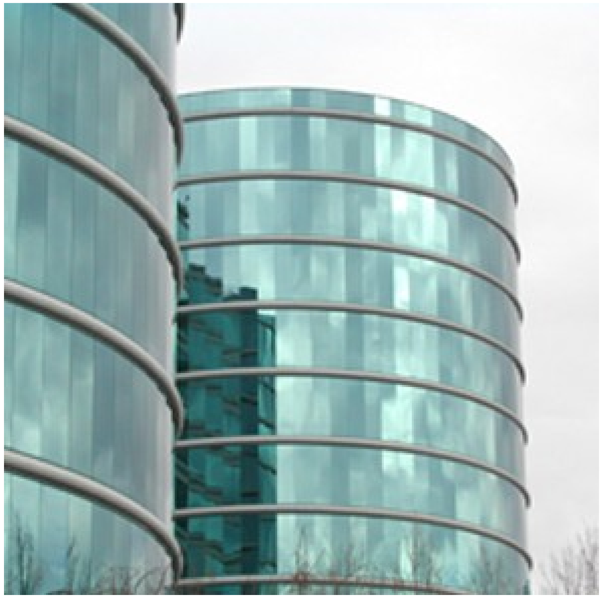
ORACLE®



Advancing The Java ME Platform

Roger Riggs
Oct 6, 2010

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Agenda

- Phase 0 – Current JSR Status
- Phase I – Modernize the core
- Phase II – Expand the platform

Oracle is Committed to Modernize Java ME

- Committed to getting JSRs moving
- Work is already underway
- Sharing CDC Next, CLDC Next texts next week, integrate feedback and file immediately

Phase 0 Get Moving Now

- Supply licensing terms for MSA 2 to SL
- Release JSRs in progress
  - JSR 290
  - JSR 242
  - JSR 229
  - JSR 253

Modernization Plan

- Phase I - Modernize ME platform
  - Adopt/Incorporate platform and language enhancements from JDK 6
  - Preserve CLDC/CDC split, but drive for larger common ground between them and the JDK
- Phase II - Expand ME Platform
  - Many requests for additional APIs/capabilities
  - Prioritize and distribute as appropriate to profiles, optional packages, or new optional packages
  - Work in parallel as much as possible

Modernization Goals

- Enable API and library re-use across ME, SE
  - Consistent core language features
  - Re-use of ME APIs with SE (Location, Sensor, etc).
  - Re-use of SE APIs with ME (JPA, Units)
- Reduce dissonance for developers
  - Consistent tools, programming patterns, etc.
- Fit within (updated) device memory budgets

Java ME Evolution – Phase I

- CLDC is the common subset of JDK for Java ME
  - Java Language
  - Java Virtual Machine
  - Java Runtime Libraries
- Minimize business disruption
  - Keep backward compatibility
  - Application ecosystem stability
- Existing Optional Packages
  - Without changes are backwards compatible
  - Can be updated to use Common Language, VM, Libraries

Language and VM Updates

- Compatible with the Java Language Specification
  - Generic Types
  - A Simple Assertion Facility
  - Annotations*
  - Extending the Java™ Programming Language with Enumerations, Auto-boxing, Enhanced for loops and Static Import
  - Unicode Supplementary Character Support (4.0)
- Compatible with the VM Specification
  - Memory Model and Thread Specification Revision,
  - Class File Specification Update

* CDC Supports full Annotations, CLDC supports a subset due to the absence of reflection

Library Updates

- Synchronize APIs already in CLDC/CDC to JDK 1.6
- Update built-in type support
  - String – extended to full Java 6 Strings
  - Math functions – Trig, BigDecimal, etc.
- Collections*
- Standard File APIs*
- Concurrency Utilities*
- NIO Buffers
- Create common specification for Generic Connection Framework (GCF)

* Subset as necessary to achieve footprint goals

CLDC Updates

- Common enhanced language, VM, and Libraries
- Retain equivalent level of functionality
  - Security model (same as CLDC 1.1.1)
  - No application dynamic loading of classes
  - Generic Connection Framework, networking, etc.

CDC Update

- Common enhanced language, VM, and Libraries
- Combine CDC and FP (No separate FP)
- Updates
  - Security - JCE, JSSE, JAAS
  - New I/O APIs for the JavaTM Platform
  - Networking,
  - Logging optional package

Discussion

- What are the issues?
- Can you commit to contribute and support?
- Lead, follow, or get out of the way!
  Which role will you take?

