Enterprise Java in 2011 and Beyond

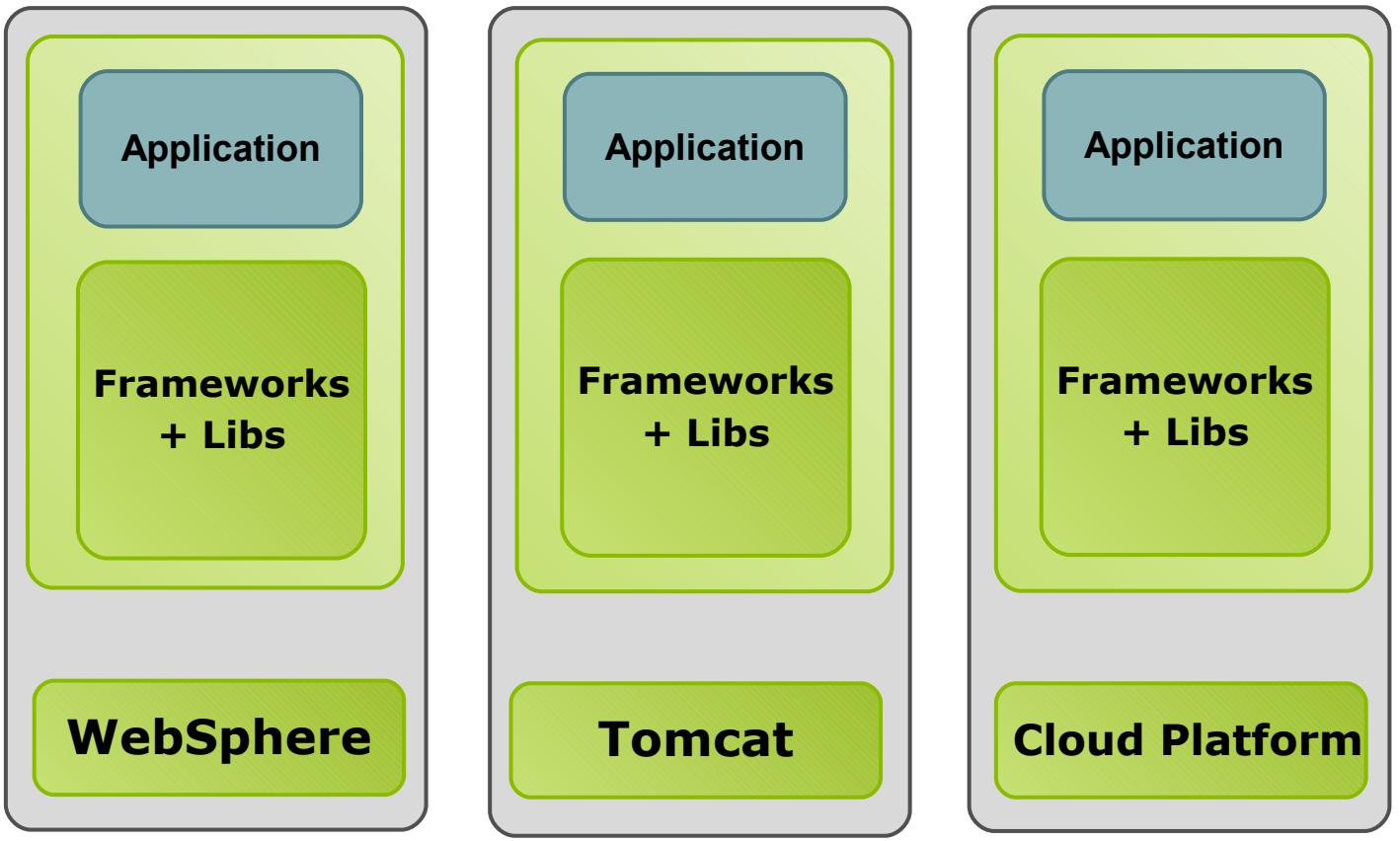
From Java EE 6 To Cloud Computing – A Spring Perspective

