Adaptable Mobile Applications: Exploiting Logical Mobility in Mobile Computing

Stefanos Zachariadis, Cecilia Mascolo & Wolfgang Emmerich

Department of Computer Science

University College London

Gower Street

London, WC1E 6BT

http://www.cs.ucl.ac.uk/staff/s.zachariadis

Outline

- Physical Mobility
- Logical Mobility
- Motivation
- Limitations of related work
  - Mobile application development
- Proposed solution: SATIN
- Future Work

Outline

- Physical Mobility
- Logical Mobility
- Motivation
- Limitations of related work
  - Mobile application development
- Proposed solution: SATIN
- Future Work

Physical Mobility

- Ubiquity of mobile computing devices
  - Laptops, PDAs, cellular phones
- Variable connectivity
  - Bluetooth, 802.11x, GSM/GPRS/CDMA/…/3G, infrared, docking
    - Nomadic, ad-hoc, base station mobility
    - Variable in cost and type
- Numbers increasing
  - 2002: 15.5 million PDAs, 2005: 700 million Bluetooth chips (Gartner)

Characteristics

- Limitations (compared to traditional computing)
  - Memory, battery power, CPU power, erratic (expensive) connectivity
  - Improving but lagging compared to desktop machines
- Different usage paradigms
  - Input/output
  - Speed, ease of use, frequent but brief usage
    - E.g. Check schedule
  - Reports show that users rarely install applications on mobile devices
    - Applications need to cater to users' needs throughout the device's lifetime

Characteristics (2)

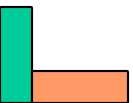
- Heterogeneity!
  - Device/Hardware (Physical)
  - Software/Middleware (Logical)
  - Network
- Very dynamic environment

Outline

- Physical Mobility
- Logical Mobility
- Motivation
- Limitations of related work
  - Mobile application development
- Proposed solution: SATIN
- Future Work

Logical Mobility

- Ability to sent parts of an application (or migrate/clone a process) to another host
- Popularised by Java
- Classification into paradigms
  - Client/Server (CS)
  - Remote Evaluation (REV)
  - Code on Demand (COD)
  - Mobile Agents (MA)
- Various middleware (mobile & stationary) systems exploit this



Examples of Logical Mobility

- Antivirus updates
- RPCs
- Browser "enhancements"
- Ringtone/Game download
- Distributed computing
- Automatic update rollouts

Advantages of Logical Mobility

- Flexibility
  - Dynamic applications
- Automatic software update
  - Maintenance
- New abilities
- Use of remote resources

Outline

- Physical Mobility
- Logical Mobility
- Motivation
- Limitations of related work
  - Mobile application development
- Proposed solution: SATIN
- Future Work

Observed Trends

- Further decentralisation of computing
- Computers: Smaller, faster, more resources, more personal, ubiquitous
  - Users are starting to carry portable processing environments of respectable computing ability
- Networking is pivotal
  - Devices can connect to various different types of networks at different situations: ad-hoc (Bluetooth, IrDA), the Internet (GSM/GPRS, 802.11b, …)

Motivation

- Potential of ubiquity of current devices largely untapped
  - Little interoperability because of heterogeneity
- New class of applications
- Investigate the use of Logical Mobility in mobile computing middleware
- Prove that logical mobility can bring tangible benefits to mobile application developers and users
  - Benefits include faster operation, less user-interaction, services offered based on context and location, reduced cost, better user experience
- Self-Organizing Systems

Outline

- Physical Mobility
- Logical Mobility
- Motivation
- Limitations of related work
  - Mobile application development
- Requirements
- Proposed solution: SATIN
- Future Work

Deficiencies of Related Work

- Limited use of LM
  - Usage of LM to provide reconfigurability to middleware
    - ReMMoC, UIC
    - Allows interaction with services provided by heterogeneous platforms/middleware systems
  - Usage of particular LM paradigms to provide particular services to applications
    - LIME (MA), PeerWare (REV), Jini (COD)
  - Others are not really geared for mobile networks
    - In Fargo-DA disconnections are announced

Current Mobile Application Engineering (PalmOS)

- Event driven, single threaded applications
- Files (Applications & Data) stored in main memory (usually 8MB).
  - Files stored as databases (collection of records)
- Developers compile application into a single file (Palm Resource, PRC)
- Application data can be stored in multiple Palm database files (PDBs).

