PRESENTED BY RED HAT

LEARN. NETWORK.
EXPERIENCE OPEN SOURCE.

www.theredhatsummit.com

Throwing Complexity Over the Wall - Rapid Development for Enterprise Java

Andrew Lee Rubinger
Senior Software Engineer, JBoss by Red Hat
23 June 2010

Agenda

- Typical Problems with Enterprise Development
- Your Role as an Application Programmer
- Splitting Up Responsibility
- The Development Lifecycle
- Exciting New Tools to Help You Develop with Confidence

The Speakah

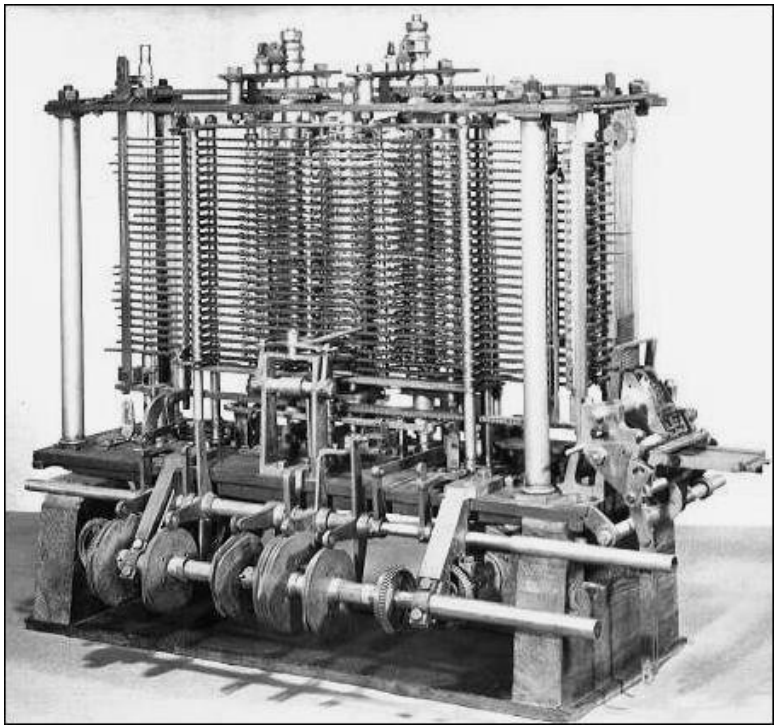
- Andrew Lee Rubinger
  - alr@jboss.org
  - Tweet, Dude: @ALRubinger
  - http://exitcondition.alrubinger.com
  - http://community.jboss.org/people/ALRubinger
- Core Developer at JBoss by Red Hat on EJB3 and Application Server; Project Lead of ShrinkWrap and Contributor to Arquillian
- Author of Upcoming “Enterprise JavaBeans 3.1 6th Edition” from O'Reilly Media – Due September
- Most importantly, longtime user of J2EE and JEE

Goals for Today

- Break down
- Examine the real problem domain we face as developers of Enterprise software
- Build up
- Introduce new Development Solutions
  - ShrinkWrap and Arquillian

Characteristics of Good Software

- Secure
- Sound / Maintains integrity
- Scalable
- Interoperable
- Robust / Resilient
- Correct / Functions as specified



Charles Babbage Analytical Engine

What's Important to Your Employer?

- Bottom Line
- Features
- Time-to-delivery
- Implicit assumption that everything is of high quality
- Not much tolerance for refactoring, stress-test scenarios, reducing “technical debt”[1], other routine maintenance