Java ME Direction – Phase II

- Phase I - Modernize ME platform
  - Adopt/Incorporate platform and language enhancements from JDK 1.6
  - Preserve CLDC/CDC split, but drive for larger common ground between them and the JDK
- Phase II - Expand ME Platform
  - Many requests for additional APIs/capabilities
  - Prioritize and distribute as appropriate to profiles, optional packages, or new optional packages
  - Work in parallel as much as possible

Breadth vs Depth Strategy

- Java ME must fill gaps in functionality
- Updates to existing APIs
  - For improvements to current functionality
- New APIs should be created for new functions
  - Must have a proactive committed owner with adequate resources
  - Must be designed to be usable across Java ME and Java SE
  - Must be developed in parallel to feed the eco-system quickly
  - Design for modular delivery with JDK 8 modules
- Avoid monolithic processes and complex decisions

Monolithic vs Agile

- JSRs with lots of functionality have proved to be less efficient models for JSR development due to scale and complexity
  - The amount of work for the SL and the complex decision process resulted in excessive delays
- Smaller more focused JSRs get results quicker
  - Maintenance also more efficient

Device Specific vs Reusable

- Profiles so far are not used and as such not reusable across all ME devices
  - MIDP on STBs
  - PBP/PP on Mobiles
- Criteria for pushing new functionality into current profiles vs new standalone reusable optional packages
  - Functionality specific to one device type only ?
  - Reusable across devices ?
  - Availability speed

Gaps to be Filled

- OpenGL ES 2.0 Java API
- UI - touch input, Stereoscopic display control
- Modular delivery of new APIs and libraries across ME, SE with explicit dependencies and fully secure delivery
- And more examples...

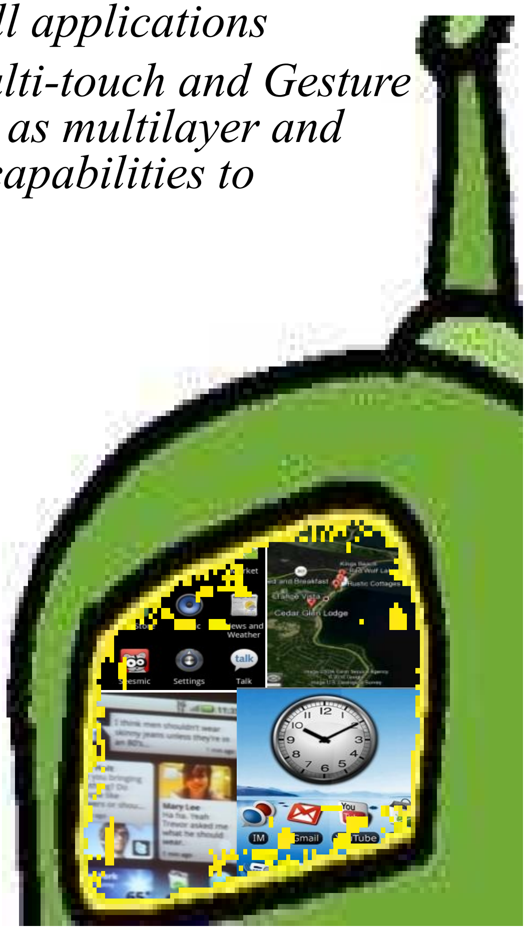
Core Modernization

Proposal

- *Java ME Core Platform modernization*

Features

- *AMS: Application Manager to support multiple application model and integrate Generic Application Store*
- *Profile: Service Framework (publish/access local services, e.g. contacts)*
- *Profile: Event Framework, i.e. cross application/platform event bus*
- *Profile: Networking extensions, e.g. Multicast*
- *Profile: Configuration Framework central to/for all applications*
- *Profile: add multi-touch and Gesture support as well as multilayer and printer output capabilities to User Interface*