Current Mobile Application Engineering (2)

- Very limited use of libraries
- Applications have a unique identifier, Creator ID (4 bytes)
  - Registered on a central database
  - Identifies PRCs & PDBs to the OS

What is Wrong with this Model?

- Very limited code sharing
  - On the device itself, between different devices
- Monolithic applications
- Difficulty to update application
- No versioning scheme for libraries
- No standard way to know which PRCs a device has.
- Difficulty to install applications
  - Statistics suggest that majority of users never install any 3rd party application

Outline

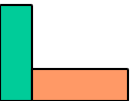
- Physical Mobility
- Logical Mobility
- Motivation
- Limitations of related work
  - Mobile application development
- Proposed solution: SATIN
- Future Work

Proposed Solution: SATIN

- Component based middleware
- Stresses modularity
  - Encourages decoupling of applications into modules
- Allows for static & dynamic configuration
- Minimal footprint
- Relies on developers following guidelines
- Offer usage of LM
- Lightweight
  - Runs in PDAs
- Network-independent
  - IP

Capabilities

- A SATIN component is a capability
  - Ranges from applications to libraries
    - SATIN applications are collections of capabilities with an "executable" one.
    - SATIN is a collection of capabilities
  - A capability provides some functionality to either the user or other capabilities.
- Provide a versioning scheme
  - Revisions of a capability
- Unique Identification
- Dependency scheme



The Core

- The SATIN Core is the main component of the middleware
- The Core is a registry for Capabilities.
  - All Capabilities can be accessed via the Core
- The Core identifies Capabilities by their identifier
- Core is a Capability itself

The Core & The Registrar

- Registration of new Components through a Registrar
- If no registrar is available, then SATIN is statically configured
- Registrar can receive capabilities from many sources (local & remote)
- Implementations of the Core may be distributed

Example Capabilities: Advertising and Discovery

- Paramount importance
- Heterogeneity!
  - Different ways to do it
  - Multicast
  - Centralised registry (Core)
  - Interoperability with other middleware platforms (e.g. Jini)

Example Capabilities: Advertising and Discovery [2]

- What to advertise?
  - Capabilities
- Advertising and Discovery techniques are themselves SATIN capabilities
- Capabilities choose which advertisers can advertise them
  - Using the Capability Identifier
- Capabilities choose advertising message
  - XML based
    - <capability id=FTP><port>21</port></capability>
  - Advertiser-independent
- Recursion:
  - Advertisers advertising advertisers
    - Discovery of multicast groups, etc.

Principles: Logical Mobility

- Encapsulation
  - LM paradigms
  - Language abstractions
  - Group various LM entities together
  - Signature
  - Identification
  - Requesting/sending
  - Deploying (containers/hosting)

SATIN's Approach to LM

- Decoupled nature of SATIN offers itself for use of LM
  - Capabilities
- Three entities represent LM to SATIN
  - Logical Mobility Units (LMUs)
  - Extendable Capabilities
  - Logical Mobility Deployment Capability (LMDC)

LMUs

- Container
- Sent around the network
- Encapsulation of Classes, Object, RPCs and Data
- Dependency scheme based on capability identification
- Size information
- Source & Target information
- Can be Signed
- Unpacker
  - Threads