Jürgen Höller, Principal Engineer, SpringSource

springsource
A division of vmware

CONFIDENTIAL

Deployment Platforms: Becoming More Diverse



The State of Deployment Platforms in 2011

- **Java EE moving on to Java EE 6**
  - GlassFish 3
  - JBoss 6
  - Other servers still on Java EE 5 (at best)
- **Tomcat moving on to Tomcat 7**
  - Servlet 3.0 based (Java EE 6 level)
- **Cloud platforms becoming a serious option for regular Java web application deployment**
  - Google App Engine: Jetty++
  - Amazon Elastic Beanstalk: Tomcat++

Java EE 6

- **Several interesting specifications**
  - Servlet 3.0
  - JAX-RS 1.1
  - JSF 2.0
  - JPA 2.0
  - EJB 3.1
  - CDI 1.0
  - Bean Validation 1.0

- **But: typically not used as a bundle yet**
  - no mainstream EE 6 servers with enterprise support available yet
  - instead: individual specifications added to Tomcat or to EE 5 servers

Cloud Platforms

- **a.k.a. „Platform as a Service“ (PaaS)**
  - „public cloud“: available through a shared, public host
  - „private cloud“: virtualization platform inside a corporate data center
- **Typically: a pre-installed web container with additional services**
  - datastores (not necessarily a relational database!)
  - messaging, clustered sessions, clustered cache, etc
- **The aforementioned Google App Engine and Amazon Elastic Beanstalk are great examples**
  - common ground: WAR deployment, Servlet API, JPA – ignoring Java EE
  - several further offerings to show their promise in the course of this year

Wide Variety of Data and Datastores

- **Not all data resides in relational databases**
  - cloud environments often suggest alternatives for scalability reasons
  - BigTable, Redis, Mongo, etc
- **Distributed caches add challenges as well**
  - not least of it all in terms of application-level access patterns
  - GemFire, Coherence, etc
- **Hardly any standardization available**
  - just an abandoned caching JSR that never achieved a final release
  - caching – but only caching – possibly getting picked up in Java EE 7
  - alternative datastore space is too diverse

Wide Variety of Web Clients

- **More and more client-side web technologies**
  - HTML 5 as a next-generation browser standard
  - Adobe Flex as a rich client technology on the basis of Flash
- **Server-side state to be minimized or even removed completely**
  - in particular: no server-side user interface state
  - strictly controlled user session state
  - state management for event-driven architectures
- **JSF's state-centric approach not too desirable anymore**
  - except for special kinds of applications (which it remains very useful for)
  - general web applications and web services based on JAX-RS / MVC style

Java SE 7: Concurrent Programming

- **A challenge: concurrent programming in a multi-core world**
  - user-level APIs and recommended programming styles?
- **Servers with more cores than concurrent requests**
  - how to actually use your processor power in such a scenario?
- **Java SE 7: java.util.concurrent.ForkJoinPool**
  - specialized ForkJoinPools to be locally embedded within the application
  - different kind of pool, separate from regular Runnable-oriented Executors
- **Oracle JDK 7 scheduled for GA release in Q3 2011**

Scala & Akka: Concurrent Programming Revisited

- **Scala as a next-generation language on the JVM**
  - combines object orientation with functional programming
  - particularly well suited for concurrent programming
  - integrates well with existing Java APIs

- **Akka as an actor-based framework for Scala and Java**
  - event-driven architectures
  - different approach towards concurrent programming
  - raises the concurrency abstraction level (but not too much)
  - provides Scala and Java APIs
  - most convenient with Scala message passing