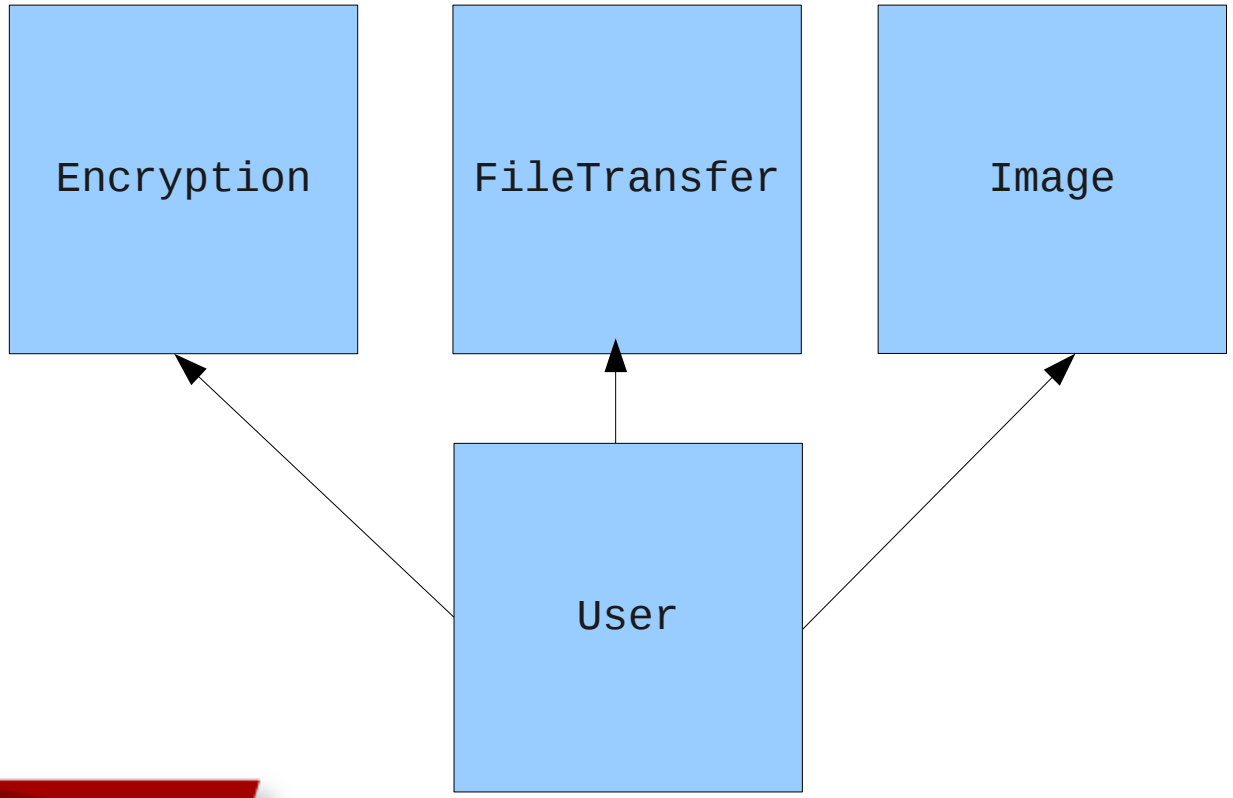
[1] Martin Fowler http://martinfowler.com/bliki/TechnicalDebt.html

Core Concerns

- Business Logic!
- This is what we're paid to do
- Domain-specific
- No one else can do this for us
- Time spent here is a Good Investment

