Software in Mobility

exploiting logical mobility techniques in mobile computing middleware

Stefanos Zachariadis

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

Outline

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

Outline

- Physical Mobility
- Logical Mobility
- Motivation & Hypothesis statement
- Limitations of related work
  - Mobile application development
- Requirements
- 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 still
- 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

Example: SonyEricsson P800 Mobile Phone ("Smartphone")

- Released in December 2002
- Symbian OS 7.0
- ARM9 200MHz CPU
- 12 MB RAM (+ "Memory Stick" External)
- 128g
- J2ME
- 3 Network interfaces
  - Bluetooth, GSM/GPRS, Infrared
  - Ability to mediate between interfaces (routing)

Outline

- Physical Mobility
- Logical Mobility
- Motivation & Hypothesis statement
- Limitations of related work
  - Mobile application development
- Requirements
- 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 & Hypothesis statement
- Limitations of related work
  - Mobile application development
- Requirements
- 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

- 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

Outline

- Physical Mobility
- Logical Mobility
- Motivation & Hypothesis statement
- 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
  - OS allows applications to call other PRCs as a subroutine
- Applications have a unique identifier, Creator ID (4 bytes)
  - Registered on a central database
  - Identifies PRCs & PDBs to the OS

What's 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 & Hypothesis statement
- Limitations of related work
  - Mobile application development
- Requirements
- Proposed solution: SATIN
- Future work

Our Requirements

- Lightweight middleware (geared for mobile devices)
- Support for different networks
- Modular
  - Encourage decoupling of applications
- Ability to advertise & discover modules
  - Important to know what is available in the current context
- Offer direct access to LM
  - Dynamic applications

Proposed Solution: SATIN

- Component based middleware
- Allows for static & dynamic configuration
- Minimal footprint
- Encourages decoupling of applications into modules
- Relies on developers following guidelines

Principles: Architecture

- Modular
- Stresses componentisation
- Component identification
  - Dependency scheme
  - Versioning scheme
  - Easy to transmit
- Dynamic addition and removal of modules

Outline

- Physical Mobility
- Logical Mobility
- Motivation & Hypothesis statement
- Limitations of related work
  - Mobile application development
- Requirements
- Proposed solution: SATIN
  - Capabilities & the Core
- Future work

Capabilities

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

Capabilities (2)

- Have a unique identifier (String)
  - Local identifiers
  - Global identifiers (PalmOS)
- Provide a 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

