# COMP5527 Review

11500811G QING Pei edwardtoday@gmail.com

April 24, 2012

## Overview

- **Mobile computing**: techniques that allow mobile users to use portable computing devices to run stand-alone applications and / or to access remote ones via wireless networks.
- Challenges
  - How to ensure **high data availability** in mobile computing environment, where frequent disconnection may occur?
  - How to ensure **desired level of consistency** among the copies of data in servers and mobile devices, where battery power is limited?
- Distinct characteristics of mobile wireless computing
  - **Wireless communication**
    - More frequent disconnection
    - Low and asymmetric bandwidth, high latency, high error rate
    - Greater variation in available bandwidth
    - Increased security risks
  - **Mobility**
    - Entities of mobility: physical (device), user (traveling people), process (mobile code)
    - Modes: personal/service/session/application
    - changes in physical/logical address & system configuration (network topology)
    - Location management problem: how does the network know where the intended recipient of a message is currently located?
  - **Mobile device**
    - screen/storage/battery life/variance in design and usage/more frequent breakdown and shorter lifespan

## Wireless networks

- WSAN Wireless Sensor Area Network
  - **ZigBee**: 250 kbit/s for embedded applications requiring *low data rates* and *low power consumption*
  - **RFID**
- WPAN Wireless Personal Area Network ~10m
  - Infrared Data Association (**IrDA**), ~1m, 875 nm wavelength, 9600 bps to 4 Mbps, require *line of sight*
  - **UWB** Ultra-wide-band: 480Mbps ranging up to 30ft, for military radar systems
  - **Bluetooth**, ~10m, 720 Kbps, does not require *line of sight*
    - 2400-2480 MHz
    - FHSS frequency-hopping spread spectrum: transmit chunks of data on up to 79 bands (1 MHz each; centered from 2402 to 2480 MHz), usually performs 800 hops per second
    - Bluetooth is a **packet-based protocol** with a **master-slave structure**. One master may communicate with *up to 7 slaves in a piconet*; all devices *share the master's clock*.
    - 1.2: data rate 1 Mbit/s, app throughput 0.7 Mbit/s
    - 2.0+EDR: data rate 3 Mbit/s, app throughput 2.1 Mbit/s
    - 3.0+HS; data rate 24 Mbit/s
    - Device ID (48 bit MAC address, unique worldwide)
    - master's clock (28 bit, 3.2 kHz, 24hr per cycle)
    - all active device with **3-bit** active member address (AMA)
    - **parked devices** cannot actively participate in the piconet but are known and can be reactivated. **8-bit** parked member address (PMA). One slave has to switch to park mode if there are 7 active slaves and a parked device wants to communicate.
    - stand-by devices don't participate
  - IEEE **802.15**: WPAN standards
    - aims at standardizing **MAC** and physical layer (**PHY**) specifications of bluetooth
    - addresses **coexistence** with other wireless devices operating in unlicensed frequency bands
    - framework enabling interoperable, stable, and scalable wireless mesh networking
- WLAN Wireless Local Area Network ~100m
  - 802.11a,b,g,n (**WiFi**)
- WMAN Wireless Metropolitan Area Network ~15km
  - 802.16a,d (**WiMAX**)
  - Multimedia QoS: connection-oriented, not only connectionless based
- WWAN Wireless Wide Area Network national coverage
- Satellite global coverage, 2.4kbps-2Mbps
- Mobility management
  - Handoff management
    - Hard handover: break-before-make
    - Soft handover: make-before-break
      - expensive hardware
      - Substantial interference due to near-far effect in CDMA networks without soft handovers.
  - Location management
- Cellular networks
  - 1G: analog, only voice comm. **AMPS**, **TACS**, **NMT**
  - 2G: digital, basic data services (9.6~19.2kbps). **DAMPS**(TDMA), **CDMA**, **GSM**(most popular), **PDC**
  - 1G and 2G are circuit-switched
  - 2.5G: packet-switch systems, speed up to 144kbps, efficient spectrum utilization, always-on but not paying by-the-minutes, only software update from 2G, foundation of 3G infrastructure. **GPRS** 115kbps (practical 40kbps), **CDMA2000 1x** 144kbps (practical 40~56kbps)
  - 3G: based on CDMA and IP, higher data rates 144kbps-2Mbps depending on the level of mobility, QoS. **EDGE** peak 384kbps, downlink 75-150kbps, **CDMA2000 1x EV** peak 2.4Mbps(EV-DO) 3Mbps(EV-DV),

**WCDMA** 144kbps/384kbps/2Mbps for diff mobility. **IMT 2000** industrial standard.

- 4G: 50-100Mbps, IPv6, QoS. 3GPP **LTE**: OFDM for downlink, adaptive modulation and coding, MIMO, all-IP, 100M down 50M up in 20 MHz spectrum, 5km~30km~100km coverage, up to 200 active users per cell (5 MHz), optimized for 0-15km/h but support high mobility, latency 5 ms user 50 ms control plane

- Mobile Ad Hoc Network (MANET)
  - for low setup time, no infrastructure, pairwise connection of devices nearby
  - Characteristics: Lack of a centralized entity. Network topology changes frequently and unpredictably.

## Mobile computing models

- Mobile Client/Server
  - Simple C/S
  - Client/Agent/Server
    - complex client requests can be managed by agent with only the final result transmitted back to the client
    - server can offload some activities to agent (e.g. compression)
    - agent can cache results to improve performance
    - need to change client code to comm with agent
    - increased system overhead
  - Client/Intercept/Server
  - Think client: mostly for online access to web content; server side plays the main functional role; limited UI and capabilities on client side.
  - Smart client: allows working in occasionally connected env with offline access and resynchronization upon reconnection; both c/s have good capabilities; client incorporated mobile db tech for persistent data storage.
- Mobile Peer-to-Peer. Hosts play the same role and cooperate to
  - Discover peers and resources
  - Route every requests
  - Perform specified tasks
- Mobile Agent
  - Code mobility (with computation, data and state)
  - Autonomous and asynchronous
- WAP Push
  - Push initiator, provides *domain of Push Proxy Gateway* and *client address*, comm with PPG using Push Access Protocol (on top of HTTP)
  - PPG uses OTA protocol (on top of HTTP or Wireless Session Protocol) to deliver content to WAP client, store the message, maintain status of each message, respond to PI using XML result notification indicating result of delivery
  - Service loading (URI to content to be cached, indicate whether content be exec immediately or just cached), service indication (message info or URI to it) and cache operation (remove obj from cache). SI and CO are optional.

- Android is a *software platform* and *operating system* for *mobile devices*, based on the *Linux kernel*, developed by *Google* and later the *Open Handset Alliance*.

## Mobile data management

- Objectives
  - reducing the amount of data transmitted over wireless networks
  - reducing the response time of accessing data via wireless networks
  - providing data caching on mobile hosts
  - maintaining consistency of data and transactions
- requirements for mobile databases: access to accurate & up-to-date information with constraints
  - some apps can tolerate bounded inconsistency sacrificing strict **ACID** (atomicity, consistency, isolation, durability) and allowing weaker consistent models
  - long disconnection should not constraint availability
- Data aggregation in WSN
  - Center at Nearest Source
  - Shortest Path Tree
  - Greedy Incremental Tree

## Mobile data dissemination

- Goals
  - reduce access delay
  - minimizing energy consumption at client
  - maximize capacity of servers
- approaches
  - pull-based (on-demand mode)
    - **not scalable**, limited by bandwidth for *parallel* pull requests
  - push-based (publish-subscribe mode)
    - **scalable** due to *sequential* access, so **latency** degrades with data volume
  - hybrid
- **access time**: time interval between **request** from the device and **reception** of interested data item from broadcasting or data pushing or responding system. Dependent on *number* and *size* of the records to be broadcast.
- Tuning time: time interval between the **mobile unit gets ready** and a **desired record is broadcast**, i.e., the time MU spent listening actively to the channel. Power consumption proportional to t_tune.
- broadcast disk
  - item-based program is expensive to generate with large # of items
  - collect items of similar access probability into partitions (disks)
  - objectives
    - allow mobile users to access broadcast data with optimal **access time** and **power consumption**

        - determine how to organize the data items and schedule the time of broadcast
        - determine how MUs should retrieve and process the broadcast data items on air
    - **Flat disk model**: round-robin, equal priority, no consideration on # of subscription to a particular record. 012301230123
    - **Circular multi-disk model**: block of records is pushed with a repetition rate proportional to its hierarchical level, disks with same rotation speed, higher level disks with less number of records repeated more times.
    - **Multi-disk model with repetition rate proportional to priority**: still same rotation speed but repetition rate of a record is proportional to its priority level, popular records are assigned to multiple levels. 010102010102
    - **Skewed disk model**: block of records are repeated as per their priorities for pushing, thus entailing consecutive repeated transmission of a record block followed by consecutive transmission of another block. 000112000112
- Selective tuning
    - MU activates only when the data arrives, saving power
    - with the help of **directory** or **index**
        - Directory: overhead at beginning of each cycle, consisting of **start sign**, **pointers to records** and **end sign**. A device wait for directory before it can get tuned.
        - index: can be broadcast interleaved with data records. Relatively small containing only key names. *offset* maps to the beginning of next index
        - (1,m) indexing: broadcast the index every fraction 1/m of the data items; all data items have an offset to the beginning of the next indexed data item
        - Distributed indexing: replicate not the whole index but only those immediately follow the index segment. 2 levels, repeated and unrepeated.
    - extends broadcast cycle and increase t_access

## Mobile data caching

- Goal: Cache the most probable data item for **best future use**, given *limited* amount of cache *storage* and potential *changes* in data item values.
- Performance: hit ratio, access response time, energy cost
- Correctness: consistency
- Cache invalidation:
    - When? **synchronous** or **asynchronous**
    - To whom? **stateful** or **stateless** servers
    - What? **invalidation** with only an update notification or **propagation** with the new values or **aggregated information/views**
- Polling (client-initiated)
    - TTL: polls after TTL expiration. TTL=0 equivalent to polling
- Pushing (server-initiated)
    - Stateless asynchronous
        - no frequent unnecessary reports, more bandwidth efficient
        - clients get the report regardless of its requirement, client mistaken invalid data as valid if a report is missing
    - Stateless synchronous (client requests if no report in a period)
        - greater reliability, better consistency
        - unnecessary transfers of invalidation reports periodically, potential inconsistent data between two reports
    - Stateful asynchronous
        - only affected clients receive reports
        - client still presume that as long as there is no invalidation report, the copy is valid, leading to invalid data
    - Stateful synchronous
        - timely synchronization
        - high bandwidth requirement
- Cooperative caching in multi-hop wireless internet access
    - challenges
        - Multi-hop reachability from data source
        - Bandwidth constraints
        - Disconnection
        - Topology changes and peer dynamism
        - Resource constraints on mobile nodes
    - issues: cache *placement*, *discovery* and *consistency*
- Weak consistency: best-effort guarantee
- Delta consistency: deviation in values or time bounded by \delta
- Probabilistic consistency: guarantee given consistency level with specified probability p

## Disconnected operations

- A mode of operation that enables a client to continue accessing **critical data** maintained at the server during temporary interruption of network connection
- the extreme of weakly connected operation (low radio signal, intermittent connectivity, noisy channel), can be *involuntary* or *voluntary*
- Why? battery life, network charge, radio silence (vital for military applications)
- objectives
    - ensure **data availability** even under disconnection
    - tradeoff between *data consistency* and *autonomy of client*
- How? **Caching**, **Prefetching**: cache items that may be useful in future but not currently needed, utilizing the otherwise idle processing power or bandwidth, **Hoarding**: prefetching to prepare for disconnection
- Client states: **connected** (request satisfied from either local cache or server), **disconnected** (all requests must be satisfied from local cache) and **reintegration** (update related info, deal with conflicts)
- consistency
    - **strong consistency** requires atomic updates, which is difficult to achieve in mobile environments
    - **weak consistency** tolerates occasional inconsistencies

- transparency of replication
  - transparent replication: allowing systems that were developed to run on a *central server* to operate **unchanged** on top of a *strongly-consistent replicated data storage*. Oceanstore
  - Non-transparent replication: allowing systems to be developed assuming a relaxed consistency model. Therefore **applications are involved** in *conflict detection* and *resolution*. The challenge is to provide the right interface to support cooperation between apps and their data managers.
    - **Bayou**: a replicated weakly consistent storage system designed for mobile computing environment. User can read and write any accessible replica. Application-specific conflict detection and resolution method. **Dependency check** and **merge procedure** on *write* operation. **Eventual consistency** due to 1 writes are performed at all servers in the same order; 2 conflict detection and merge procedures are deterministic. **Anti-entropy protocol** for propagating writes among servers (pairwise propagation, all replicas receive all updates).
    - **Coda** filesystem from CMU: providing *constant availability* through *replication*; introducing a *disconnected mode* for portable computers (which makes Coda different from AFS).
      - cache whole file whenever possible, callback based
      - automate the portable computer model, auto-syn feature
      - Use server replication to improve scalability, read from any server, write to all servers.
      - disconnected mode: Venus emulates the server, logs updates locally in Client Modification Log
      - hoarding to let users keep important files in their hoard database
      - Upon reconnection: enters **data reintegration state**, Venus integrates locally changed files to servers. Does not prevent inconsistent updates but guarantees that inconsistent updates will always be detected and then apply user conflict resolution.
    - Andrew File System (**AFS**): adopts upload/download model; whole file transferred to and cached in client sites. **callback promise** token. Write-on-close. Only final close operation retain all updates it makes. AFS servers are **stateful**.

## Location-dependent index and query

- 3 key elements of location-dependent applications
  1. location-dependent queries: result depend on the location of the **origin of the query** or the **object being queried for**
  2. locations: obtained by mobile positioning tech, maintained by spatial databases
  3. location-dependent database
- location representation: **geometric** (n-dimension coordinate) or **symbolic** (logical real-world description)
- spatial data: data *representing* (location of a road) or *associated* with locations (name or speed limit of that road)
  - *discrete* or *continuous*
  - **vector data model** using points as basis to model lines, areas and so on or **raster data model** using grid to represent spatial feature. Generalized raster model called **tessellation** model, in which elements can be of *arbitrary shapes*.
- spatial queries
  - **new operators**: containment, *overlap*, *neighbor*, *spatial relationship* (to the left of, etc.), *spatial join*
  - typical queries: *range query* for objects with an area, *nearest neighbor*, *k-nearest neighbors*
- location-dependent queries (what's more than spatial queries)
  - client may move and be part of the query, result depend on direction of movement
  - may not directly contain location info but may have multiple values corresponding to different locations
  - temporal features (gas stations in the next 20 min)
- indexing
  - allow direct access to tuples satisfying the condition
  - mostly based on hash-table (for point query) and B-tree/B^+-tree
  - In practice, B^+-trees, allowing sequential access to range data by chaining together leaf nodes.
- indexing spatial data
  - partition the set of objects into subsets according to their locations
  - R-tree, R^*-tree, R^+-tree (2-D equivalence of B-trees)
  - R-tree: minimum bounding box, mbb can *overlap*
    - for point(range) search, check the mbb of each branch, if the mbb contains(intersects with) the query, search recursively down the sub-branches
    - complicated insertion
  - R^*-tree with some optimizations for insertion
  - R^+ tree
    - mbb at the *same level* do **not overlap**
    - object may have to be duplicated
    - higher storage overhead, more complicated insertion, faster searching
- moving object database
  - costly insertion, even worse for moving objects
  - trade-off: precision v.s. update efficiency
- indexing time dimension
  - 3D-R-tree for historical trajectories as 3d minimum bounding rectangle (MBR), simple but not scalable, no efficient query support
  - multi-version: nodes unchanged are linked
- Continuous query. Caching and prediction are useful.