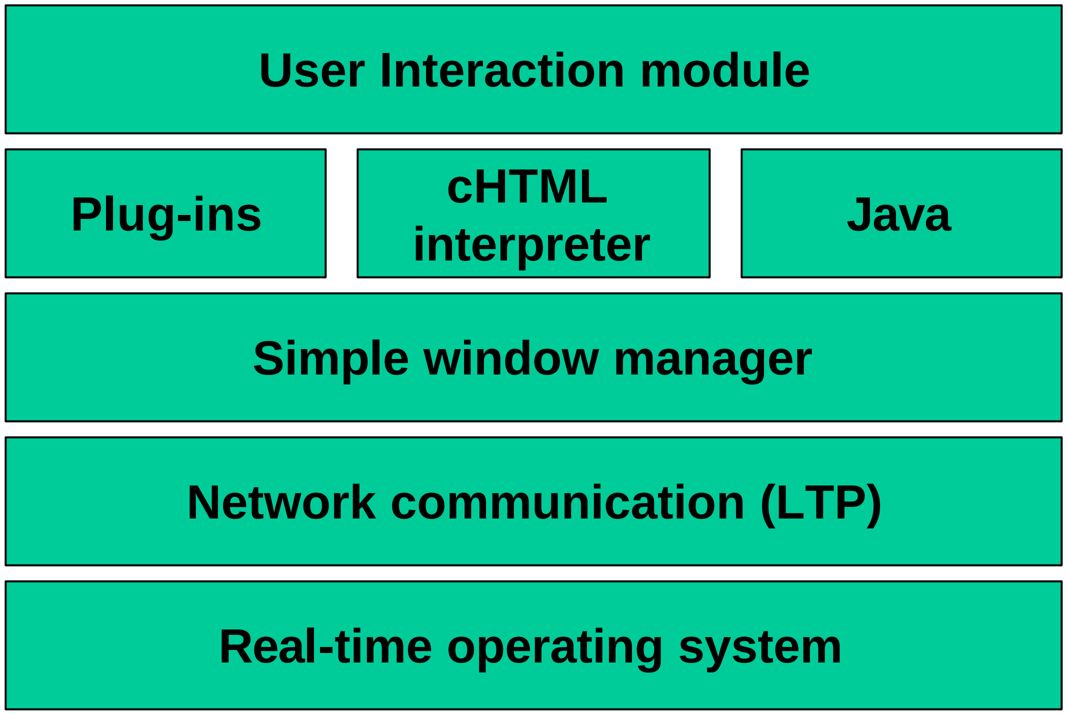
I-Mode

- I-Mode is massive success in Japan because:
  - Few people own PCs
  - Local phone access is expensive
  - Lots of time spent commuting
- Different circumstances for US and Europe
- I-Mode structure and operation:
- Handsets speak Lightweight Transport Protocol (LTP) over wireless link to protocol conversion gateway
- Gateway converts request to TCP request
- Gateway has fiber-optic connection to I-Mode server
- I-Mode server caches most pages for performance



I-Mode

- I-Mode protocol stack:



- I-Mode pages programmed in cHTML
- Java functionality based on J2ME (Java 2 Platform Micro Edition) based on the Kilobyte Virtual Machine (KVM)
- Maximum of 5 applets can be stored at a time