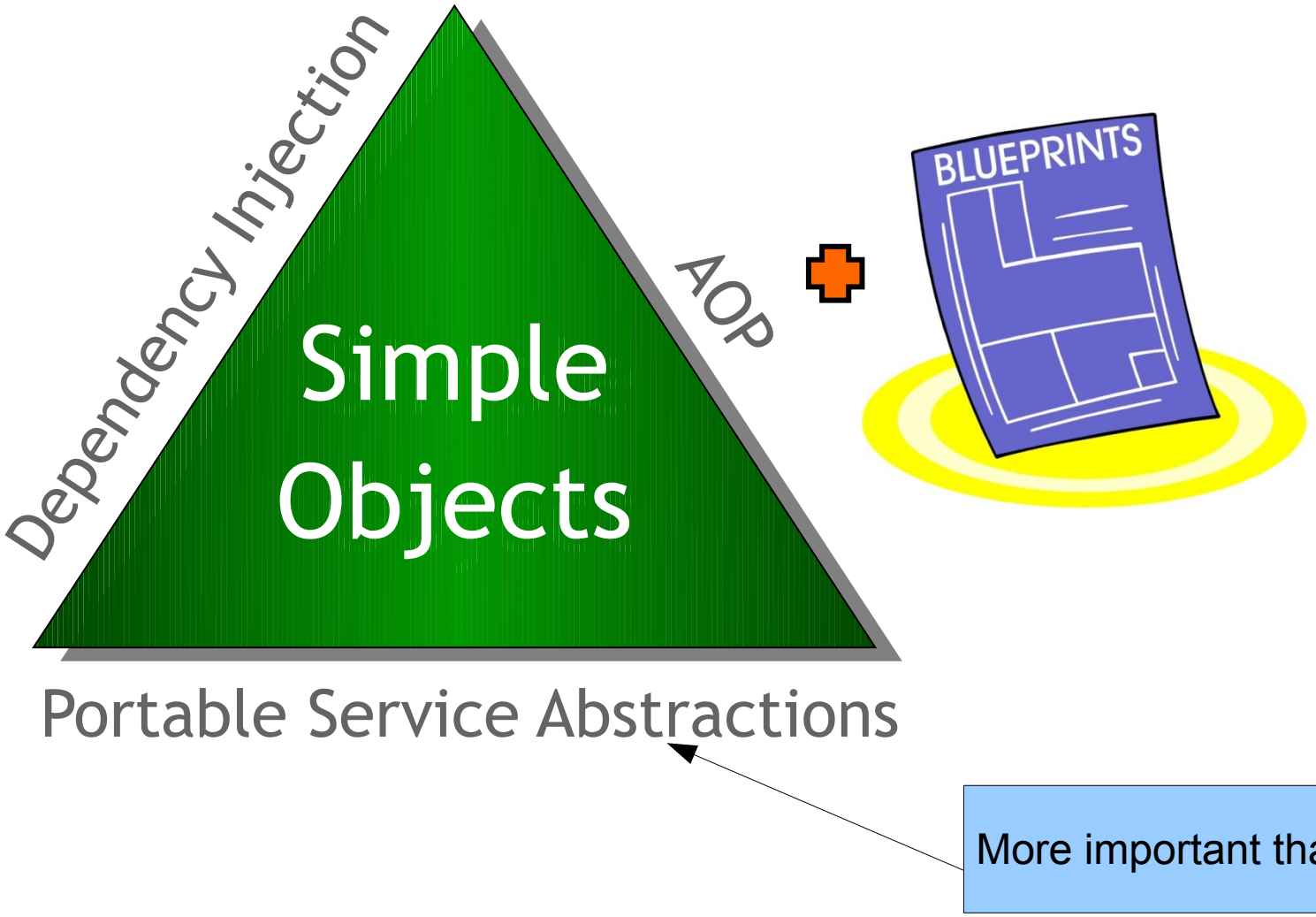
Java EE 6 Revisited

- **How relevant is Java EE 6 in the context of recent trends?**
  - as usual, Java EE 6 tends to solve yesterday's problems
  - the fate of specifications with a multi-year expert group process
  - even worse, EE server vendors take years to implement a full platform release

- **Some recent trends change this industry quite rapidly and radically**
  - cloud platforms challenge the notion of dedicated servers
  - alternative datastores challenge relational databases and their access APIs
  - concurrent programming trends do not match traditional EE assumptions

- **Java EE 7 to the rescue?**
  - let's see – once it gets released in 2013...

Key Elements of Spring: Ready for 2011 & Beyond



Spring Framework 3.1 & 3.2: Key Themes

- Addressing previously discussed industry trends...

- **Spring Framework 3.1**
  - Environment profiles for bean definitions
  - Cache abstraction & declarative caching
  - Conversation management
  - Servlet 3.0 based web application deployment

- **Spring Framework 3.2**
  - Comprehensive support for Java SE 7
  - Preparing for language enhancements in Java SE 8

Environment Abstraction

- **Grouping bean definitions for activation in specific environments**
  - e.g. development, testing, production
  - possibly different deployment environments
  - custom resolution of placeholders
- **Environment association of specific bean definitions**
  - XML 'profile' attribute on <beans> element
  - @Profile annotation on configuration classes
  - @Profile annotation on individual component classes
- **Ideally: no need to touch deployment unit across different stages/environments**