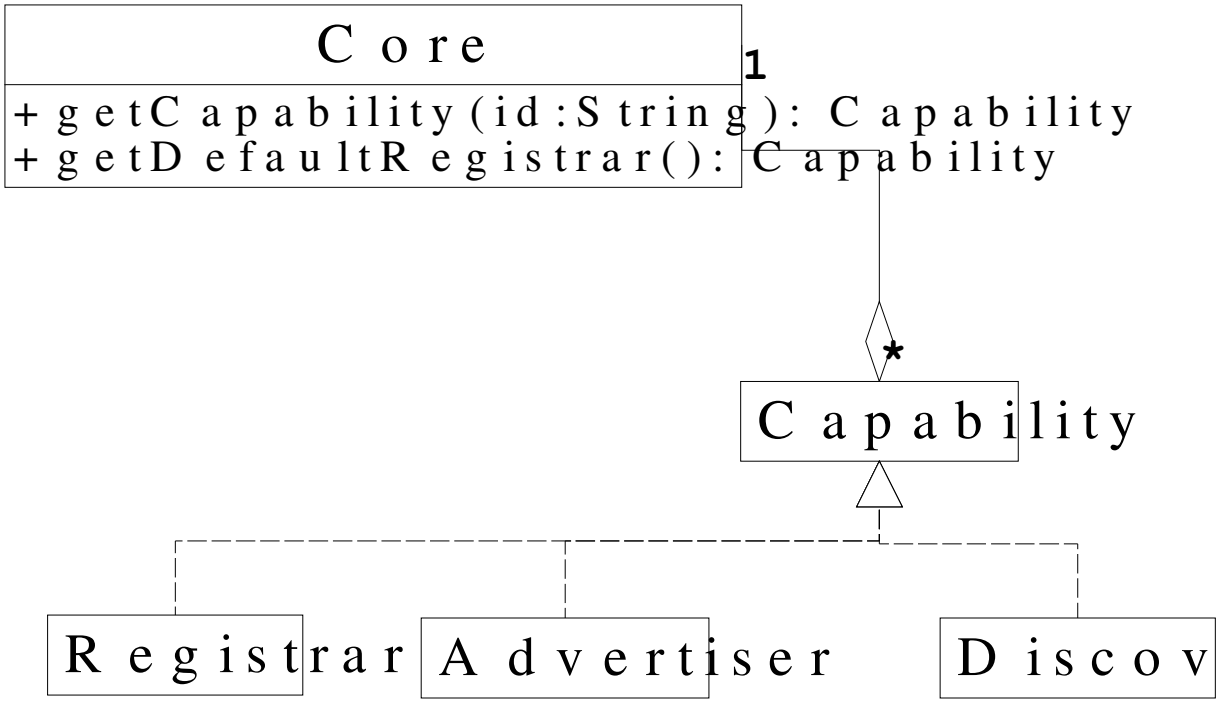
The Core & The Registrar

- Core needs a Registrar to add new Capabilities
  - Can have a default registrar
  - Registrar is a capability itself
- 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

Capability Types

- Applications
  - run()
- Advertisable Capabilities
  - Supply advertising message in an XML-based description protocol
    - <port>4662</port>

Recap (Core & Capabilities)



Outline

- Physical Mobility
- Logical Mobility
- Motivation & Hypothesis statement
- Limitations of related work
  - Mobile application development
- Requirements
- Proposed solution: SATIN
  - Capabilities & the Core
  - Advertising & Discovery
- Future work

Principles: Advertising & Discovery

- Paramount importance
- Heterogeneity!
  - Different ways to do it
  - Multicast
  - Centralised registry (Core)
  - Interoperability with other middleware platforms (e.g.. Jini)
- Allow components to register for existence of particular functionality
  - Ability to react to context changes

Advertised Capabilities, Advertising & Discovery

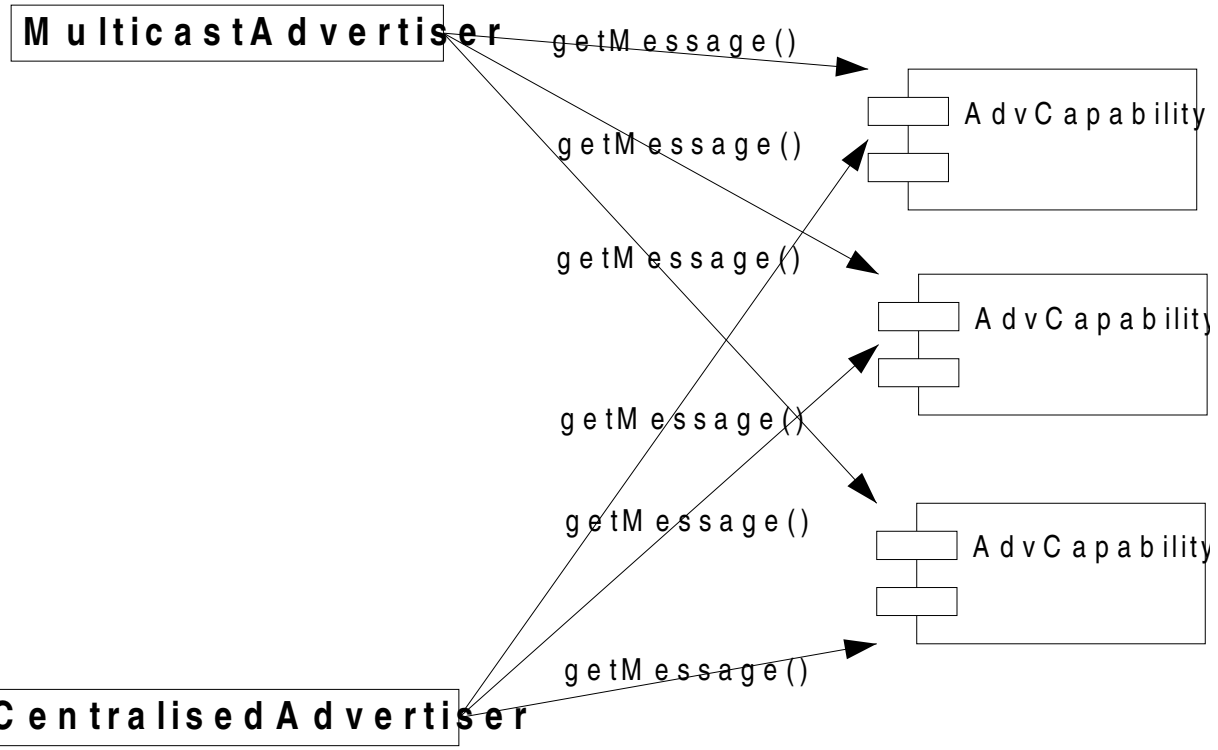
- Advertisers query Core for Advertisable Capabilities
  - Can specify which advertisers to allow advertising them
- Advertisers can be advertiseable capabilities themselves
  - Allows discovery of advertising techniques (say a multicast group)
- Advertisers create advertising message
  - <ed2k version="1"><port>4662</port></ed2k>

