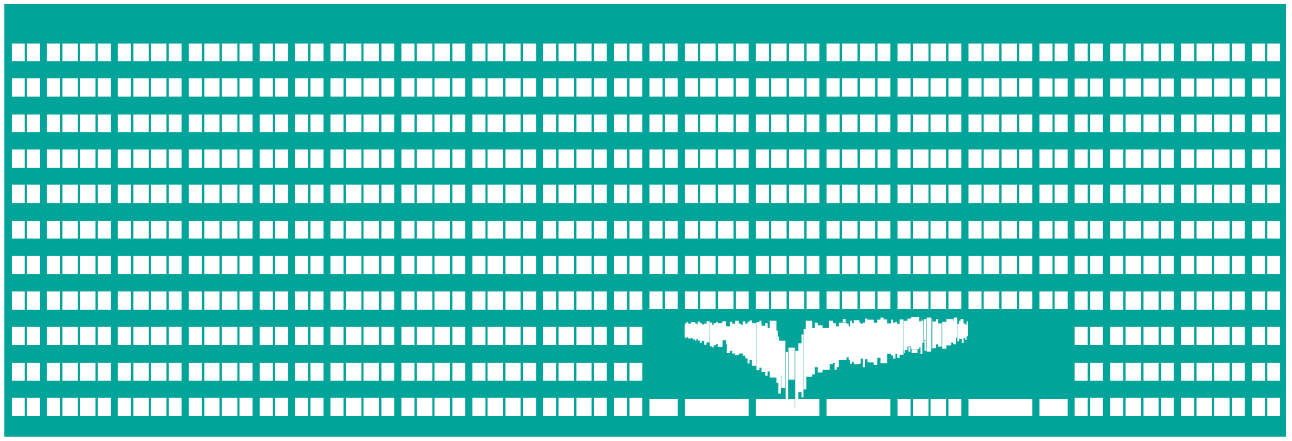
VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE | DEPARTMENT OF COMPUTER SCIENCE

Introducing Mobile Systems



MS (Mobile Computing)
Lecture 1

Contact Information

Office: EA-409

Address: Dept. of Computer Science, FEECS
VSB-TU Ostrava
17. listopadu 15
708 33 Ostrava-Poruba

Phone: +420 59 732 5896

E-mail: pavel.moravec@vsb.cz

Web: http://wiki.cs.vsb.cz/index.php/Moravec/cz
http://www2.cs.vsb.cz/moravec/MS/ (VSB only)

References

- Kamal R.: Mobile Computing Course Materials http://www.dauniv.ac.in/Mobilecomputing.html
- JING J., HELAL A., ELMAGARMID A., Client-Server Computing in Mobile Environments, ACM Computing Surveys, červen 1999
- Taha S., Shen X.: Secure IP Mobility Management for VANET, Springer, 2013, ISBN: 978-3319013503, 118 stran.
- Satrapa P.: IPv6. 2011, 409 stran, ISBN 978-80-904248-4-5.
- Faludi R.: Building Wireless Sensor Networks, O'Reilly Media, 2011, 322 stran, ISBN 978-0596807733.
- Kamal R.: Mobile Computing, 2007, Oxford Press, 582 stran, ISBN 9780195686777.
- Mavromoustakis C., Pallis E., Mastorakis G.: Resource Management in Mobile Computing Environments. Springer, 2014, 597 stran, ISBN 978-3-319-06704-9.
- Pitoura, E.: Data Management for Mobile Computing http://www.cs.uoi.gr/~pitoura/summer-school98/index.html

What is a Mobile Device?

A pocket-sized computing device, typically with:

- a small visual display screen for user output
- a miniature keyboard or touch screen for user input.

Examples:

- Mobile phones
- **Smartphones**, Tablets
- Navigation devices
- Cameras, portable media players
- Handheld game consoles
- IoT devices, monitoring devices, “smart” devices

...

Limitations of Mobile Devices

- Small (often lower-resolution) display (on some types of devices even grayscale/BW) on specialized devices
- Limited input options (phone/software touch keyboard)

- Lower operating memory & internal storage space (the latter partly solved by external memory cards)
- Limited CPU power + absent or limited GPU capabilities
- Limited instruction set (e.g. only software-emulated floating point arithmetics, no SSE, vector-processing instructions)
- Network connectivity issues

- Smaller library of system functions

- Power considerations
- ...

Mobile Operating Systems

- Vendor-specific firmware, vendor SDKs for custom builds
- Real-time operating systems
- iOS on Apple devices
- Linux-based:
  - Android
  - OpenWRT, DDWRT, …
  - Ubuntu Mobile (?)
  - ~~Firefox OS, Tizen, Sailfish OS, Jolla~~
- Windows
  - ~~Windows Phone 8.x, Windows RT (8), Windows CE~~
- ~~BlackBerry OS~~
- ~~Samsung BADA~~
- ~~Symbian OS~~

Mobile Frameworks

- HTML5-based frameworks
  - AngularJS, jQuery Mobile, Sencha Touch, …
  - Cordova, Ionic, ReactNative, PhoneGAP, …
- Xamarin
- ~~Flash Lite~~ – commercial
- CodeName One
- ~~JavaME – Java 2 Micro Edition~~, JavaFX
- 3rd party frameworks (of varying quality)
  - Java-based
  - Python-based
  - QT-based
  - LUA, …

Programming languages

- HTML(5) + Javascript + CSS(3)
- Java (SE – Android Development, ME, EE – web)
- Objective C, Swift – iOS
- C/C++ – native libraries, e.g. Android NDK, single-chip solutions, custom firmware development, RTOSes
    - Processing language (Arduino)
- C# – Windows 8, .NET compact framework, Xamarin

- Python
- Exotic choices: LUA, Lazarus (Pascal), Shell script, PHP, …
- …

New platforms changed everything!

- Device fragmentation → Platform fragmentation
- Vendors try to keep customer by vendor lock-in
- Applications sold by vendors directly through their Play Store/AppStore/Marketplace/Market/...

- 2 dominant platforms:
  - Android
  - iOS (~15%)
  - ~~Windows (Windows Phone 8, Windows Mobile)~~
  - ~~Blackberry (US) / Symbian (EU)~~

- Feature phones may still offer Java ME

Current multi-platform development

iOS

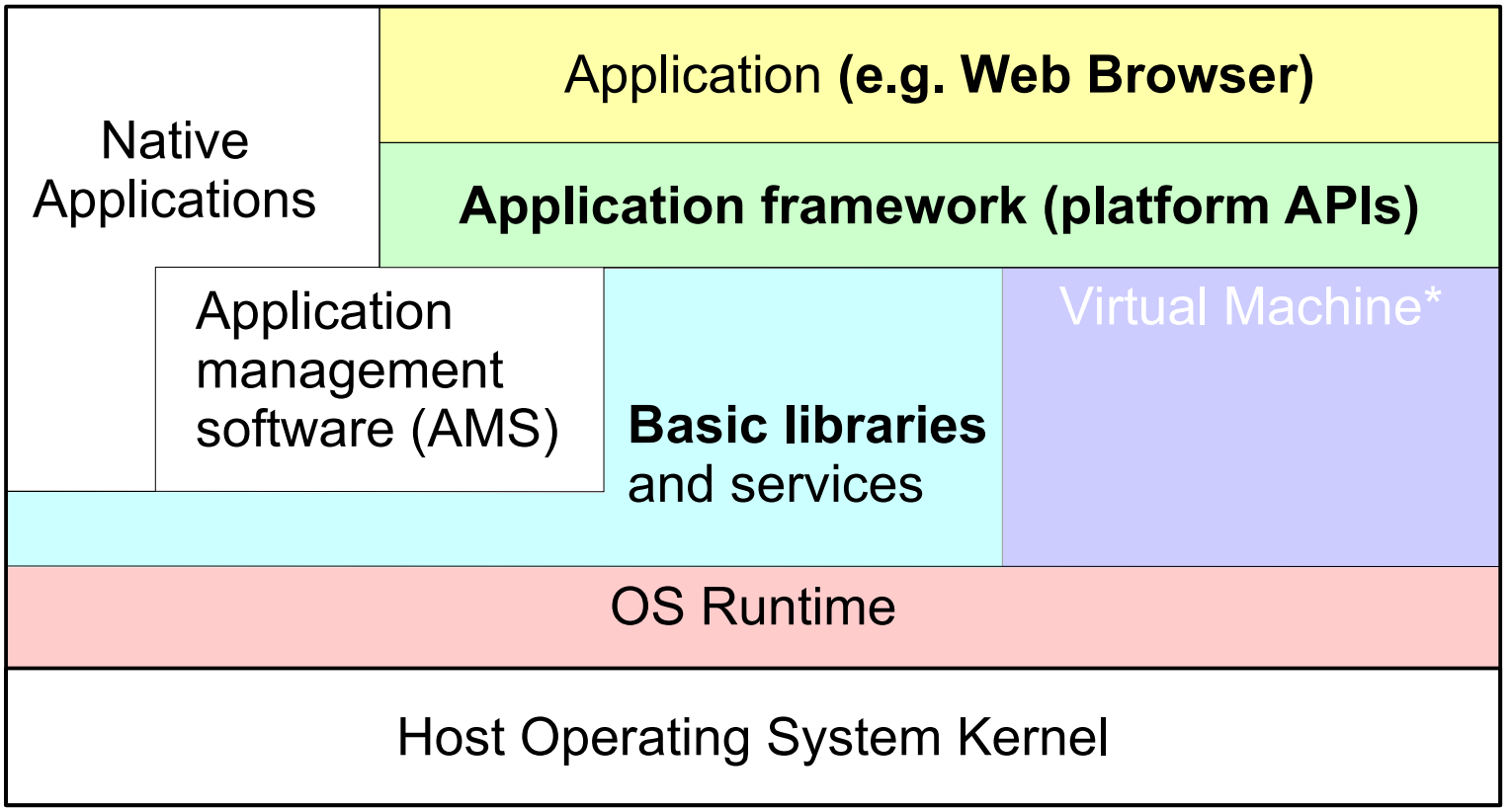
Windows Phone

Android	**iOS**	**Windows Phone**
User interface	User interface	User interface
Application layer (using native API)	Application layer (using native API)	Application layer (using native API)
Java/NDK	Objective C	SilverLight/XNA

Core library (C, C++, C#, HTML5+CSS+JS, …)

Business Layer
Access Layer (Data Access, Service Access)
Data Layer

Generic Mobile OS Architecture



* Virtual machine is not used on some platforms, which run native code directly. Usually, we use **platform APIs** for development, but on most platforms we can also write **native code** using built-in or even user-supplied **libraries,** or a HTML+JS+CSS application run in the web **browser** or embedded web view.

Mobile operating systems are discussed in OSMZ subject

Mobility

- The ability to move from one place to the other (e.g. student mobility in Erasmus).
- In multi-agent systems (according to FIPA – Foundation for Intelligent Physical Agents) we can distinguish between:
  - *Platforms* (locations) providing *services* and restricting/controlling *access* to them
  - *Agents* (objects) embedded in platforms at given time
  - *Mobility* is a type of special service
- In computer networks, *nodes* may also change *location*

Movement

Mobility is an interplay between locations (platforms) and objects (agents, nodes).

Differences are introduced by the direction of the synchronization necessary to perform the migration and its preconditions:

- Spontaneous move (no synchronization)
- Object move
- Transportation (location-initiated migration)
- Consensual Move (both sides agree)

Challenges of Mobility

- Location discovery
  - Reactive approach – send out a request when traffic must be delivered
  - Proactive approach – keep track of locations of all devices anytime
- Move detection
  - Best communication path
  - Cell selection in cellular networks
- Update signaling – location changes, device is alive
- Path (re)establishment – re(build) path to the device so that traffic can be delivered to its new location

AAAs of the Mobility vs. Security

- **A**nytime
- **A**nywhere
- **A**ny device

- **A**uthentication
- **A**uthorization
- **A**ccounting

Forms of mobility

- Wireless mobility
  - Campus mobility
  - Roaming
- Nomadicity
- Seamless mobility
- Ubiquitous computing
- …

Wireless mobility

- Allows movement in given area
- Real-time communications whenever and wherever the device is turned on (in ideal case)
- Evolution of other types of devices besides laptops:
  - smartphones, personal digital assistants (PDAs), tablets, phablets,
  - embedded navigation system, …
- Well known in 802.11 and cellular networks
- Different degrees of mobility: Limited to single node → connecting to different nodes on the go → passing through different nodes maintaining connection and transfers (e.g. call)

Campus mobility

- Mobility within a single administrative domain
  - university campus
  - hospital
  - conference rooms in a hotel
- No roaming across global Internet
- More easy to implement
- May be solved fully on 2nd layer of OSI-RM – VLANs shared among several nodes providing the network connection

Roaming

- Extension of connectivity service in a location that differs from the location where the service has been registered (home location).
- Wireless device is kept connected to the network, without losing the connection
- Technically supported by mobility management and AAA
- Home network and visited network (e.g. cellphones)
- SIM based/certificate based/username+password based, …

Nomadicity

- Ability to move from one location to another and start communication
- The communicating entity needs to terminate and restart communication as a result of the move
- Usually no communication on the move between nodes in nomadic environment infrastructure
- A nomadic environment is said to be transparent to the user, regardless of location, the device/platform, available bandwidth, and whether or not they are in motion at any given time.

Mobile vs nomadic user

- Mobile user
  - Stays *always* connected to the infrastructure when it is reachable
  - Uses the best available link
- Nomadic user
  - Does not usually require services during the move
  - Makes use of local infrastructure once the location has been reached
  - Uses potentially shareable devices and has a unique session on the device

Seamless Mobility (coined by Motorola)

- In ideal case, the user is not aware that the migration took place, or notices minimal changes.
- One device in motion, vs. moving between devices
- Typical examples are Internet connection and cells
  - Transfers continue when switching network connection type, ideally the user retains the same IP address
  - Cellular call is not interrupted during the movement
- Future: calls from landlines resumed on cellphone when leaving the office, automatic handover of call to integrated phone/handsfree in car, home entertainment synchronized as you move between rooms, etc.

Ubiquitous computing (1)

- Omnipresent devices in given environment
- Information processing is thoroughly integrated into everyday objects and activities
- Small, inexpensive, robust networked processing devices with “natural” interaction
  - Dust – devices usually with no visual output (nm→mm)
  - Tabs - wearable devices (cm scale)
  - Pads – hand-held devices (dm scale)
  - Boards – interactive display devices (m scale)

Ubiquitous computing (2)

- Additional forms for ubiquitous systems:
  - Skin – flexible 2D non-planar display surfaces and products such as clothes and curtains. E.g. fabrics based upon light emitting and conductive polymers, organic computer devices, foldable OLED displays.
  - Clay – arbitrary 3D shapes as artefacts resembling many different kinds of physical object (see also Tangible interface).

Recent trend – BYOD

"Bring your own device"

- Employees may bring personally owned mobile devices to the workplace
- Many issues:
  - End node security (device may contain a virus, malware, spyware, may be compromised, etc.)
  - Risk of data breaches
  - Data ownership issues, company policies
- Company applications
  - How to install, update, manage
  - Legality of use when "at home"
- Remote wiping of organization's data