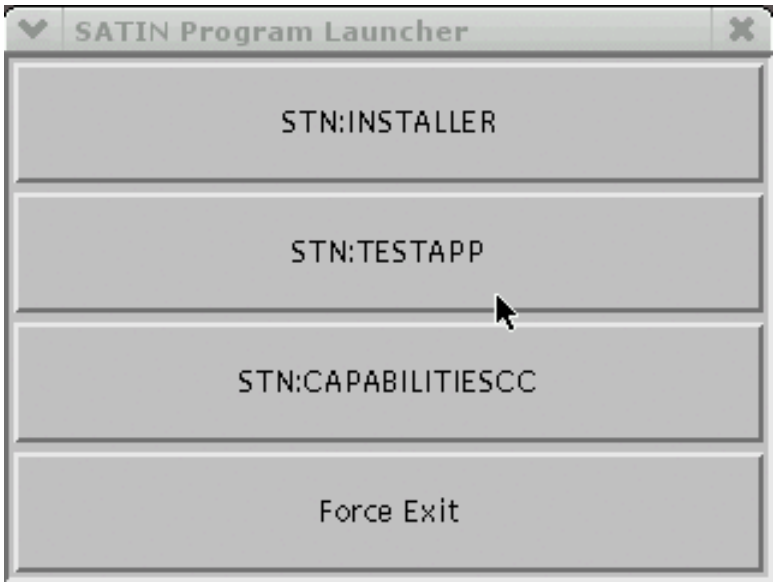
Extendables/LMDC

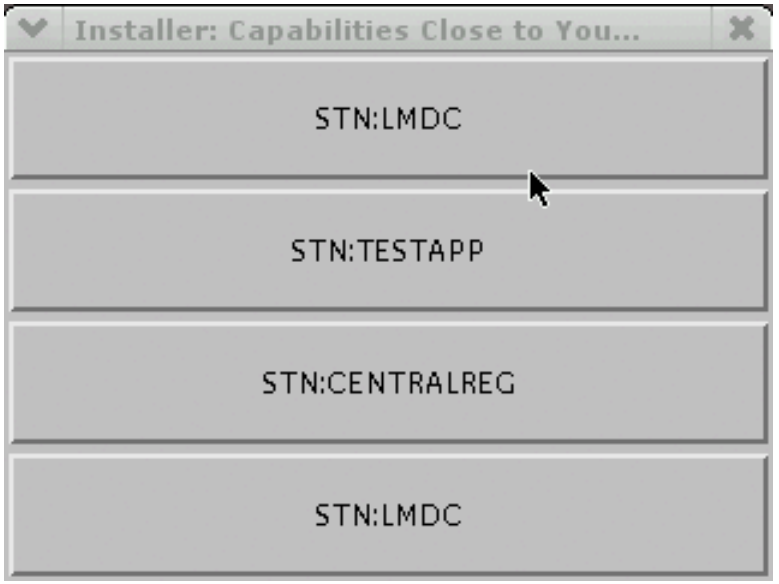
- Extendable capabilities can receive and host LMUS
  - Can accept or reject the LMU
  - Core or any other capabilty
- LMDC abstracts the usage of Logical Mobility
  - Requesting, sending, receiving, deploying

