Enterprise Middleware for the 21st Century

Professor Mark Little

Red Hat

Overview

- **Where are we today and why?**
- **Mobile and Cloud**
  - Ubiquitous computing in the large
- **Hardware and software forces in action**
- **Let's avoid reinventing the wheel yet again**
  - Though we may need to invent some spokes
- **The death of middleware or an evolutionary jump?**
- **What does this mean for today's middleware offerings?**
  - Is Java EE dead?
  - Can open source help?
- **JBoss plans**

A Brief History of Distributed Systems

- **Distributed systems are essentially the same today as 20 years ago**
  - The development model has not really changed
- **Message passing-to-RPC-to-Objects-to-Components-to-Services-to-…**
  - Living in the shadow of the past
- **The underlying enterprise requirements have not changed**
  - Messaging, transactions, security, …
- **Many successful years ahead providing current infrastructures …**

But … 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
  - Ubiquitous computing is here to stay
  - 20 million iPads already
  - 1 in 2 Americans predicated to have smart phones by the end of 2011 compared to 1 in 10 in 2008
- **Multi-core technologies will only increase**
- **Machine-to-Machine communication will dominate all other traffic**

Thin clients or rich clients?

- **Thin clients have certain advantages**
  - Easier to manage, less to go wrong
  - More scalable architectures
  - Rely on “good enough” connectivity or periodic connectedness
- **Rich clients have certain advantages**
  - Your data is where you need it, when you need it
  - Able to execute a wider range of applications
  - Can run disconnected and be useful for a long period of time
- **Thin clients are typically the domain of constrained devices**
  - But the definition of *constrained* is relative

30 years ago …

- 16K was considered a lot of memory
- 140K floppy disks were the standard
- 10 mbps ethernet was decadent
- 8 bit 6502 processor was king for personal computing
- Wireless was what people listened to when there was nothing on TV

Today …

- 512Meg memory is standard on smart phones, 64Gig storage
- 256Gig USB sticks are becoming the norm
- 100Gig ethernet at work and 30mbps to the home
- 64 bit quad core processors in laptops, 1GHz ARM in iPhone
- WiFi throughout many cities

Laptops and Linux and Java, Oh my!

- **The laptop concept originally devised for children**
  - Dynabook in 1968
  - Initially considered for word processing and learning
- **Now laptops replace desktop**
  - More power than early mainframes
- **Arguably Java and Linux helped to popularise middleware**
  - Drive it to the masses
- **But Java is not cool any more**
  - Ruby, Scala, Erlang, JavaScript, …
- **Coolness is iPhones, Androids, HTML5, …**
- **Those are the new frontiers of application development**

“Non-middleware” devices

- **Basic word processors on PCs**
  - Publisher-quality implementations now on laptops
- **Basic doc readers on smart phones**
  - Editors can’t be far off
- **Games pushing the envelope from Pong through Space Invaders to CoD**
  - MVCC
  - Distributed systems
  - Grids
    - PSP, Nintendo
- **Mobile devices contain more and more personal data**
  - Wallets via NFC
  - Documents, photos, etc.
- **Disconnected operation is the normal situation**
  - C.f. 1980’s research

Application requirements

- **Types of applications increasing in complexity**
  - Auctions
  - 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

What about Cloud?

- **One (virtual) computer system that is able to …**
  - Cope with arbitrary loads and applications
    - "Run my applications securely when I want you to."
  - Managed by someone else
  - Secure
  - Reliable
  - Cheap (cost effective)
- **"Emulate the system I would deploy locally if I could afford it."**

Different flavours of Cloud?

- **Public Clouds important**
- **Private Clouds probably more important**
  - Security and data consistency implications
- **But *Ubiquitous Cloud* may become a reality**
  - Many devices are predicted to have processors/sensors in the future
  - It will be cheaper to give them “high end”/off-the-shelf processors/memory than bespoke dye-factories
    - In the next few years your light bulb or washing machine may be as powerful and capable as your laptop today
  - Tapping into this local cloud could be possible
    - Whether intended originally or not!

Cloud = death of middleware?

- **<sarcasm>Yeah, right!</sarcasm>**
- **Many commonalities between "traditional" middleware and PaaS**
- **In the Java world extending EE would suffice for many applications**
  - Virtual middleware stack
  - Add billing
- **But lack of standards for Cloud means Wild West**
  - But it doesn't have to be this way!

Revolution?

- **Many existing applications want to be cloud-enabled**
- **Should not require reimplementing existing infrastructual investments**
  - Databases
    - NoSQL versus RDBMS?
    - Use the right tool for the right job
- **The industry has spent 40+ years designing enterprise infrastructures**
  - Many of them work well!
- **However, middleware today is often not designed for Cloud**
  - Large scale
  - Autonomous
- **But things are common, or similar enough**
  - Security
  - Transactions
- **But new capabilities and approaches are required**
  - Much 80s and 90s research is now applicable

Present and future Cloud

- **Build on existing implementations where possible**
- **We must provide a natural upgrade path for existing users**
  - We cannot afford to repeat the DCE/CORBA, DCOM/.NET or CORBA/J(2)EE days
- **If the answer is "Cloud 2011" the question is wrong!**
  - Today's Cloud is not the final solution, it's just another step
- **Today "Cloud" means "servers"**
  - It should be wider

