Java. Cloud. Leadership.

Is Enterprise Java Ready for Mobile and Cloud?

Dr Mark Little
Red Hat, Inc.

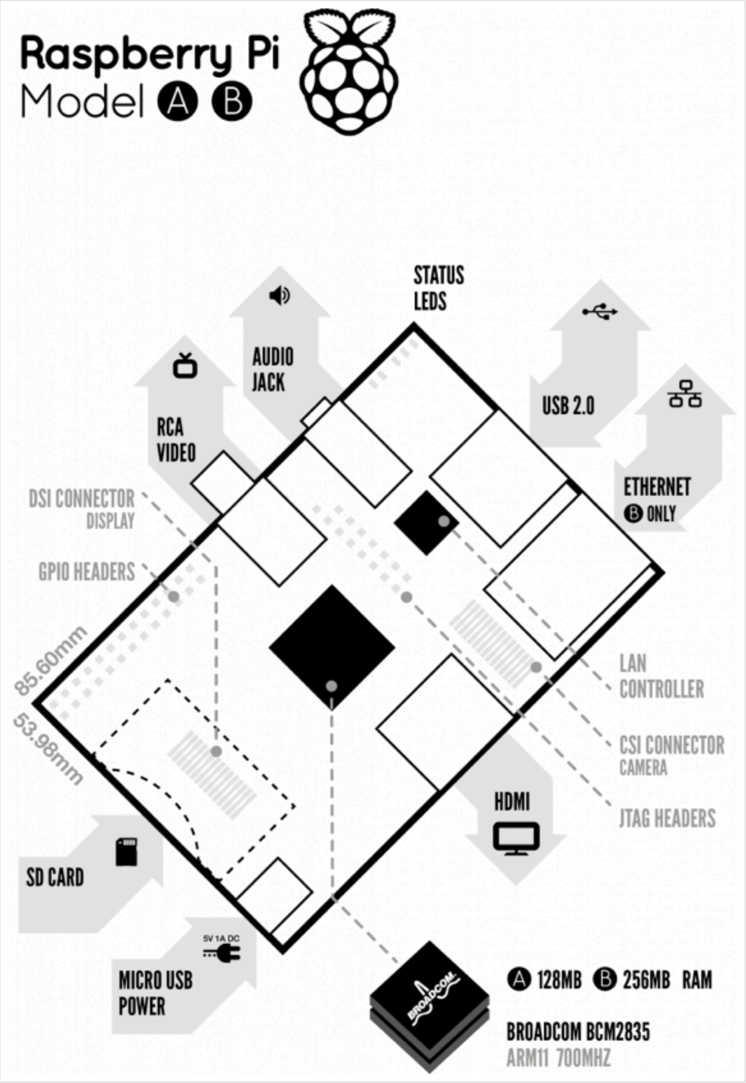
1/11/2011

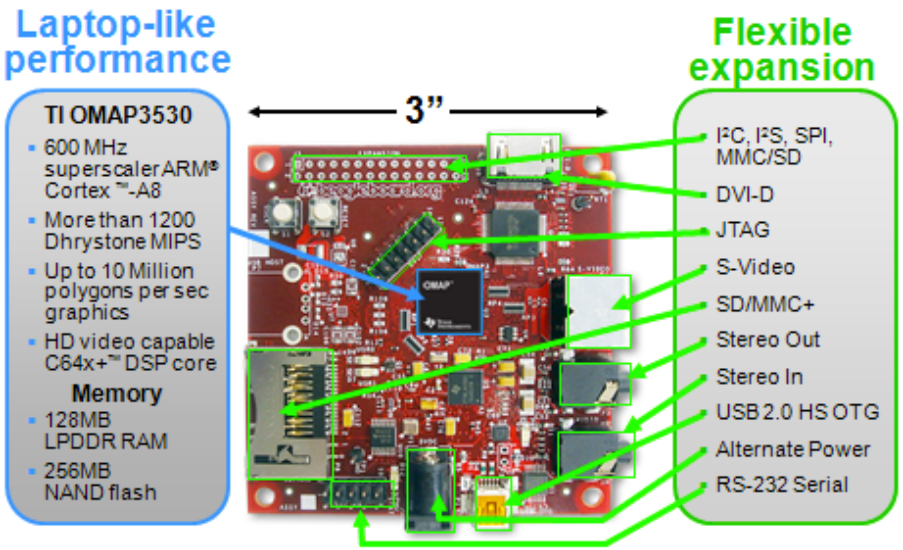
Overview

- Where are we today and why?
- Mobile and Cloud
  - Ubiquitous computing in the large
- Hardware and software forces in action
- What does this mean for today's middleware offerings?
- The future of Java and standards

