redhat.

Java. Cloud. Leadership.

JavaEE.next: 8 and Beyond

Dr Mark Little, Red Hat

What's past is prologue (The Tempest)

- Java EE has become mainstream
- EE6 has simplified development of complex applications
- But EE moves slowly
  - Many standards suffer from this
- New frameworks, stacks, etc.
  - Most driven by open source
  - Mobile, cloud, ...
- How can EE remain relevant?
  - Should it even try?

No fate but what we make

- “The future, always so clear to me, has become like a black highway at night. We were in uncharted territory now... making up history as we went along.” Sarah Connor, Terminator 2

Beyond EE8

- Mobile distinction is becoming arbitrary
- Asynchronous frameworks and stacks
- More modular profiles
  - Web profile was a good addition
  - Cloud profile?
  - Mobile profile?
- Further ease of use
  - CDI was the defining improvement in EE6

Asynchronous programming

- Node.js has popularised async THIS TIME
  - Event driven only a few years previously
- But async has been around for a long time
  - Initial distributed systems were async
  - Prior to threading, it was often the only way to work
- Async seen as superior for certain problems
  - C10K
- However, async presents its own problems
  - Failure detection
  - Cannot reach consensus

Problems with parallelism

- We have yet to make parallel programming easy
  - Only automatic parallelism has “won”
    - Parallel decision support databases
  - Managing computer clusters is a major cost
- New computer architectures are highly parallel
  - 1000 instruction steams per chip are coming
  - Programming biological systems: each cell is a system
- The scale-up problem is not solved

Polyglot

Before 2010
1 JVM, 1 Language

JVM

Today
1 JVM, Over 20 Languages

JVM

redhat. | JBoss by Red Hat

Java. Cloud. Leadership.

New language requirements

- Customers and community wants:
  - Interoperability
  - Guaranteed message delivery
    - Even in the presence of failures
  - Transactions
    - Though not necessarily ACID
  - Audit trails and bullet-proof security
  - Machine-readable SLAs
- N-tier approach with different languages

Enterprise capabilities

- Java in 1996 did not possess enterprise features
  - J2EE took several years to evolve
    - Some implementations layered on existing services
- Popular JVM languages experiencing similar problem
  - Lack of enterprise capabilities
- Two ways to resolve
  - Build from scratch in language
  - Leverage existing implementations in other languages

Middleware and IoT

- Oracle officially announced OSGi is their baseline
  - RFC 192: ZigBee Device Service
  - RFC 196: Device Abstraction Layer
  - RFC 199: EnOcean Device Service Specification
  - RFC 200: Resource Management
  - RFC 202: USB Device Category
  - RFC 209: Network Interface Information Service
- Eclipse main foundation doing IoT
  - liblwm2m
  - Californium (CoAP)
- Messaging is key component
  - MQTT
  - AMQP

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

What does this mean for Java EE?

- Don’t fall into the trap of trying to make Java EE fit for everyone
  - It will become fit for no one
- Profiles may be insufficient
- What about something entirely different and bespoke within the JCP?
  - May be too early to standardise these things
  - When time does come to standardise it may be that the JCP isn't even the right body for the work

Learn from other efforts

- Maybe Java isn't even the right language for some of these things
  - In which case we need to be cautious about rewriting everything from scratch
  - Remote access to key services and capabilities written in Java (running at the "back end") may offer a much more reliable and mature approach than starting from nothing and reimplementing

Java is the future?

- There are a plethora of new and old languages in use today
  - Java is being challenged
- Hear that these new languages aren't enterprise ready so we could ignore them and tell everyone that Java is the right language for enterprises
- Some developers who are using JVM languages may agree, but most will not want to touch Java or have the time or skills to do so.
- Therefore, if we do nothing they will reimplement from scratch. Not only will that be a waste of their time but it will signal the decline of Java
- We need to assist these new developers. Help them, help Java. It can and should remain relevant

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.