Example Application: Dynamic Launcher

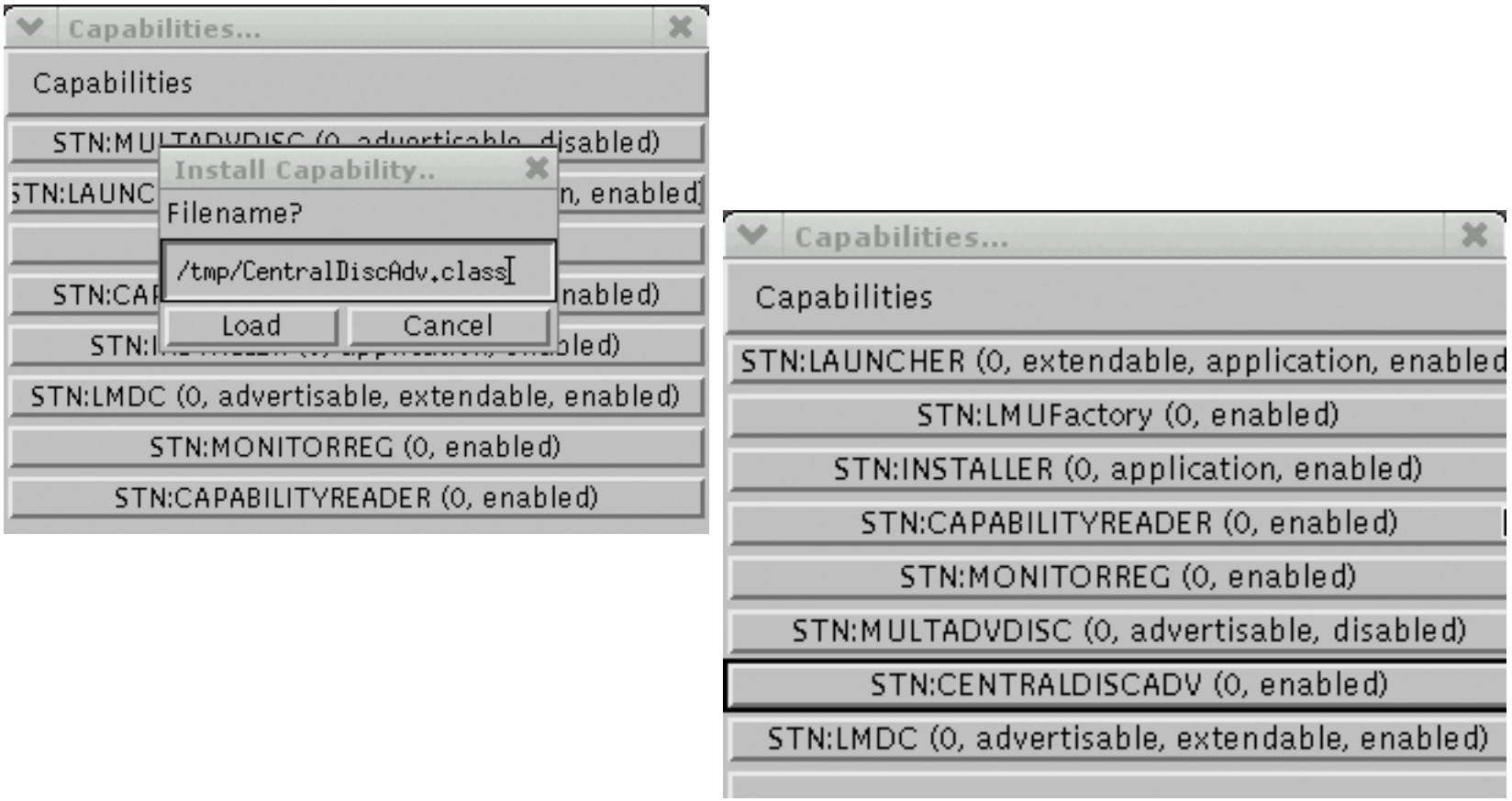
- Similar in Functionality to PDA Launchers
- Installs Capabilities from multiple sources
  - Centralised Source, p2p...
  - Uses any discovery techniques installed to find capabilities available
  - Uses LMDC to request and receive capabilities
- Transparent update
  - Using any discovery techniques installed and LMDC

Dynamic Launcher [2]





Dynamic Launcher [3]



Some Numbers

- Prototype
  - J2SE
  - Personal Java & J2ME considered
- Sizes:
  - 62K dist/satin-20030714.jar
  - 24K lib/kxml2.jar
  - 40K lib/µcode.jar
- Times
  - Startup Time on PDA: 21 seconds
  - Memory Usage on PDA: 1155KB
  - Update to PDA from peer: 2063 ms

Future Work

- Further Evaluation
  - More Applications
  - Comparison to similar applications that don't use LM
- New Classes of Applications Possible
  - Self-Organisation
- Scalability

Conclusion

- Physical Mobility
  - Increased popularity
  - Increased abilities
- Logical Mobility
  - Principles
  - Harness potential of mobile devices
- SATIN
  - Superset of previous approaches
  - Flexible use of LM to applications