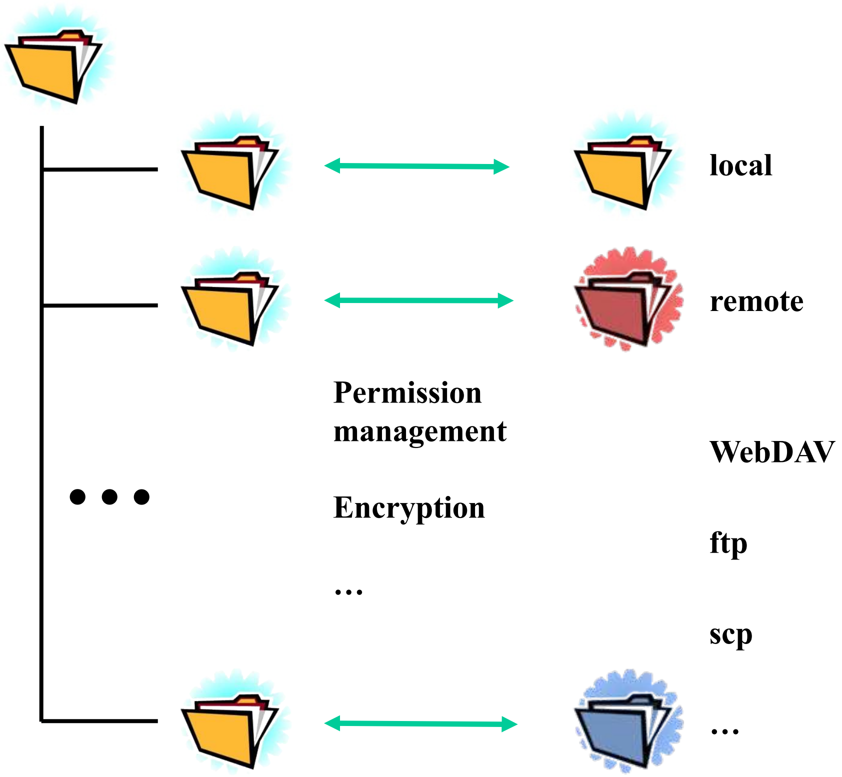
Advanced File System

Scope

Specifies a GCF based generic file system to access local and state of the art remote file systems such as Dropbox, Box.net, Mobile Me, WebDAV, …

Main Feature

- *support local file systems*
- *mount remote file systems over WebDAV, s\ftp, scp…*
- *allows applications to register new file system plugin (e.g. such as Dropbox, or Encrypted File System…)*
- *Generic file system interface adding permission management, file list retrieval, …*



UPnP DLNA

Scope

- *Specifies ability for apps to discover and remote control UPnP DLNA compliant devices on the local network. Gives the application a way to find devices, list available media (Music, Video) and control play,stop,... them ... i.e. act as a Control Point*
- *The specification could allow for the application to behave as a UPnP AV Media Server*

Main Feature

- *UPnP DLNA Client*
- *UPnP AV Media Server: Multiple applications can register different content however all is offered merged as one Media Server*

Related to

- *Platform support for Multicast*

SIM Support

Scope

Specify enhanced support for communication at the interface between card and handset

Main Feature

- *secure channel specification from ETSI*
- *contactless service registry defined by GlobalPlatform to provide a user interface for the contactless services in the card*
- *discovery of SIM card with SCWS / Servlet engine*
- *support events coming from SIM via TCP/ STK/HCI*

✓Secure Channel
✓HTTP support
✓Contactless registry objects
✓Events
✓…

Social Networking Interface

Scope

- *Specify APIs to JavaME allowing thereby to applications and local web services (servlets) to interface and host 3rd party social applications (e.g. hi5, LinkedIn, MySpace, Netlog, Ning, orkut, XING, Yahoo! …) like OpenSocial.*
- *This interface is both used for open Internet as well as Enterprise social networking.*

Main Feature

- *Generic Framework*
- *Interface for applications to define new Containers*
- *Retrieve Social Networking information*