The times have changed

- There are already more mobile devices than computers
- There are 4x more processors on the planet than people
  - Most have TCP stacks
    - dsPIC33FJ12GP 16-bit microcontroller has as much horsepower as a VAX (40MIPs), can handle 16+ sensors, and is 1/8 the size of a penny
  - 30 million iPads already
    - 1 in 2 Americans predicated to have smart phones by the end of 2011 compared to 1 in 10 in 2008





Laptop-like performance

TI OMAP3530

- 600 MHz superscaler ARM® Cortex ™-A8
- More than 1200 Dhrystone MIPS
- Up to 10 Million polygons per sec graphics
- HD video capable C64x+™ DSP core

Memory

- 128MB LPDDR RAM
- 256MB NAND flash

3”

Flexible expansion

- I^2C, I^2S, SPI, MMC/SD
- DVI-D
- JTAG
- S-Video
- SD/MMC+
- Stereo Out
- Stereo In
- USB 2.0 HS OTG
- Alternate Power
- RS-232 Serial

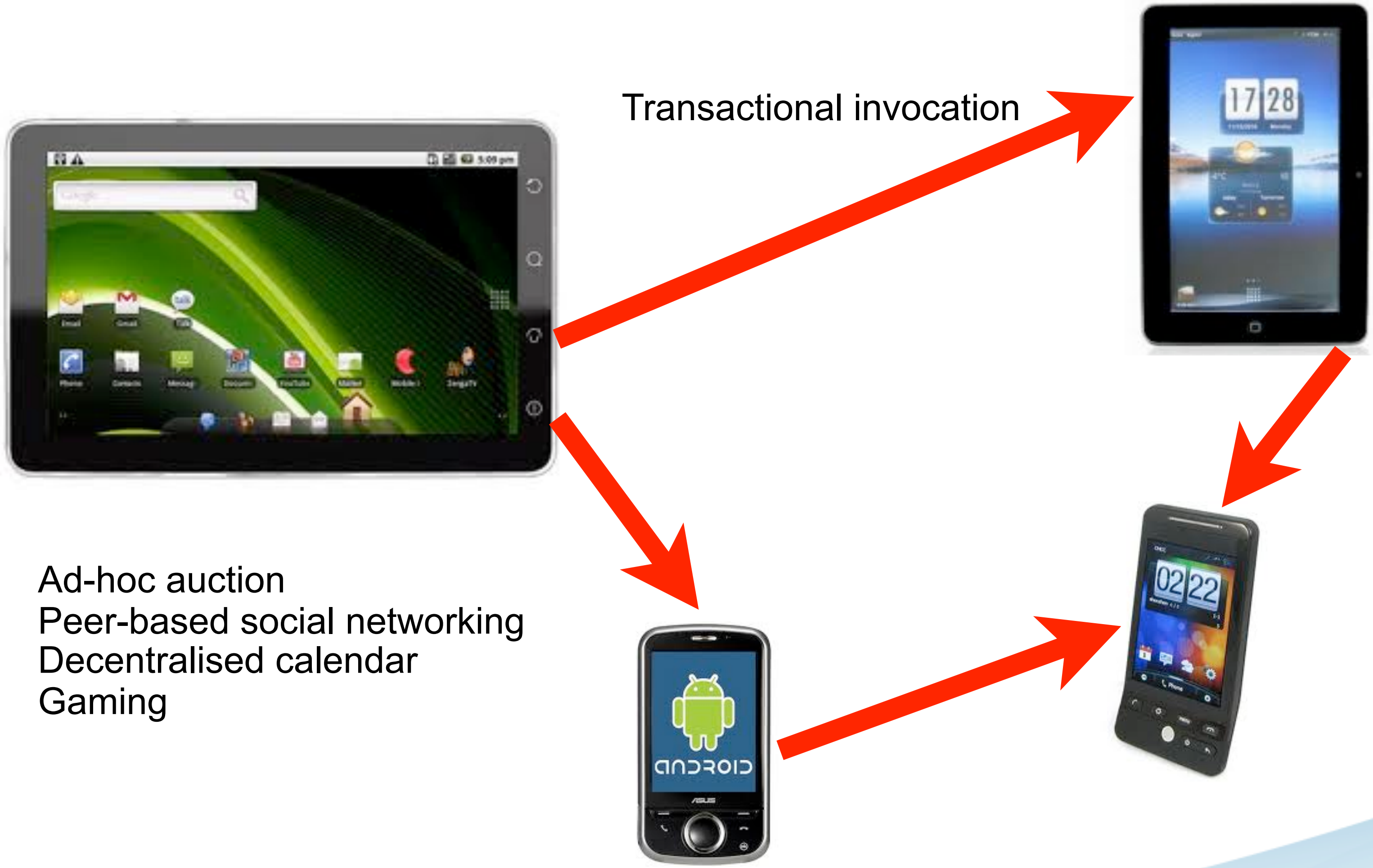
“Little’s law” (thanks to Parkinson)