Context Discovery

- Applications (Capabilities) can register with discovery services for notification when a particular Capability is in reach
  - Using the XML description message
- Allows for reaction to context changes

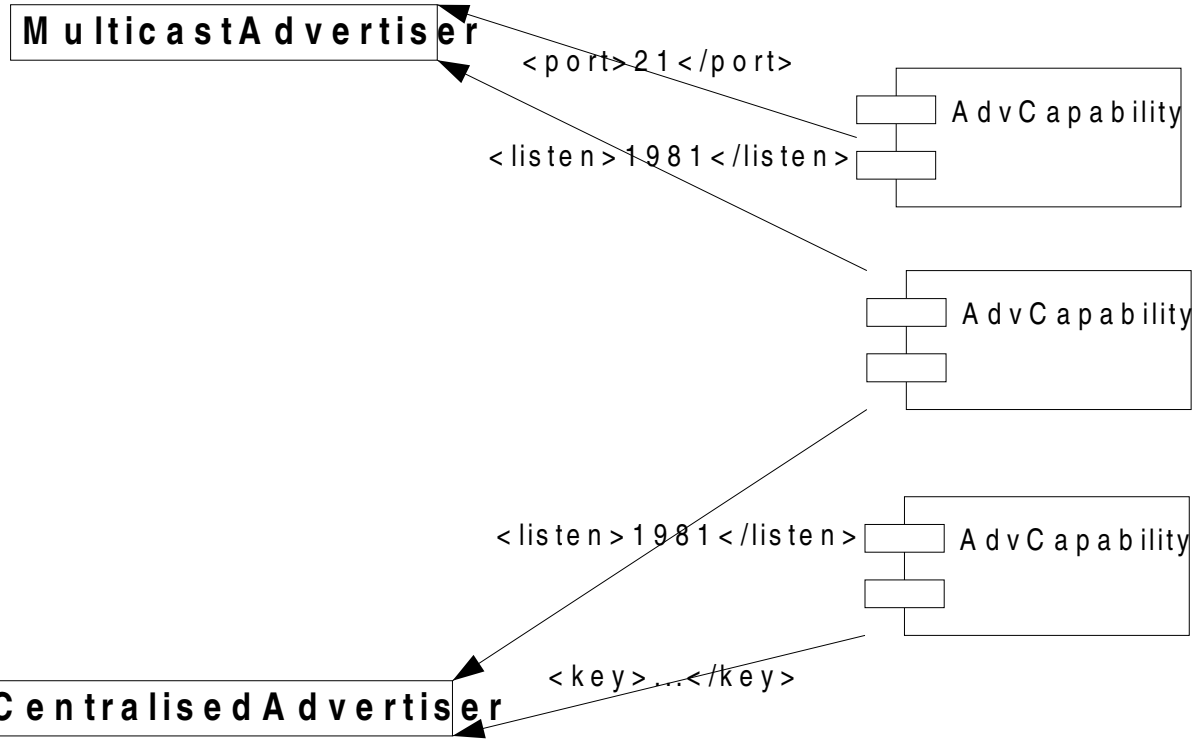
Recap (Advertising)