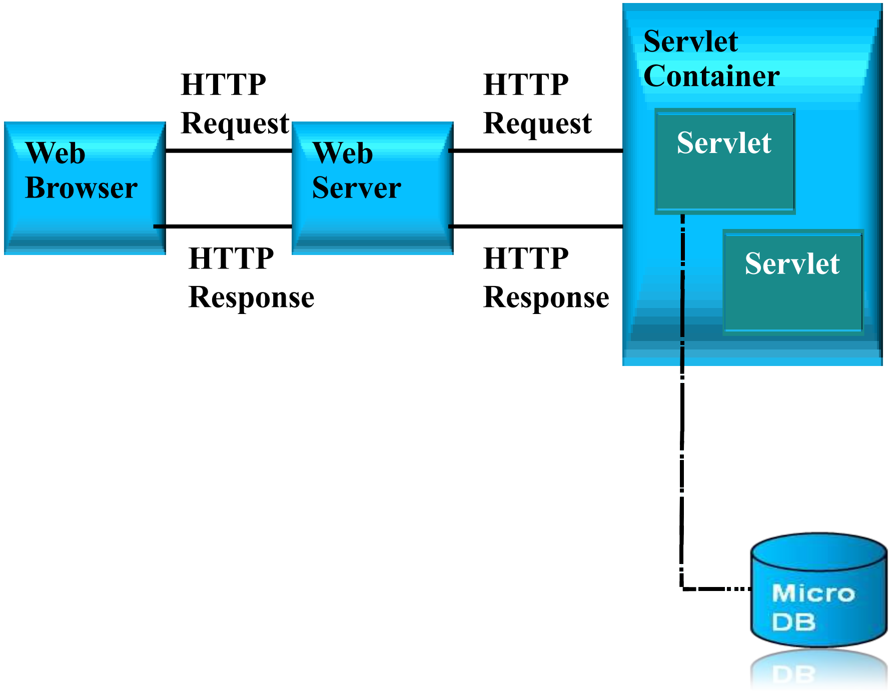
Servlet Application Model

Scope

Introducing a new Application Lifecycle Model to Java ME based on Servlets. This model lets a given bundle to register straight away as a service and to bind to the platform's local web server. It supports the on going trend to have service access for local or remote clients.

Main Feature

- *Combine Servlet Application Model as offered by JSE/JEE with constraints and limitation similar to those defined by Java Card 3 Connected Servlets*



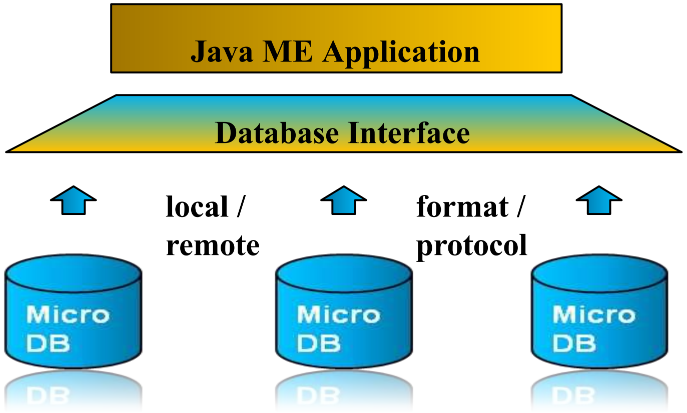
Database Interface

Scope

Specifies generic access to local and remote Databases

Main Feature

- *generic interface*
- *defines hooks for most widespread micro DB without mandating any*
- *JDBC inspired, investigate CLDC reusability*
- *allows applications to register as new DB plugin e.g. to interface either own local/remote format/protocol DB*
- *allows typical random access and relational searches*