- “Work expands to use the power available”
  - Basic word processors on first PCs
    - Publisher-quality implementations now on laptops
  - Games pushing the envelope from Pong through Space Invaders to CoD
    - Distributed systems
    - Grids
  - Mobile devices contain more and more personal data
    - Wallets via NFC
  - Disconnected operation is the normal situation

Application complexity

- Types of applications increasing in complexity
  - Online purchases
  - Distributed peer-to-peer interactions
- Enterprise requirements becoming a necessity
  - Security and identity
  - High performance, low latency, reliable messaging
  - Database updates with transactions
  - Workflows as inter-app interactions increase
- But mobile is critical for another wave ...

Mobility and “enterprise” apps



Transactional invocation

Ad-hoc auction
Peer-based social networking
Decentralised calendar
Gaming

Mobile displaces consoles

Cloud meets mobile

- Public Clouds important
- Private Clouds probably more important
- But Ubiquitous Computing has become a reality
- Mobility and embedded devices are the Cloud
- Thin clients are not sufficient
    - Shannon’s Limit - overlooked but we all know about it

Ubiquitous computing

Smartphones and Tablets

- Smartphones and tablet shipments overtaking PCs
- Multi-device support the reality

Field Service / Warehouse Devices

- Primarily Windows based
- Application tied to device and OS

Embedded / Unattended

- New use cases for embedded processors / computing

“New age” development

- New architectures
- New implementations
- New frameworks
- New operating systems
- New new new ..?

For whom the bell tolls? Middleware?

- Hold on ... haven’t we heard this death knell before?
- Remember Web Services/REST/ <blank> as death of middleware?
- Let’s be objective
  - No hidden agendas please

Some computing realities

- Trust is important!
- Trust is measured in:
  - Who is providing the service?
    - And are they doing it in a way that matches my requirements?
  - Are they living up to my required QoS
    - Fault tolerance, performance, etc.
- Several well publicised Cloud outages and intrusions
  - Mobile viruses, identity theft ...

So what does this mean?

- Middleware is needed whatever the deployment environment
  - Mainframes, servers, laptops etc.
- Don’t tie the definition of middleware to an implementation
- Mobile and Cloud should not be new silos for developers!
  - It’s inefficient
  - It’s expensive
  - It’s time consuming

So are cloud/mobile the death of middleware?

- Many commonalities with “traditional” middleware
  - Enterprise requirements for all but trivial apps
    - Messaging
    - Transactions
    - Security
    - ...
- Obviously Java is not the only application language
  - But why not a common runtime?
  - Lots of popular JVM-based languages

Java Enterprise Edition 6

- Turns out that EE6 has many of the required capabilities
    - Standards based too!
- EE6 represents a great evolution for 40 years of work!
- New capabilities (e.g., JAX-RS, CDI, BeanValidation)
- Input from wider open source communities and users
- Profiles

“Java EE is too bloated”

- Differentiate the standard from implementations!
  - Let’s not live in the past
- It is possible to be lightweight and enterprise ready

The Open Source Java application server *reignited*

Designed for flexibility.
Amped with electrifying speed.
Launch your Java EE applications in a flash!

Lightning Fast... *start-up / deployment / configuration*

Standards

- JCP reinvigorated
- Java ME
  - Merger of Java ME EC with SE/EE
  - ME.next?
- Java PaaS
  - Various non-standard solutions
  - OSGi, OMG, JCP
- Open source helps drive experience
  - Don’t standardise too early!