Core Concerns as Modules

- Modules may have dependencies
- Encourage separation of concerns

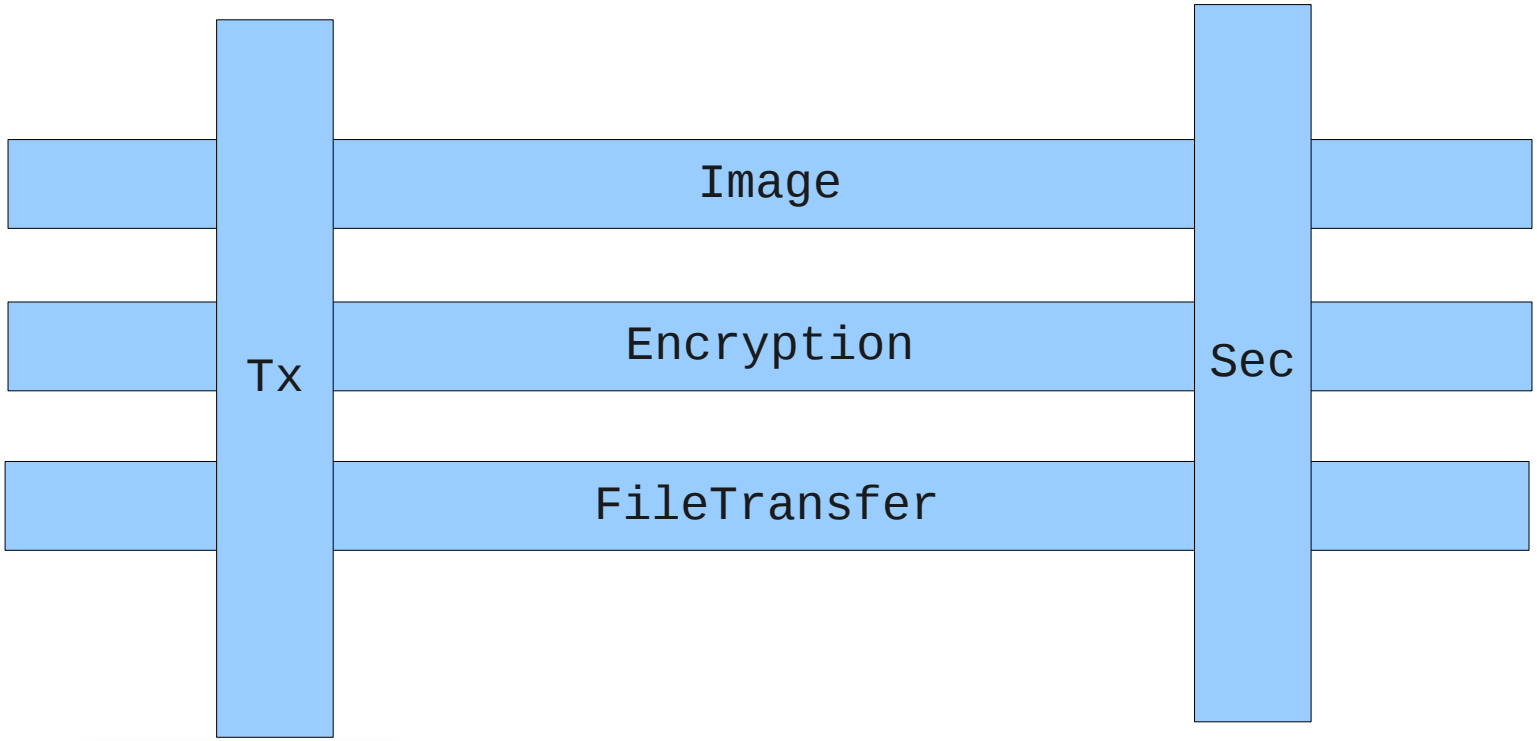


Cross-Cutting Concerns

- Generic
  - Independent of business logic
  - No compile-time dependencies on other components
- Targeted
  - Do one thing
- Composable
  - May be combined (or not) with others
  - Invisible to the core
- “Aspects” / “Advice”
- “Orthogonal” == Perpendicular

Cross-Cutting Concerns Modeled

- Aspects don't have dependencies upon each other or core modules
- May be applied consistently



Cross-Cutting Concerns Examples

- Servlet Filters
  - Page Compression
- Custom Aspects
- EJB Interceptors
  - Security
  - Transactions
  - Other Services

Plumbing

- Gets data from Point A to Point B
- Often handles transformation
- Does not affect state
- Enables loose coupling of disparate components
- Is generic
- Typically not the best use of an application developer's time

Plumbing Examples

- Servlets
  - HTTP Request > Object Model
- JAX-RS / RESTEasy
- Swing Event Dispatch
- Hibernate / JPA
  - Object Model > SQL
- Usually implemented by frameworks

So What Should You Code?

- Business Logic
- Core Concerns
- Leave the rest to Servers and Frameworks
- The less you write, the less you test and maintain
- Reduce “conceptual weight” - Joshua Bloch

Component Models as a Solution

- Run in a “Container”
- The Container manages the environment
  - Wires up the runtime
- User code executes as deployable components
- Components follow a standard form, or model

And What is Java EE?

- A unified collection of specifications which allow us to write business logic as components.
  - A recipe to write less
  - Increase your signal to noise ratio
- A platform defining how containers must handle wiring and services for us
  - Get powerful mechanisms for free

What's Been Missing?

- A cohesive way to develop and test our applications
- Applying the same component model paradigm to our tests
  - Let's not waste time on a custom test harness

The Importance of Testing – The Obvious

- Ensures your code works
- Future-proofs against maintenance breakages

The Importance of Testing – The Often Ignored

- Forces developers to be users
- Key in proving the API design makes sense
- Self-documenting
- Gives you a sustainable path forward
- Slims the release process
  - Testing along the way takes away the “big bang” rush at the end of a development cycle or iteration

“Testing” is not Necessarily “TDD”

- Test-Driven Development means “write the tests first”
- Absolute ideals
- Blocks iterative development

Excuses, Excuses

- Testing is not enjoyable
  - It should be!
- We're under pressure to deliver
  - Test code does not provide bottom-line benefit to the featureset
- Some folks even delude themselves into thinking that testing has no measurable benefit.

Unit Testing

- Finely-grained
- Tests an individual piece of work
  - Guideline: Single API call
- Speed is important
  - Run before commits
  - IDE integration is very helpful
- POJO Programming Model of Java EE
  - Already promotes Unit Testing!