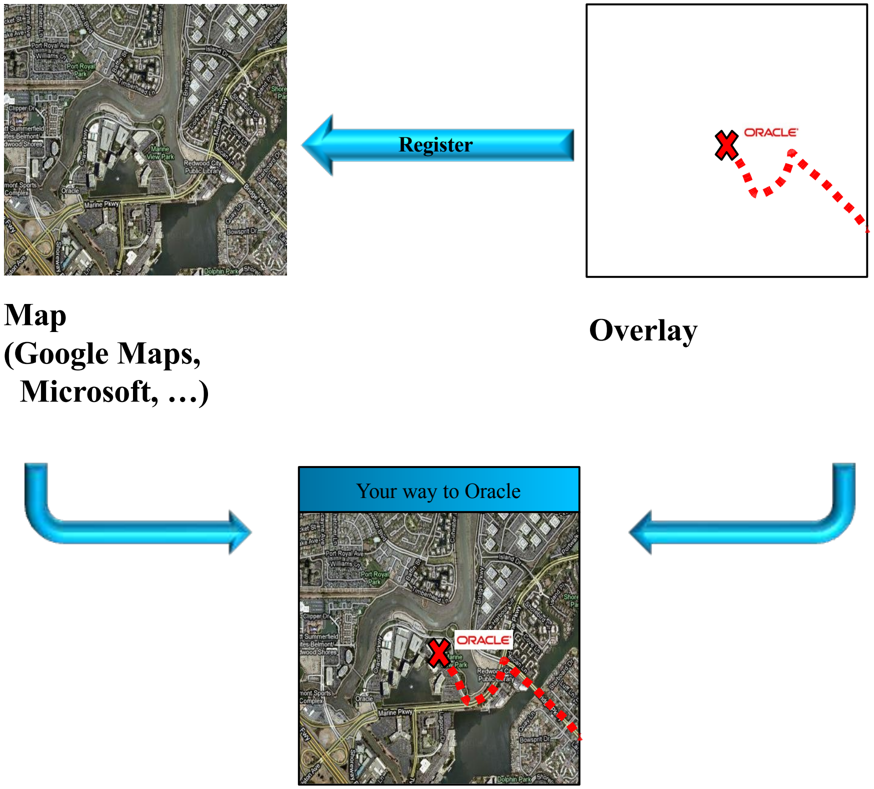
Maps

Scope

Specifies the mean to draw maps and overlays based on a given location by wrapping the platform native map engine.

Main Feature

- *Draw maps in a viewport given a location*
- *Interface for applications to register being an overlay*
- *Interface for applications to load/draw registered overlays*
- *Retrieve a selected/pointed location*



Register

Map (Google Maps, Microsoft, …)

Overlay

Your way to Oracle

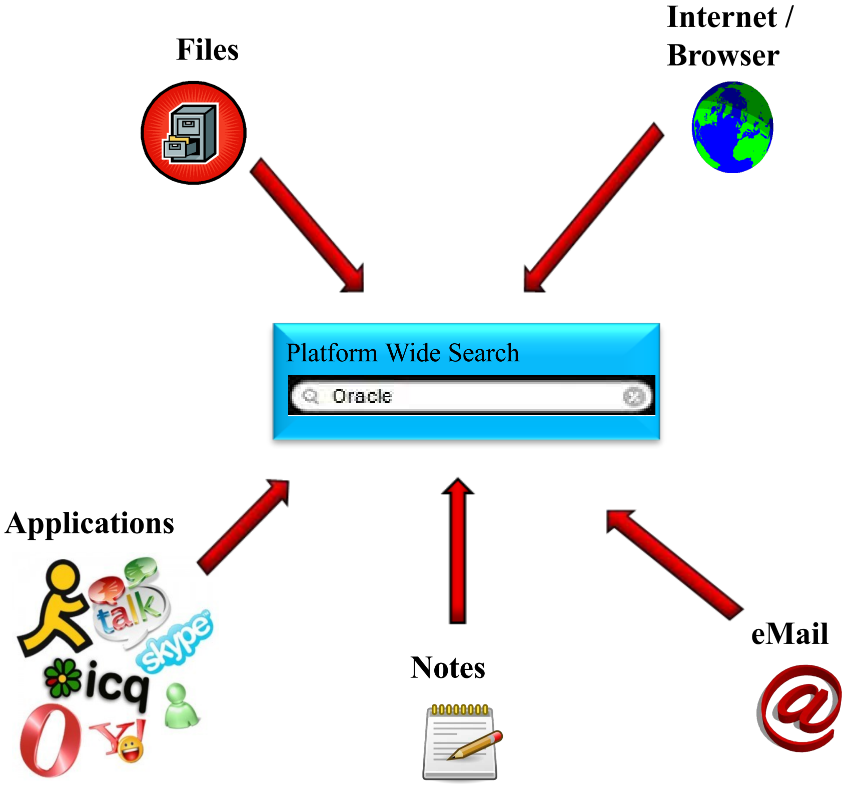
Platform Wide Search

Scope

Specifies platform wide search function allowing apps to look by keywords for platform objects

Main Feature

- *search by keyword*
- *returns list of platform objects accessible/ controllable by the java application (e.g. file, native/java apps, PIM objects, email, online store, ...)*
- *interface for applications to become searchable (allows thereby also to create new bridges to the internet: Yahoo, Google, Facebook ...)*



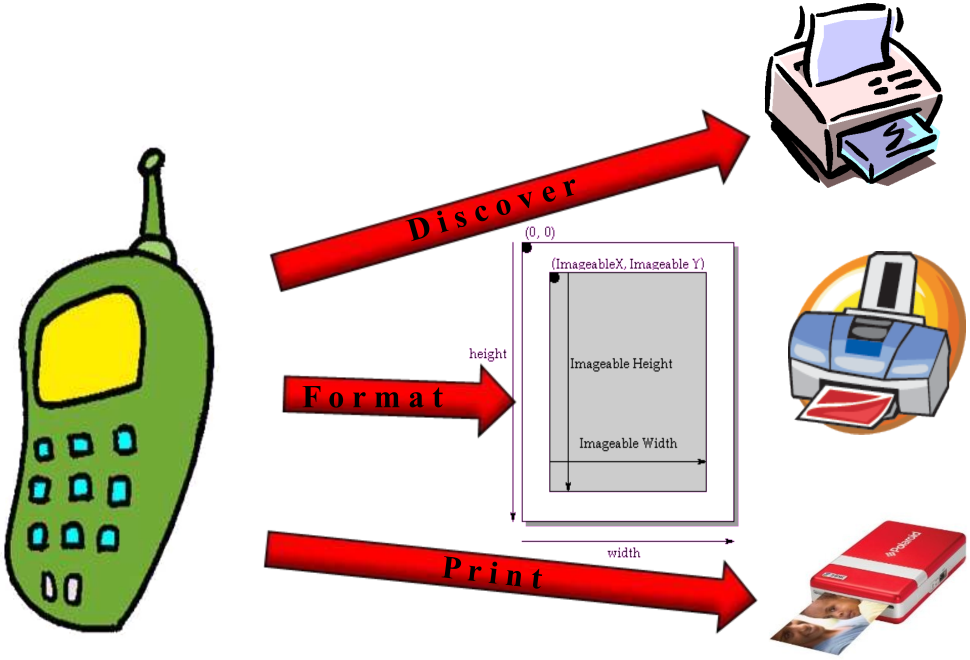
Mobile Printing

Scope

- *Add Printing capabilities to ME*
- *Create a suitable subset of JSR 6 (Printing for SE)*

Main Feature

- *Printer discovery (connected, proxied, Wifi, Bluetooth)*
- *Printer capabilities*
- *Page rendering*



JSR Finalization

Proposal

- *JSR 230: DS*
- *JSR 246: DM*
- *JSR 259: Ad hoc Networking*
- *JSR 266: Unified Message Box*

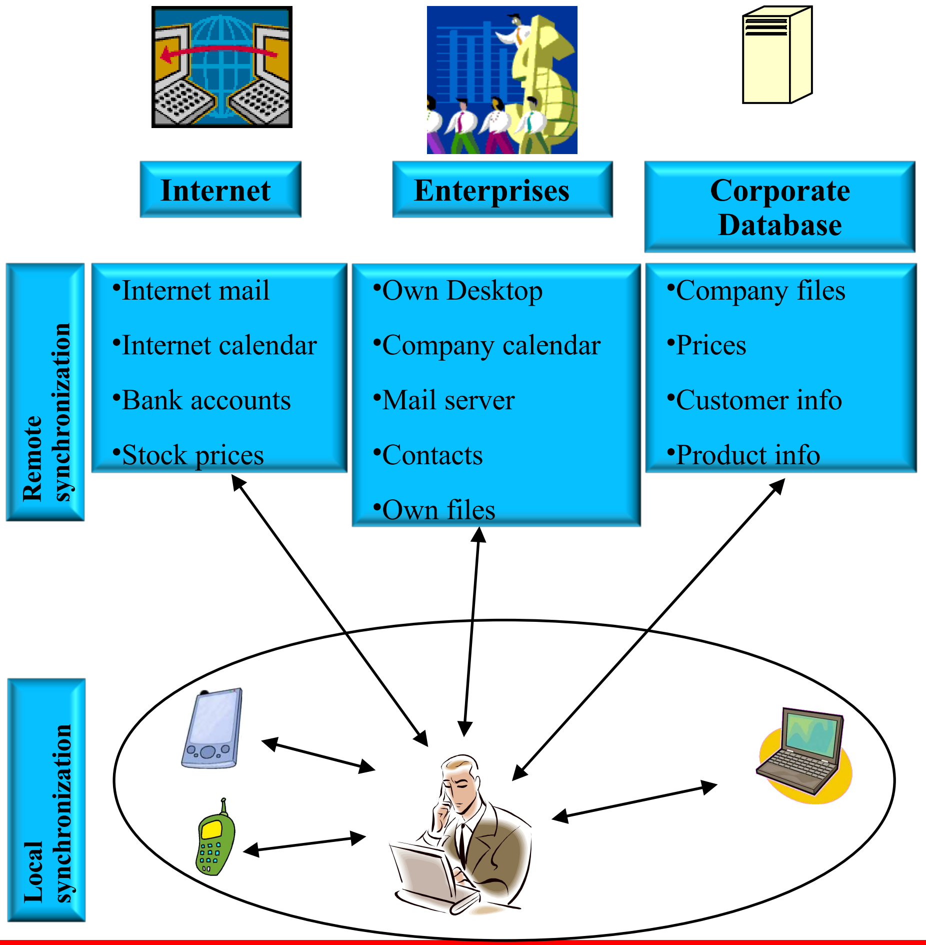
JSR 230: Data Sync API

JSR Scope

APIs allows to synchronize application specific data stored in the terminal with corresponding data stored on a server

Main Feature

- *API to synchronization engine (e.g. for adding, deletion, update of data)*
- *sync protocol agnostic*
- *usage of containers for data exchange*



	Internet	Enterprises	Corporate Database
Remote synchronization	•Internet mail •Internet calendar •Bank accounts •Stock prices	•Own Desktop •Company calendar •Mail server •Contacts •Own files	•Company files •Prices •Customer info •Product info

Local synchronization

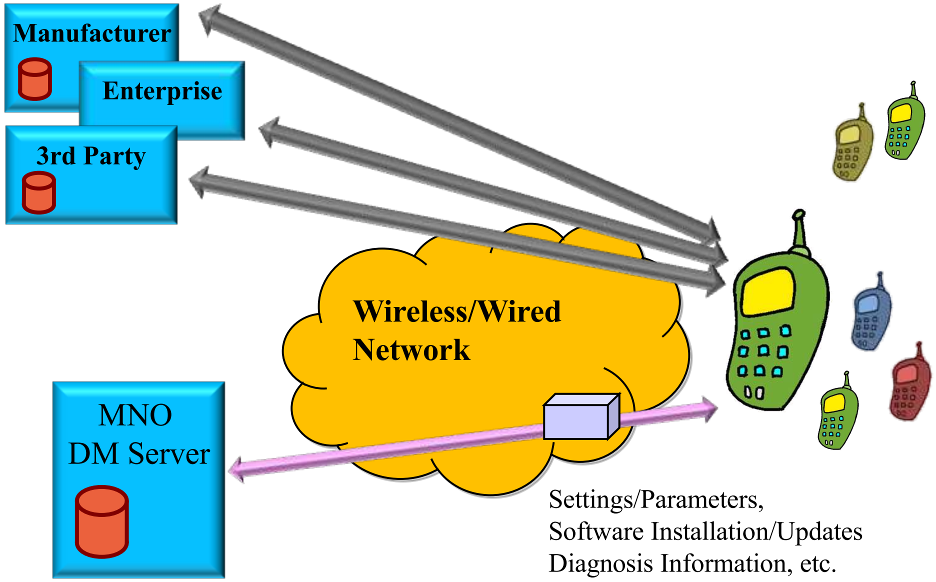
JSR 246: Device Management API

JSR-Scope

generic interface to device management implementation in the terminal

Main Feature

- *provides access to application parameters managed remotely on the device*
- *mechanisms to manage events, data monitors, session notifications*



Settings/Parameters,
Software Installation/Updates
Diagnosis Information, etc.