I-Mode

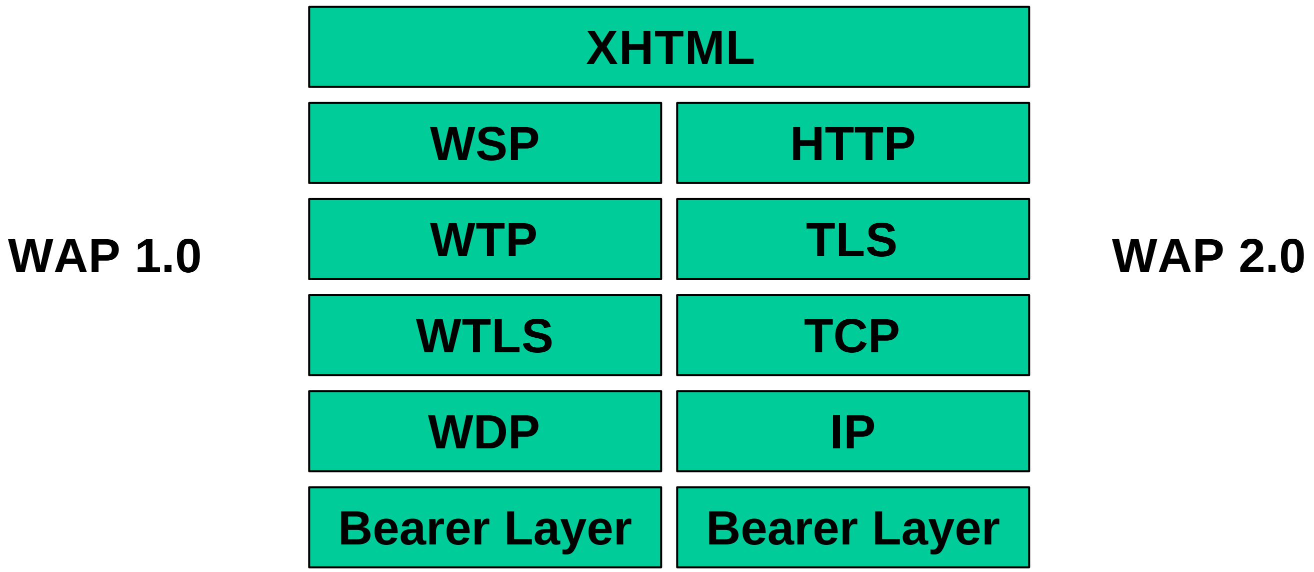
- cHTML
  - Developed by Access, embedded software maker
  - based on HTTP 1.0, with omissions and extensions
  - Most HTML tags allowed. E.g. <body>, <ul>,
, etc
  - New tag to dial phone number, phoneto
  - E.g. phoneto on a restaurant's page lets you dial number
  - HTML-based: can view I-Mode pages on regular browser
- I-Mode Browser:
  - Limited
  - Allows plug-ins and helper applications e.g. JVM
  - No Javascript support, frames, background colors/images, JPEG (takes too long)
- I-Mode Server-side:
  - Full-blown computer, all bells and whistles
  - Supports CGI, Perl, PHP, JSP, ASP, most web standards

WAP 2.0

- Goal: fix WAP 1.0 shortcomings
- Features:
  - Push model as well as pull
  - Integrated telephony (voice and data) into applications
  - Multimedia messaging
  - Include 264 pictograms (emoji)
  - Interface to storage device (e.g. flash memory)
  - Support for browser plug-ins (also new scripting language, WMLScript)

WAP 2.0

- New protocol stack based on TCP and HTTP/1.1
- Modified TCP (compatible with original)
  - Fixed 64KB window
  - No slow start
  - Maximum 1500-byte packet
  - Slightly different transmission algorithm
- WAP 2.0 supports new and old (WAP 1.0) protocol stack



WAP 2.0

- WAP 2.0 supports XHTML basic
- NTT DoCoMo has agreed to support XHTML so that pages will be widely compatible
- Hopefully, this will end format wars
- XHTML targetted at low end devices (mobile phones, TVs, PDAs, vending machines, pagers, watches, etc)
- Thus, no style sheets, scripts or frames
- WAP 2.0 speed 384 kbps
- WAP threat:
  - 802.11b (11Mbps) and 802.11g (54Mbps) can download regular web pages, becoming available in coffee shops
  - People will prefer 802.11 where available
- Hybrid solution: dual mode devices that use 802.11 where available and WAP otherwise

Application-Aware Adaptation

- Background:
  - Satyanarayanan and group at CMU have been leaders in mobile/ubiquitous computing field for over 10 years
  - Major work on Coda file system (covered later), odyssey and now project Aura (last Friday's talk)
- Application-aware adaptation:
  - System resources like memory, network bandwidth, etc vary unpredictably
  - Client needs to adapt
  - Each application has different ways to adapt
  - Let application determine its preferred way to adapt
  - Important in environment with concurrent applications running since resource requirement/usage is interdependent