Environment Example

```
<beans profile="production">
  <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
        destroy-method="close">
    <property name="driverClass" value="${database.driver}"/>
    <property name="jdbcUrl" value="${database.url}"/>
    <property name="username" value="${database.username}"/>
    <property name="password" value="${database.password}"/>
  </bean>
</beans>

<beans profile="embedded">
  <jdbc:embedded-database id="dataSource" type="H2">
    <jdbc:script location="/WEB-INF/database/schema-member.sql"/>
    <jdbc:script location="/WEB-INF/database/schema-activity.sql"/>
    <jdbc:script location="/WEB-INF/database/schema-event.sql"/>
    <jdbc:script location="/WEB-INF/database/data.sql"/>
  </jdbc:embedded-database>
</beans>
```

Cache Abstraction

- **CacheManager and Cache abstraction**
  - in org.springframework.cache
    - which up until 3.0 just contained EhCache support
  - particularly important with the rise of distributed caching
    - not least of it all: in cloud environments

- **Backend adapters for EhCache, GemFire, Coherence, etc**
  - EhCache adapter to be shipped with Spring core
  - plugging in custom adapters if necessary

- **Specific cache setup per environment profile?**
  - potentially even adapting to a runtime-provided service

Declarative Caching

```
@Cacheable
public Owner loadOwner(int id);


@Cacheable(condition="name.length < 10")
public Owner loadOwner(String name);


@CacheEvict
public void deleteOwner(int id);
```

Conversation Management

- **Abstraction for conversational state**
  - basically HttpSession++
  - more flexible lifecycle
  - more flexible storage options
- **Management of a current conversation**
  - e.g. associated with browser window/tab
  - or manually demarcated
- **For use with MVC and JSF as well as messaging**
  - conversation identified by request parameter, message header, etc
  - generalized 'conversation' scope for scoped beans
  - programmatic access at any time

Servlet 3.0: No web.xml anymore?

- **Explicit support for Servlet 3.0 containers**
  - such as Tomcat 7 and GlassFish 3
  - bootstrapping Spring-based web applications in new ways
- **Support for XML-free web application setup (no web.xml)**
  - Servlet 3.0's ServletContainerInitializer in combination with Spring 3.1's AnnotationConfigWebApplicationContext plus the environment abstraction
  - delivers a nice overall experience
- **Additionally: support for asynchronous request processing**
  - Servlet 3.0's startAsync facility
  - primarily as a foundation for special-purpose frameworks

Support for Java SE 7 & 8

- **Java 7 is an important driver for Spring**
  - making best use of JRE 7 at runtime
  - support for JDBC 4.1
  - support for concurrent programming on top of the fork-join framework
  - blueprints for fork-join use in Spring-based applications

- **Java 8's language enhancements in mind already**
  - preparing Spring APIs for Java 8 closures
  - „Single Abstract Method“ (SAM) types: interfaces with one method
  - common in Spring already: ResultSetExtractor, HibernateCallback, etc
  - Java 8 language enhancements will be immediately available to existing versions of Spring once JDK 8 is released

Related Spring Portfolio Projects

- **Spring Integration**
  - practical enterprise integration patterns
  - generalized messaging infrastructure
  - closely integrated with Spring Framework core

- **Spring Data**
  - collection of support packages for alternative datastores
  - Redis, Mongo, Neo4j, etc
  - tracking the latest and most popular products in that space

- **Spring GemFire**
  - access to GemFire's distributed caching capabilities in Spring style

Summary

- **Disruptive forces approaching the Enterprise Java space**
  - deployment to cloud platforms
  - access to alternative datastores
  - concurrent programming challenges
- **Different speed of evolution**
  - slow-moving Java EE, with even slower adoption in data centers
  - fast-moving, fast-innovating, immediately available cloud platforms
- **Java SE 7 and Servlet 3.0 as common ground for years to come**
  - embedded application frameworks such as Spring are a perfect fit on top
  - embracing new programming models trends as they emerge
  - selected specifications as one key part, 'proprietary' APIs as another key part

QCon

杭州站 · 2011年10月20日~22日

www.qconhangzhou.com（6月启动）

QCon北京站官方网站和资料下载

www.qconbeijing.com

全球企业开发大会

THE ANNUAL INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE