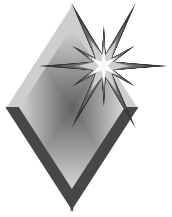
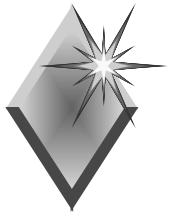


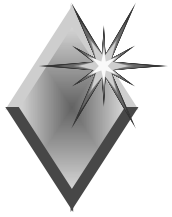
Course Overview

- Introduction and History
- Regulatory Issues
- Fundamentals of Cellular Systems
- AMPS and CDPD (Cellular Digital Packet Data)
- GSM (Global System for Mobile communication)
- Mobile IP
- TCP over Wireless Link
- Issues in Mobile Application Design
- Outlook and Conclusions



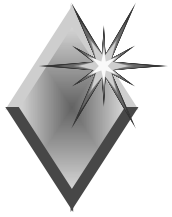
What Exactly does “Mobile” mean?

- user mobility vs. computer mobility
  - user mobility: always work in same computing environment, problem is how to “teleport” environment to location of user
  - computer mobility: computing environment moves with user, utilizes wireless links, has to deal with changing environments
- scales of mobility:
  - within house/building
  - campus/town/city
  - state or region
  - country or continent
  - world
- we will look at potentially world-wide computer mobility



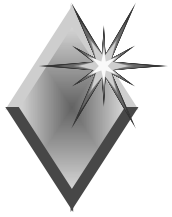
Challenges in Mobile Computing

- wireless communication is substantially different from wired communication:
  - frequent spurious disconnections (handoff, shadowed areas)
  - lower bandwidth (GSM, CDPD: less than 20 kbps)
  - high bandwidth variability (roaming outdoors vs. plugged into docking station)
  - heterogeneous networks (outdoors: cellular, indoors: infrared LAN)
  - security risks (everyone can listen in on wireless channel)
- mobility:
  - address migration (e.g., location management)
  - location dependent information (where is closest printer?)
  - migrating locality (optimal connection changes over time)



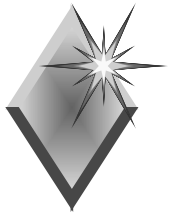
Challenges in Mobile Computing

- portability (of computing device):
  - limited power supply (battery)
  - data security (someone steals your laptop)
  - small user interface (look at screen size of a PDA)
  - limited storage capacity (disks, for example, are a “power liability”)
- solutions (or suggestions) exist to address all these issues, however, the solutions are often conflicting:
  - do not keep data on portable device to reduce security problem
  - do not send data if not necessary to save valuable energy



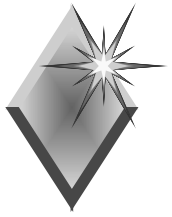
Solutions and Research Directions

- solutions to overcome hardware disparities:
  - sophisticated power management (sleep modes, disk mgmt, clock speed, prefer receiving data over sending: “ether as cache”)
  - delegate tasks to stationary computers (application partitioning, stationary computer has disk space, high bandwidth connection, powerful CPU)
  - trade increased data processing for reduced network bandwidth requirements (on-line compression, difference-based updates, filtering, less restrictive consistency semantics)
  - reduce network latency (data prefetching, batch multiple messages)



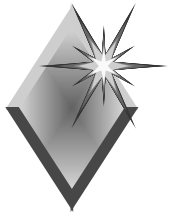
Solutions and Research Directions

- solutions addressing mobility:
  - flexible directory services to enable mobile to find location-dependent services, often based on “shopping” or negotiation:
    - scaleable, fault-tolerant architecture (exploit communication locality, replicate directory data, weakened consistency semantics: allow directory info to go stale as long as we can improve on suboptimal routes during connection)
    - brokerage service (e.g.: trading in ODP)
    - accounting service: mediate cost accounting between mobile client and servers
    - rebindable service interfaces (switch service providers “on-line”)
  - ubiquitous security and authentication (IPv6 will go a long way towards providing this goal)



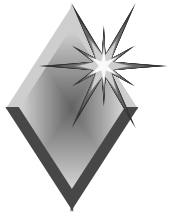
Solutions and Research Directions

- there seems to be general agreement that some sort of application partitioning is a good thing:
  - reduce bandwidth requirements
  - overcome the physical limitations of the mobile computer
- one open question: how do you partition and when?
  - static partitioning is easier (and we will see examples for WWW later)
  - however, computation environment can change radically during execution:
    - MH moves to area with better networking infrastructure (maybe gets back into his docking station)
    - PCMCIA cards all of a sudden add more memory
    - radio signal is dropped completely temporarily
  - implies growing interest in dynamic application partitioning



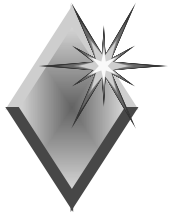
A Formal Model

- one way to examine issues in designing algorithms for a mobile environment is to develop and use a formal model (Badrinath et al.):
  - network consists of MH and MSS (Mobile Support Stations)
  - MSS covers a cell, uses broadcast within cell
  - MSS connected via a wired backbone
  - to communicate between two mobile hosts MH1 and MH2, message is routed through the corresponding MSSs
  - MSS may have to determine location of serving MSS for outgoing call through broadcast
  - MH are mobile, no time bound between leaving a cell and entering the next
  - FIFO MH-MSS channel within cell. MH typically receives only prefix of the messages sent (might have left cell)