Application-Aware Adaptation

- When faced with scarce resources, mobile clients can react in two ways:
- **Option A:** Reduce use of scarce resource, increase use of abundant resource. E.g.
  - Lossless compression reduces bandwidth use, increases computation
  - Caching reduces bandwidth due to misses, increases computation in replacement policy, etc
- Option A can cope with small swings in availability.
- E.g. mobile client bandwidth may vary by orders of magnitude (recall av. error rate is 10^{-3})
- **Option B:** trade application quality for resource consumption (used in Odyssey). Eg. If bandwidth drops,
  - Use video stream with fewer colors
  - Web browser fetches highly compressed images

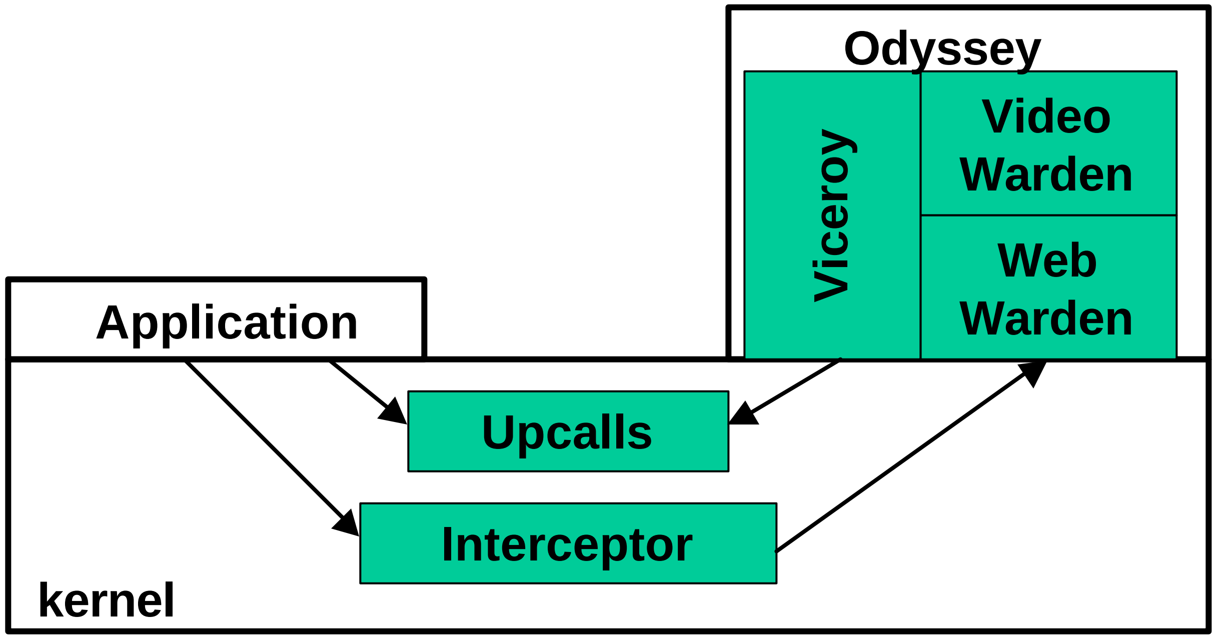
Application-Aware Adaptation

- Odyssey uses option B
- Define new notion of **fidelity** to quantify quality
- Every data item has a most detailed copy **reference copy**
- Mobile user ideally uses reference copy
- If resources get scarce, degrade reference copy in some way
- **Fidelity** defines how much degraded copy varies from reference copy
- Fidelity is data type-specific E.g.
  - Video may be degraded by dropping frames
  - Maps may be degraded by removing features such as buildings and leaving roads and rivers

Application-Aware Adaptation

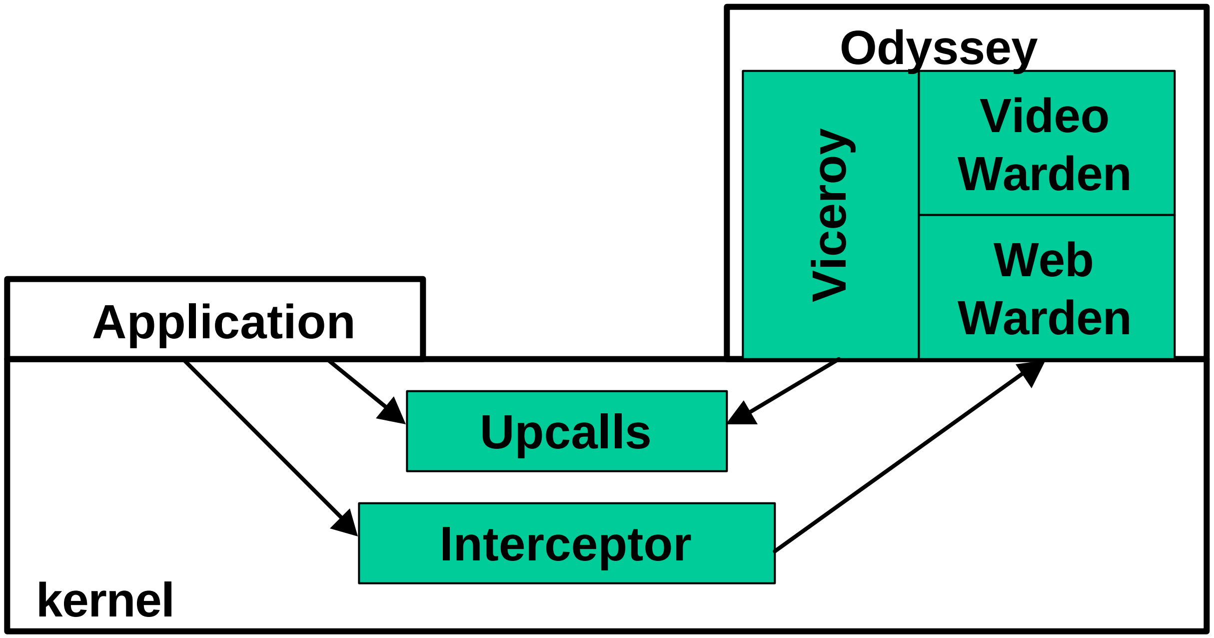
- Models for adaptation:
  - **Laisez-faire:**
    - Each application adapts separately,
    - Concurrent applications may lead to conflicts
  - **Application-transparent:**
    - OS does all adaptation
    - Legacy applications run well
    - Problems with diverse applications
- Collaboration between application and OS is called **application-aware adaptation**
- Odyssey is platform for mobile data access, incorporates application-aware adaptation

Odyssey Architecture



- Interceptor provides VFS client which forwards file system requests to the **viceroy**
- **Viceroy** monitors resource availability, manages their use
- Wardens provide application-specific fidelities that applications can pick

API



- Two new calls:
  - **Resource request:** used by application to inform Odyssey which resources it is interested in E.g video -> bandwidth
  - **Type-specific operation:** used by application to change data fidelity. E.g video may reduce frame rate if BW lower
- Request API also declares **window of tolerance**, no reaction within window
- Resources: bandwidth, latency, disk space, CPU, battery power

Odyssey Operation

- Each application declares resources to monitor, window of tolerance
- Viceroy records these values
- When resource changes, check recorded table to see if window of tolerance is exceeded
- When window is exceeded, viceroy sends upcall to application to inform it of changes
- Application reacts by changing fidelity of accessed data
- Fidelity changes are carried out by wardens

Experiments

- Principal resource managed in paper was network bandwidth
  - Volatile resource
  - Orders of magnitude changes
- Bandwidth estimation at the transport layer
- Timestamp and measure round trip times
- Use as estimate to predict near-term future bandwidth use
- Samples may be choppy, use simple linear filter to smoothen
- Viceroy divides bandwidth based on following algorithm:
  - Application which currently use lots of bandwidth will continue to need lots of bandwidth
  - Reserve small portion so no total starvation of applications that have been dormant for prolonged periods

Example Applications

- Instrumented 3 applications to run on Odyssey:
  - Video player: Xanim
  - Web browser: Netscape
  - Speech recognizer: Janus
- These applications are:
  - relatively rich, can take some degradation
  - Implement application-specific warden
- Represent data at various fidelity levels, 1 for reference copy
- Key questions in experiments:
  - Effort required to modify application to use Odyssey
  - Can Odyssey support multiple diverse applications concurrently?
  - Is source code essential? Are binaries sufficient?

Video Player: XAnim

- Xanim video player had sources available
- Used QuickTime video format
- Reads requested file from disk, plays it back, skips late frames
- Integration with Odyssey: split functionality into client, warden server
- Pre-encode movies into multiple versions or tracks (fidelities)
- Meta-data also specifies for each track, sizes and offsets of each frame in track
- Implemented 3 tracks per movie
  - Color JPEG at 99 quality (fidelity = 1.0)
  - Color JPEG at 50 quality (fidelity = 0.5)
  - Black-and-white (fidelity = 0.01)
- No interframe compression, 10 Frame per second

Video Player: XAnim

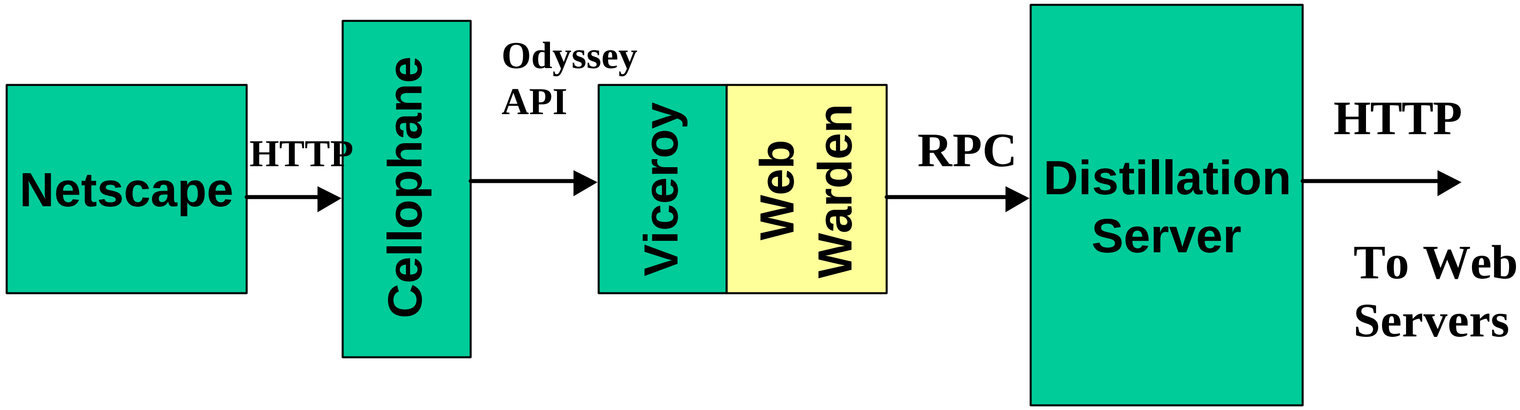
- Late frames are not shown, simply dropped
- Client's performance metric: number of late frames it skips
- User perception: consecutive drops worse than intermittent drops
- Client adaptation policy:
  - Estimate bandwidth required for each fidelity level
  - Play best quality track (fidelity) without dropping frames
- If client is notified of changing bandwidth, changes fidelity level

Web Browser: Netscape

- Integrated Netscape browser with Odyssey
- Netscape was shrink-wrapped, no source code available then
- To work around lack of source code, use Netscape's proxy facility
- Proxy facility routes HTTP traffic through a designated host
- Placed proxy, called **cellophane** on client between Netscape and Odyssey
- Cellophane routes all netscape requests through file system
- Odyssey views cellophane as adaptive application

Web Browser: Netscape

- Web warden forwards all cellophane requests to a remote **distillation server**
- Distillation server connected to rest of web and can fetch HTML pages, images
- Distiller can degrade images on the fly using JPEG compression to reduce transmission time
- Distiller focusses on images for 2 reasons:
  - Bandwidth hungry
  - Natural compression mechanism (JPEG compression)



Web Browser: Netscape

- Distiller degrades images above threshold of 2K in size
- Assign fidelity levels to:
  - Original image (fidelity = 1.0)
  - 3 degraded versions (fidelity = 0.5, 0.25, 0.05)
- Adaptation policy:
  - Calculate *y* = 2 x download time of original on 10 Mbps Ethernet
  - As bandwidth gets lower, fetch image fidelity that takes *y* time to download
  - Heuristic based on fact that user will wait *y* time for image

Speech Recognier: Janus

- Speech recognition:
  - **Potential:** leaves mobile users hands free for other activities
  - **Challenge:** requires high accuracy, mobile environments can be noisy
- **Janus** was freely available speech recognizer
- Today: Dragon dictate is main package?
- Janus
  - **Input:** raw sampled speech utterance from microphone
  - **Output:** ASCII representation of utterance
- Above conversion is very expensive in both CPU cycles and virtual memory
- Mobile client is resource-constrained: offload conversion where possible

Speech Recognizer: Janus

- 2-phase voice recognition process:
  - **Vector quantization:** transforms raw speech into compact format
  - Remainder of recognition process
- Initial Janus setup:
  - Uttered speech to an Odyssey speech object begins recognition
  - Reading from speech object returns recognized ASCII text
- Integration to Odyssey:
  - Write simple speech front end which collects raw speech utterance, writes it to speech object and reads results
  - Two speech servers (local and remote) can be used by warden

Speech Recognizer: Janus

- Speech warden has 3 alternatives
  - Use local recognition server
  - Use remote recognition server
  - Hybrid: use local server to compact (quantize) and send smaller content to remote server
- Fidelity metric:
  - Remote or hybrid: use the best recognition possible (fidelity = 1.0), allows other applications to run
  - Local: use smaller acoustical model, vocabulary and grammar (fidelity = 0.5)
- Performance metric: latency (time to recognize utterance)
- Janus application estimates bandwidth and does remote if bandwidth is sufficient, else execute locally

Lessons Learned

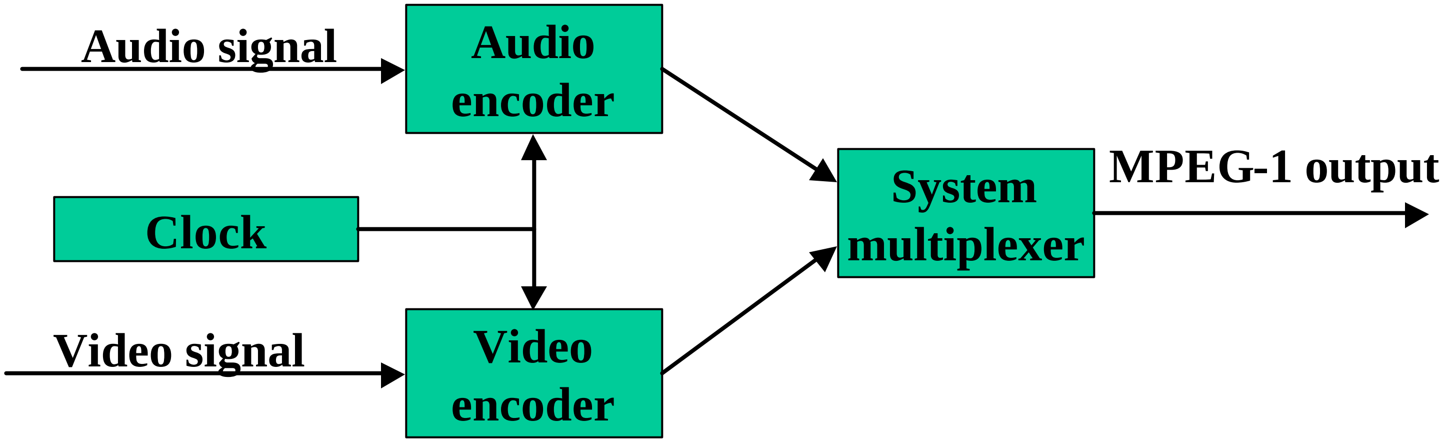
- Porting applications was easy since:
  - Adaptive code is outside body of application limiting scope of required modifications
  - Chosen applications already capable of decoding multiple representations of a data type (fidelity levels)
- Shrink-wrapped applications:
  - were adaptable especially if they already had mechanisms such as proxies
  - Could also use a run-time library which re-routes calls
- Reduced burden on applications by division of duties:
  - OS manages resource usage for all machine,
  - application decides its own goals and adaptation policy
- Balancing agility and stability: adapt quickly, but not to the point where impact on user is disconcerting! (maybe feedback to user required)

Mobile MPEG

- MPEG (Motion Picture Experts Group) standard for compressing video files since 1993
- Movies contain sound: MPEG can compress both audio and video
- Different generations of MPEG
- MPEG-1:
  - Goal: video-recorder quality (352 x 240 for NTSC) using a bit rate of 1.2Mbps
  - Uncompressed at 24 bits per pixel requires 50.7 Mbps
  - Compression ratio of 40 required to reduce to 1.2 Mbps
- Notes:
  - NTSC is video standard in US
  - PAL is standard in Europe

Mobile MPEG

- MPEG-2:
  - designed for compressing broadcast-quality video into 4-6 Mbps (to fit into NTSC and PAL broadcast)
  - Also forms basis for DVD and digital satellite TV
- MPEG-1 and 2 are similar: MPEG-2 almost superset of MPEG-1
- MPEG-1: audio and video streams encoded separately, uses same 90-KHz clock for synchronization purposes



Mobile MPEG

- Compression techniques usually take out redundancies
- MPEG compresses using **spatial** and **temporal** redundancies in movies
- Think of streaming movie as sequence of still (JPEG) images
- Spatial coherency is redundancy within 1 still image (each JPEG)
- Temporal redundancy
  - exploits the fact that consecutive frames are almost identical
  - reduced in new scenes in a movie, etc
  - Increased for slow-moving objects, stationary camera/background
- Every run of 75 similar concurrent frames can be compressed

Mobile MPEG

- MPEG-1 output consists of four kinds of frames:
  - **I (Intracoded)** frames:
    - self-contained JPEG-encoded still pictures
    - Act as reference, in case packets have errors, are lost or stream fast forwarded, etc
  - **P (Predictive)** frames:
    - Block-by-block difference with last frame
    - Encodes differences between this block and last frame
  - **B (Bi-directional)** frames:
    - Difference between the last or next frame
    - Similar to P frames, but can use either previous or next frame as reference
  - **D (DC-coded)** frames:
    - Encodes average values of entire block
    - Allows low-res image to be displayed on fast-forward

Mobile MPEG

- MPEG-2:
  - I, P, B frames supported
  - D frames NOT supported
  - Supports both progressive and interlaced images
  - Encodes smaller blocks to improve output
  - Also supports multiple resolutions

Mobile MPEG

- Mobile multimedia applications are either **indoor** or **outdoor**
- Indoor applications have low mobility, high bandwidth (e.g. on WPI wireless LAN)
- Outdoor applications have higher mobility, low bandwidth (e.g. on Sprint PCS cellular network)
- Conflict:
  - Low bandwidths argue for more efficient encoding/compression, less redundancy
  - High wireless error rates argue for more redundancy to recover
- Conclusion: be careful with what redundancy you take out

Mobile MPEG

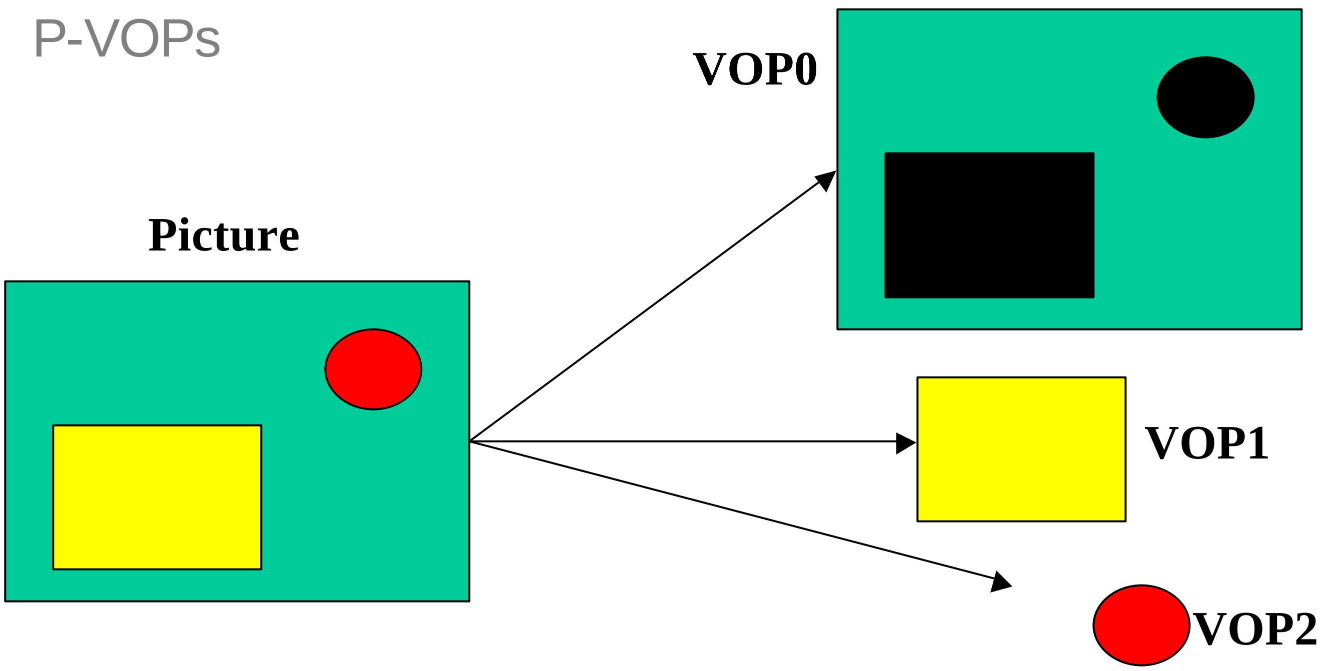
- MPEG-4:
  - In addition to previous audio, video encoding and multiplexing, also has
    - coding of text/graphics and synthetic images
    - Representation of audio-visual scene and composition
  - Has some wireless features
  - New features considered important included robustness to errors and coding efficiency
  - Example applications:
    - Internet and Intranet video
    - Wireless video
    - Video databases
    - Interactive home shopping
    - Video e-mail, home movies
    - Virtual reality games, simulation and training