JSR 259: Ad-hoc Networking API

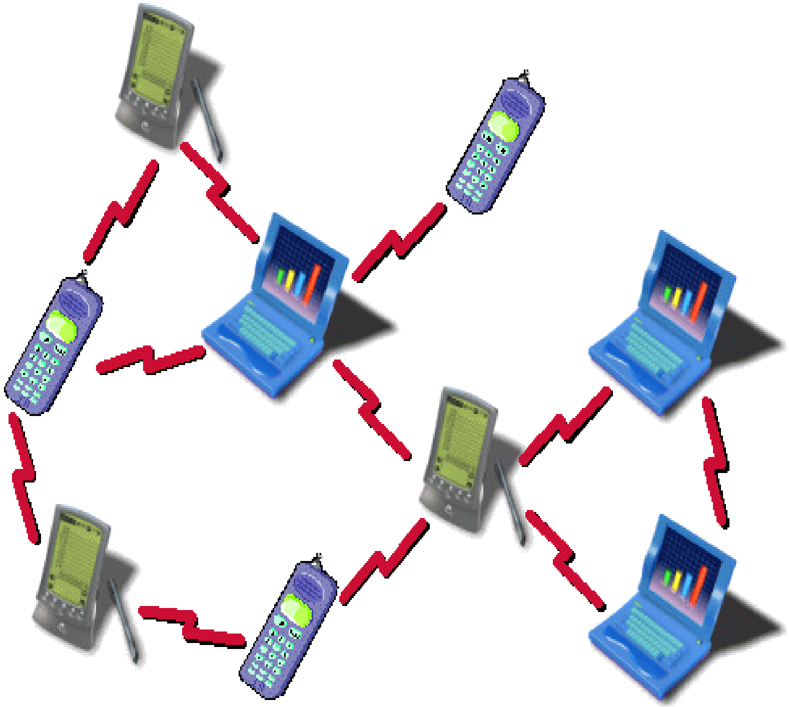
▪

<u>JSR-Scope</u>

APIs enable communication between mobile devices in a bearer agnostic peer-to-peer ad-hoc network environment

<u>Main Feature</u>

- *Service discovery*
- *Service registration*
- *Service availability alert*
- *Service and service capability inquiry*
- *Remote service consumption*



JSR 266: Unified MessageBox

JSR-Scope

- *Unified Message Box Access API (UMBA-API)*

Main Feature

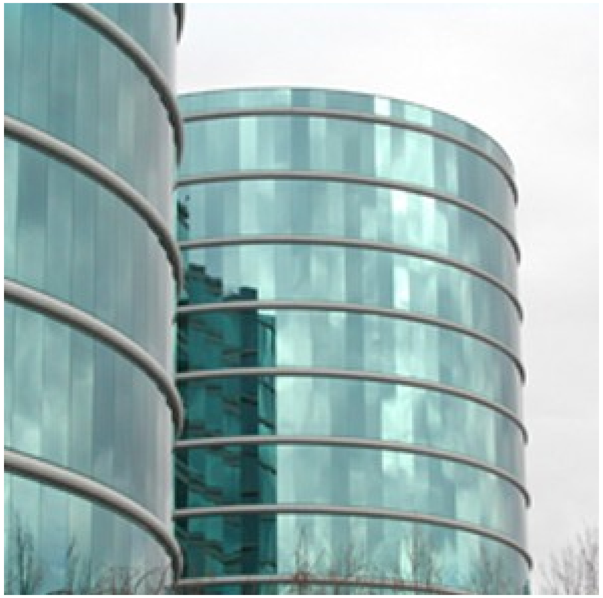
- *Management of message boxes. (query of info, rename, copy, delete?)*
- *Read, write, copy, move and delete content*
- *Access to message attributes*
- *Register listeners to be notified about modifications to the message box*

+ Extensions to support voice mail message boxes

ORACLE IS THE INFORMATION COMPANY

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.
The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE®



Backup

MSA 2 – Standard Platform

- 6Mb static footprint
- CLDC
- MIDP 3.0
- Mobile Media
- Messaging – SMS/MMS
- File and PIM
- Bluetooth
- Mobile I18n
- Sensor API
- SVG 2.0

- SIM card API
- Content Handler API
- Location API
- XML – SAX, STaX, DOM
- Internet Messaging Svc
- Additional Optional APIs
  - Crypto
  - SIP
  - OpenGL ES 1.1
  - Contactless Comm
  - XML UI

Target Devices

- CLDC
  - 1Mb static memory
  - 1Mb dynamic memory
- CDC
  - 4Mb static memory
  - 2Mb dynamic memory

* Profiles and Application memory are in addition

CLDC Packages (Common)

- java.lang [subset]
- java.lang.ref [subset]
- java.io [subset]
- java.util [subset]
- java.io [subset]
- java.security [subset]
- java.nio
- javax.microedition.io

CDC Packages

- java.io
- java.lang
- java.lang.ref
- java.lang.reflect
- java.lang.annotation
- java.math
- java.net
- java.nio
- java.nio.channels.*
- java.nio.charset.*
- java.text.*
- javax.microedition.io
- java.security
- java.security.cert
- java.security.interfaces
- java.security.acl
- java.security.spec
- java.util
- java.util.jar
- java.util.zip
- java.util.spi
- java.util.regex
- java.util.concurrent.*
- java.util.logging
- javax.security.auth.x500