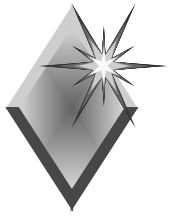
A Formal Model

- some consequences from model:
  - communication costs are no longer uniform:
    - relatively cheap to communicate between two MSS
    - relatively expensive to communicate MSS-MH or MH1-MH2
  - mobile hosts may disconnect (or doze off), reappear later
  - underlying network structure changes all the time
- model applied to study a well-known distributed mutual exclusion algorithm (Lamport's Algorithm):
  - if original algorithm implemented directly, poor performance:
    - high search costs for all MH-MH messages
    - all hosts required to participate, so disconnections not allowed
    - data structures need to be maintained at the mobile hosts, so high power usage



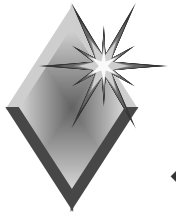
A Formal Model

- redesign of Lamport's Mutual Exclusion Algorithm:
  - two-tier structure: most of the coordination between MSS only
  - if MH wants to enter critical section, it contacts local MSS
  - local MSS obtains permission to enter critical section, which it passes on to requesting mobile
  - if requesting mobile leaves cell, update status during handoff (the model provides for "leave" messages when stations disconnect)
  - result: most communication limited to fast backbone, mobile stations are allowed to disconnect and to doze off
- using two-tier strategy is basically similar to "application partitioning"



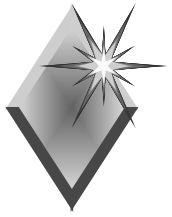
Typical Applications

- you most likely will **not** use your PDA to solve the *grand challenge* problems (after all, this is what MPPs and supercomputers are there for)
- unifying theme: mobile computer is interface to information anywhere at any time:
  - "access ramp" to NII
  - AOAC (always on, always connected)
- at least four main classes of applications
  - information browsing
  - personal communication
  - multiperson interaction
  - data entry



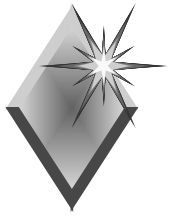
Typical Applications

- information browsing:
  - querying traditional databases
  - retrieving information from electronic books, magazines, newspapers (Newsnet News)
  - navigating through hypertext and other interactive documents (Gopher, Wais, WWW)
- personal communication:
  - e-mail and fax
  - key: from any location, at any time: asynchronously
- multiperson interaction:
  - groupware applications, electronic whiteboard (Bell commercial)
- data entry:
  - on-site entry: car rental agencies, warehouses, archaeology



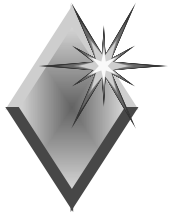
WWW: The Canonical Example

- one of the most-studied example application for mobile computing is WWW browsing
- there is a great variety of architectural proposals
- common theme: overcome bandwidth and physical limitations by statically splitting WWW browser into two pieces:
  - user interface, executing on MH (PDA, laptop)
  - a proxy, executing on base station or close to the base station
- concrete way of splitting application differs widely, depending on characteristics of mobile computer



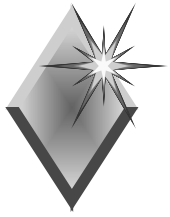
WWW: The Canonical Example

- example 1: W4
  - WWW access from a Newton PDA
  - proxy does all interaction with WWW servers, computes a screen image and sends the image to PDA
  - PDA caches and prefetches screen images and interacts with user
- example 2: Dynamic Documents
  - proxy retrieves documents from WWW server
  - instead of sending document to browser, send a “script” which dynamically generates document at mobile
    - idea: program describing page is lot smaller than page data
  - MH runs a modified version of Mosaic - relatively powerful machine



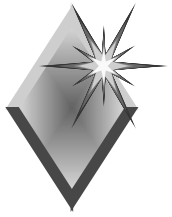
WWW: The Canonical Example

- example 3: TeleWeb
  - proxy is caching proxy between browser and server
  - proxy caches documents locally
  - GET requests routed through proxy, which checks current connectivity and cost of retrieving
  - may suspend GET until lower cost connection is obtained
  - maintains user profile to suspend automatically, user can override system decision
- example 4: Mowgli
  - part of larger mobile/GSM project (Mowgli: Mobile Office Workstations using GSM Links)
  - main contribution: keep one long-lived TCP connection between MH and proxy (instead of one TCP connection per request)



WWW: The Canonical Example

- example 5: PDAs as Mobile WWW Browser
  - proxy requests documents, preprocesses them
  - image loading can be delayed, done in background while user reads text
  - if requested, images can converted (from true color to greyscale or B/W 1 bit)
- example 6: Glomop
  - part of Daedulus project
  - uses destillation: essentially a datatype specific compression
  - refinement: access a part at a greater resolution
  - user registers specific preferences for images
- last two examples: proxy *filters* information to reduce bandwidth requirements, mostly done for images only



WWW: The Canonical Example

- last but not least, work in our own lab:
  - prefetching HTML documents and inline images
    - prefetching is good to reduce perceived latency
    - prefetching requires lots of disk space
    - prefetching over a shared wireless channel detrimental to user traffic
  - therefore, prefetching is done in the proxy
    - more resources
    - latency reduced if we primarily prefetch far-away documents
    - proxy will always give preference to user requests
  - so far, we identified and implemented a number of different prefetching strategies, initial results are somewhat surprising
  - currently, more systematic effort under way to collect trace data and run experiments in a more controlled environment