- Advertisers (periodically) query Advertisable Capabilities for message



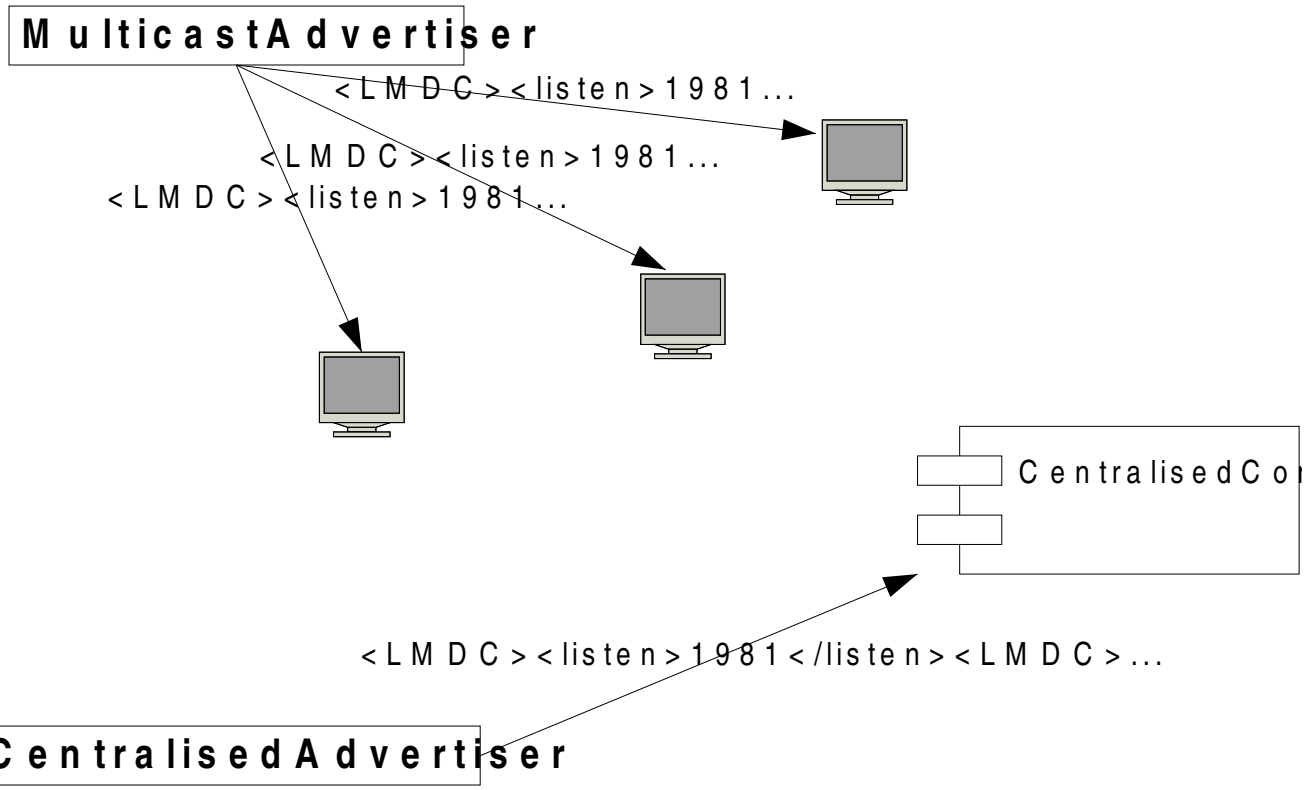
Recap (Advertising) [2]

- Message returned to selected advertisers



Recap (Advertising) [3]

- Advertising takes place



Outline

- Physical Mobility
- Logical Mobility
- Motivation & Hypothesis statement
- Limitations of related work
  - Mobile application development
- Requirements
- Proposed solution: SATIN
  - Capabilities & the Core
  - Advertising & Discovery
  - Logical Mobility
- Future work