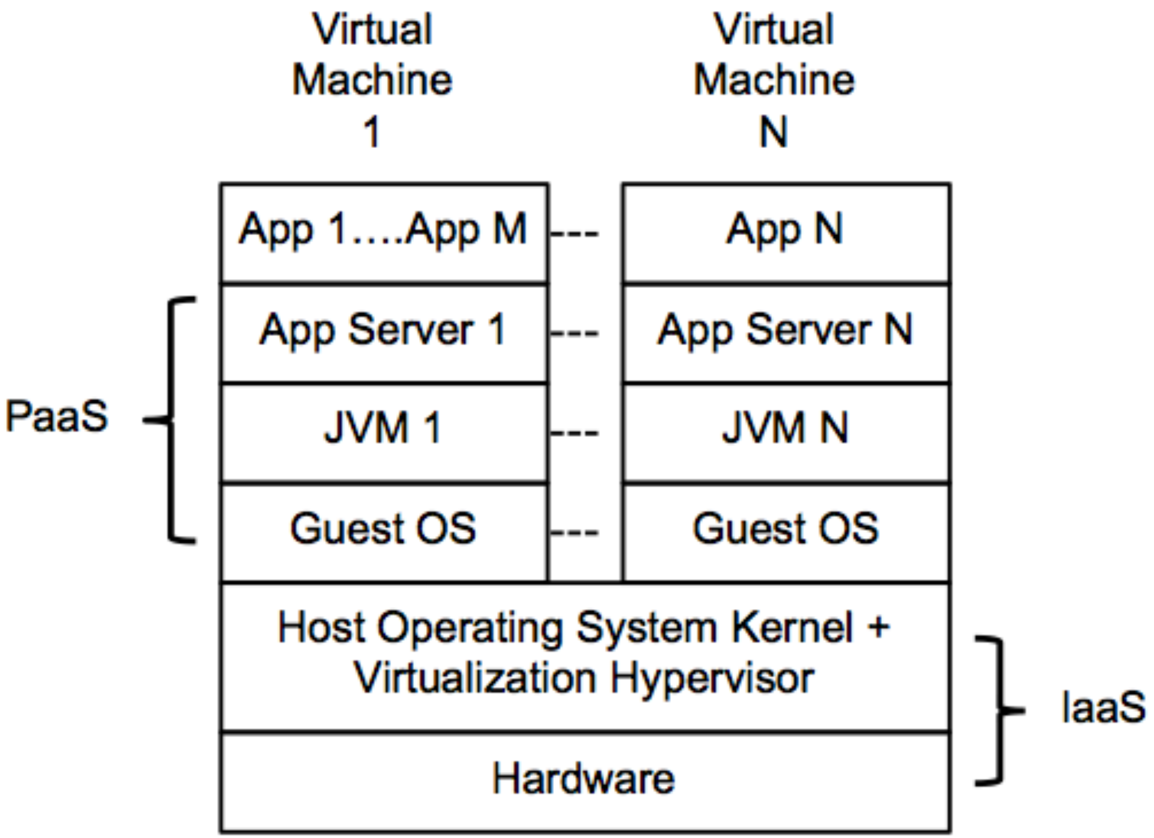
But there are still open areas

- The next generation of middleware focus ...
  - Coordination
  - ***Resource Sharing***
  - ***Replication in the large***
  - Negotiation and enforcement of SLAs
  - Accountability
- We will look at 2 of these today

Resource sharing

- Virtualisation has driven Cloud through IaaS and now PaaS
- PaaS requires high density applications
    - Multiple applications running in each VM instance
- Modern middleware stacks are typically not architected for multi-tenancy
    - JVM and application server run one application at a time
    - We forgot some of the good things we learnt in OS classes!

Middleware stack



Virtual Machine 1

Virtual Machine N

App 1….App M --- App N

App Server 1 --- App Server N

JVM 1 --- JVM N

Guest OS --- Guest OS

Host Operating System Kernel + Virtualization Hypervisor

Hardware

PaaS

IaaS

What’s needed?

- The JVM is here to stay
    - More languages moving to it due to ecosystem
- It needs to become the operating system
    - Multiple processes per JVM
    - Red-zone protected
        - Independent failures
    - Real-time
    - Deterministic scheduling
        - Garbage collector issues
- Dalvik does a good job but ...

Replication in the large

- RDBMS are great generalist solutions
  - ACID transactions
  - Strong consistency
  - Replication to reduce single point of failure
- But they were not designed for large scale data
  - Quantity
  - Distributed
- Resulted in NoSQL and NewSQL efforts
  - And OldSQL

NoSQL

- Weak consistency replication
  - Domains of strong consistency
- Gossip protocols for disseminating updates
- Typical (topical?) to ignore transactions
  - CAP theorem cited but sometimes an excuse
- But weak consistency requires applications to be aware
  - Complicated for developers
  - Requirements may change over time

What's needed?

- RDBMS and SQL aren't going away
  - But a rearchitecture can improve their applicability
- Transactions or extended transactions in NoSQL
  - Consistency is important for many applications
  - Relaxing ACID properties may help
- Explore the trade-offs between strong and weak consistency
  - Cacheing is becoming the new primary data store
  - Impact on applications, new frameworks, design guidelines

Richard Hamming, 1968 Turing speech

- Whereas Newton could say, "If I have seen a little farther than others, it is because I have stood on the shoulders of giants," I am forced to say, "Today we stand on each other's feet." Perhaps the central problem we face in all of computer science is how we are to get to the situation where we build on top of the work of others rather than redoing so much of it in a trivially different way.

Conclusions

- Cloud and mobile will evolve
- Enterprise middleware applications aren't going away
    - The industry cannot afford to track multiple platforms
    - Middleware components should be available to all
- The next decade will be defined by ubiquitous computing
- There are still areas that need to be addressed