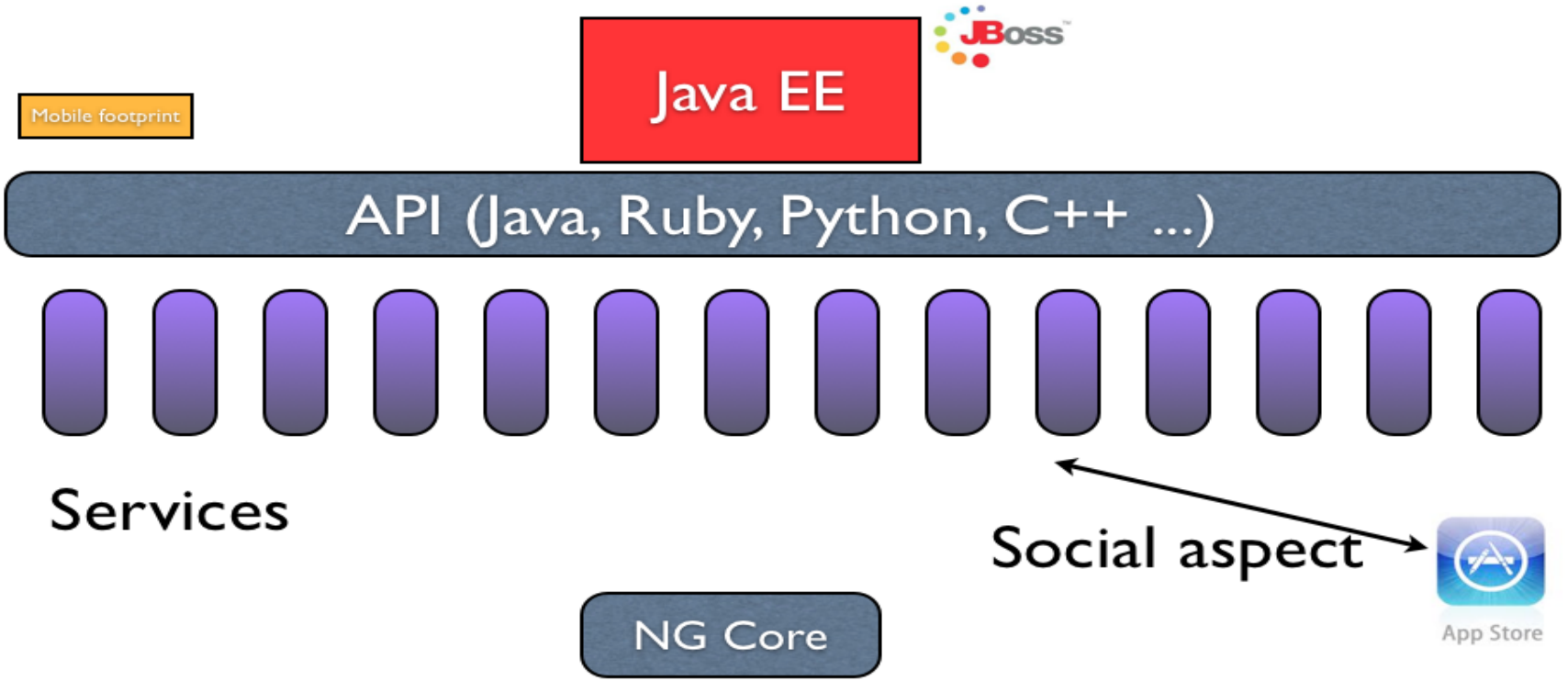
So what does this all mean?

- **Middleware is needed whatever the deployment environment**
  - Mainframes, servers, laptops etc.
  - HTML5
  - Constrained devices? Huh?
- **Don't tie the definition of middleware to an implementation**
  - "Middleware is that s/w which sits between the OS and the application."
- **But mobile and cloud are potentially new silos for developers to work within!**
- **Middleware elitism?**
  - Mainframe versus mini versus desktop versus laptop?
- **Enterprise requirements transcend deployment realities**

Middleware for tomorrow

- **Stop designing just for today or yesterday**
- **Flexible**
  - Different environments (not all Java)
  - Different component implementations
  - Cannot assume a single stack
    - Or that users will want the full stack from any single vendor
    - Mix-n-match
- **Adaptable**
  - Dynamic and static
  - Applications could migrate between environments
- **Reliable**
- **Securable**
- **Available**
- **Scalable**

For example …


Java EE
Mobile footprint
API (Java, Ruby, Python, C++ ...)
Services
Social aspect
App Store
NG Core

The open source approach?

- **Open source makes middleware available to everyone**
- **Open source has many of the technologies to help developers on a range of platforms**
  - HornetQ, Infinispan, Camel, RESTeasy, Seam, …
- **This is more like building a new jigsaw puzzle from the same pieces**
  - And incorporating existing completed jigsaws!
- **We need to facilitate approaches that build on what we have already**
  - No more reinventing the wheel
    - Sharing of experiences as well as code
  - Makes it easier to transition, understand etc.
    - Easier for others to get involved

Conclusions

- **Cloud will increase and evolve**
  - Already evolved from Grid and ubiquitous computing
- **Mobile is going to impact far more people than Cloud**
- **Enterprise middleware applications aren't going away**
- **The industry cannot afford to track multiple platforms**
- **Middleware components should be available to all developers**
  - Maybe even adaptable middleware stacks
- **The network connectivity laptops have today should be the vision for applications, wherever they reside**
- **The next decade will be defined by the mobile generation**
- **And what about JBoss ..?**