Principles: Logical Mobility

- Ability to encapsulate all LM paradigms
  - Hosting environment
  - Requesting / sending
  - Deployment
    - Containers, acceptance/rejection
- Language abstractions
  - Objects, Classes, RPCs…
  - Code which does not map directly to the underlying platform is data
- Group various LM entities together
  - How to use unknown code?
- Signature
- Identification

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

- Receptacles for LMUs
  - Implemented as interfaces
- Core or any other capability
- Can accept or reject the LMU

LMDC

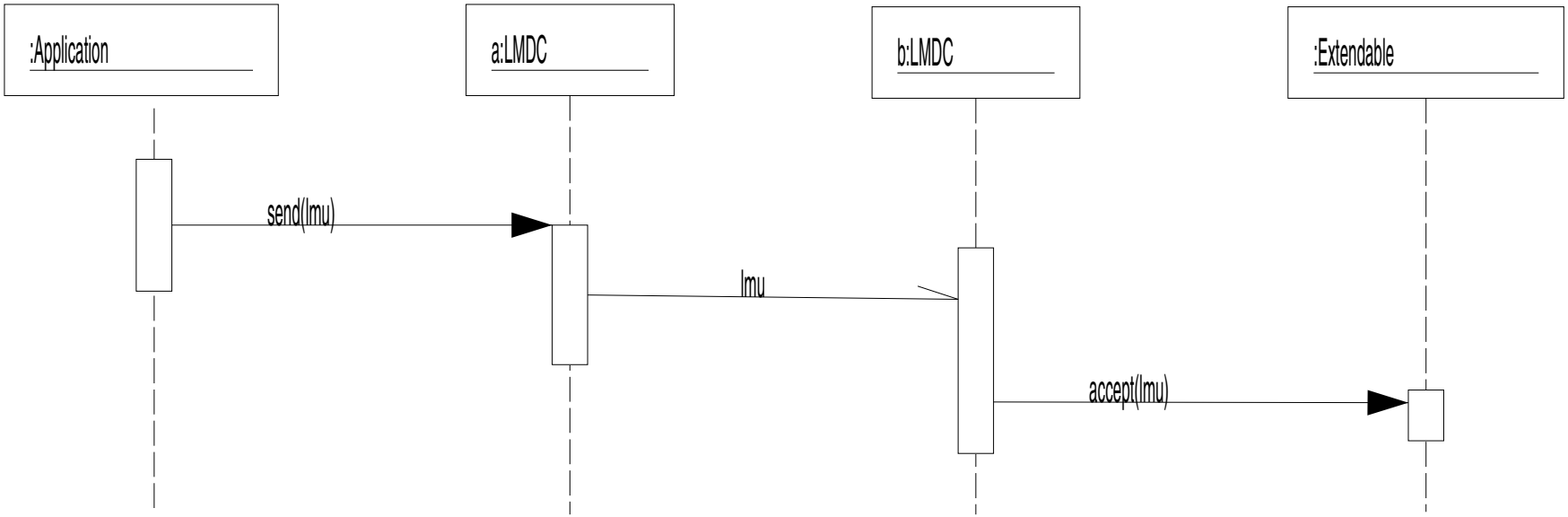
- Responsible for requesting and receiving LMUs
- Deploys LMUs in Extendables
- Capabilities send LMUs via the LMDC

Discovery, LMDC & LMUs

- Hosts can register for the advertising of specific functionality with an Advertiser capability
- Upon notification use the LMDC to receive required functionality
  - eg. codec

Recap (LM)

Sending an LMU in SATIN



Some Numbers

- Prototype
  - J2SE
  - Personal Java & J2ME considered
- 40K dist/satin-20030314.jar
- 24K lib/kxml2.jar
- 40K lib/µcode.jar

Outline

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

Future Work

- Looking for the killer app
  - Mobile code routing (NOT)
  - Adaptive applications
- Evaluation of approach
  - New applications possible
  - Comparison to applications that don't use LM
    - Definition of "best"?

Where we are so far

- Identified hypothesis statement
- Identified advantages of LM
- Identified shortcomings of related work
- Developed/implemented architecture
- Next Steps:
  - Applications
  - Evaluation

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

Logical Mobility?

- Popularity of bytecode
  - Java, .NET IL
- Interpreted languages (Scripts)
- Application profiles & data ?