Mobile MPEG

- MPEG-4 specific wireless-friendly standards requirements:
  - **Universal access:** *"Robustness in error prone environments: The capability to allow robust access to applications over a variety of wireless and wired networks and storage media. Sufficient robustness is required, especially for low bit-rate applications under severe error conditions"*
  - **Compression:** *"Improved coding efficiency: The ability to provide subjectively better audio-visual quality at bit-rates compared to existing or emerging coding standards"*
- Formal tests to verify these requirements with:
  - high random Bit Error Rate (BER) of 10^{-3}
  - Multiple burst errors

MPEG-4 Video Basics

- Input video sequence = series of related snapshots/pictures
- Elements of a picture = Video Object (VO)
- Video Objects are changed by translations, rotations, scaling, brightness, color, etc
- Several MPEG-4 functions access these VOs not pictures
- Video Object Planes (VOPs) described by texture variations
- Similar to I, B and P frames, there are I-VOPs, B-VOPs and P-VOPs



MPEG-4

- Other features such as:
  - sprite coding for games,
  - scalable video coding for variable video quality
  - robust video coding
- Robust video coding including:
  - Object priorities: lost low priority objects have little effect
  - Resynchronization: errors don't accumulate
  - Data partitioning:
  - Reversible VLCs
  - Intra update and scalable coding
  - Correction and concealment strategies (not specified due to channel-specific nature). E.g. addition of FEC bits

Projects

- Term long project
- May team up in groups or alone
- Hopefully, you will work on things you enjoy, good at
- Need to decide:
  - Top 3 areas you may like to explore (deadline Feb. 10)
  - Your strengths/weaknesses are
  - Nature of research you like doing
    - Mathematical
    - Algorithmic
    - Experimental/measurement
    - Simulation
    - System design and development

Projects

- 4 Project deadlines:

Description	Deadline
Decide project area:	February 10
Propose project:	March 16
Mid-project update:	April 6
Present results	April 27

- **Note:** March 6: no class (term break)

Presentations

- I will talk for 20 minutes in all lectures followed by 3 paper presentations
- 30-minute talk based on selected paper
- Extra 10-minutes for discussions/questions (during and after)
- Make sure you understand the paper
- Select:
  - Aspects to present/omit
  - Supplemental material to add to improve understanding
- Rough talk outline (of research paper):
  - Introduce problem/give overview
  - Explain main proposed solutions
  - Other improvements
  - Future work

Presentations

- This powerpoint template is on website. Please use for uniformity!!
- Note: send me your powerpoint slides latest noon on the day of your talk, so that I can put it on website
- If you are unsure of how to use your 30 minutes, you can ask me. E.g. if paper looks long
- You can send me outlines, rough drafts of slides, etc.