Integration Testing

- Coarsely-grained
- Tests interaction involving many components
- May tie into the Agile notion of “user stories”

Traditional Integration Testing in Java EE

- Verbose or time-consuming pre/post lifecycle
- Usually involves bringing up a new runtime
- Example: JBossAS TestSuite
  - Ant tasks to launch new process
  - Deploy a JAR/WAR/EAR
  - Run the test over remoteable protocols
  - Bring down the house

In-JVM (Embedded) Integration Testing

- Pros
  - Rely on shared memory
  - Pass-by-reference
  - No need to expose/build remotable views
  - Manage concurrency
- Cons
  - Lack of isolation
  - JVM startup params may differ

A Hybrid Approach to Integration Testing

- Container in its own process
- Test is deployed as an archive
- Test runs inside the container
- TestRunner obtains the result remotely

Test Reliance Upon the Build

- Adds an extra step to the development/test cycle
- Packaging
  - Defines a unit / component
  - Regulates ClassLoading

Faking the Environment for Unit Tests

- Mock Objects
  - Stubs out APIs which may not be available
  - Gets you running in a POJO environment
  - http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/tru
  - http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/tru

Introducing ShrinkWrap

- ShrinkWrap provides a simple mechanism to assemble archives like JARs, WARs, and EARs in Java.


```
JavaArchive archive  =
  ShrinkWrap.create(JavaArchive.class,”archive.jar”)
  .addClasses(MyClass.class,MyOtherClass.class)
  .addResource("mystuff.properties");
```


Micro-Deployments

- Using ShrinkWrap to deploy components in isolation
- Test one thing at a time
- Don't rely on your full application to run an intermediate level of integration tests

ShrinkWrap Container Integration

- JBoss EmbeddedAS
  - Supports ShrinkWrap deployments natively
- OpenEJB
  - http://anonsvn.jboss.org/repos/common/shrinkwrap/trunk/ex
- Jetty
  - http://anonsvn.jboss.org/repos/common/shrinkwrap/trunk/ex
- GlassFish v3
  - http://anonsvn.jboss.org/repos/common/shrinkwrap/trunk/ex

ShrinkWrap Project Information

- Project Home: http://jboss.org/shrinkwrap
- Wiki: http://community.jboss.org/en/shrinkwrap
- Issue Tracker: https://jira.jboss.org/jira/browse/SHRINKWRAP
- User Forums: http://community.jboss.org/en/shrinkwrap?view=discussions
- Development Forums: http://community.jboss.org/en/shrinkwrap/dev?view=discussion
- Anonymous SVN: http://anonsvn.jboss.org/repos/common/shrinkwrap/trunk
- Committer SVN: https://svn.jboss.org/repos/common/shrinkwrap/trunk

Removing Plumbing from the Test Harness

- What would be left?
  - http://anonsvn.jboss.org/repos/common/shrinkwrap/trunk/ex

Introducing Arquillian

- The mission of the Arquillian project is to provide a simple test harness that developers can use to produce a broad range of integration tests for their Java applications (most likely enterprise applications).
- Abstracts out server lifecycle and deployment
- Write less code
- Tests become container non-specific

Arquillian Simple Example – Stateless EJB

- http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/tr

Arquillian Container Support

- JBoss EmbeddedAS
- JBoss Application Server (Remote) & EAP
- JBoss Embeddable EJB 3.1 (Roadmap)
- GlassFish v3 Embedded
- Jetty
- OpenEJB
- Weld / JSR-299 Contexts and Dependency Injection RI
- Other ideas?

Arquillian Project Information

- Main Space: http://jboss.org/arquillian
- Wiki: http://community.jboss.org/en/arquillian
- Issue Tracker: https://jira.jboss.org/jira/browse/ARQ
- User Forums: http://community.jboss.org/en/arquillian?view=discussions
- Development Forums: http://community.jboss.org/en/arquillian/dev?view=discussions
- Anonymous SVN: http://anonsvn.jboss.org/repos/common/arquillian/trunk
- Committer SVN: https://svn.jboss.org/repos/common/arquillian/trunk

Use Case: Calculator EJB

- Let's see how slim we can get these tests

```
/**
   * Adds all arguments
   *
   * @return The sum of all arguments
   */
  int add(int... arguments);
```


Use Case Example: Calculator EJB

- Business Interface
  - http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/tru
- Bean Implementation Class
  - http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/tru

Use Case Example: CalculatorEJB Unit Tests

- Just a POJO test in JUnit
- Creates a regular instance, invokes its business logic
- http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/tr

Use Case Example: CalculatorEJB Integration Test

- Starts up an EJB Container
- Deploys into it
- Looks up the EJB Proxy (typically in JNDI)
- ...but you don't write any of that
- http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/trunk/

Problems with Testing MDBs

- Asynchronous; how is the client to know when the server is done processing?
  - Thread.sleep sets you up for transient failures
- No return value; how do we check postconditions?
- In-container testing helps a bunch
  - Server and client run in the same JVM

Use Case: Asynchronous Components

- How is the test to know when processing is completed?
- What we shouldn't have to do:
  - http://anonsvn.jboss.org/repos/jbossas/projects/ejb3/trunk/te

Use Case Example: Asynchronous Component

- MDB which uses a shared barrier or latch
  - http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/tru

Use Case Example: Asynchronous Component Test

- Test uses a barrier, shared by the MDB
- http://anonsvn.jboss.org/repos/jbossas/projects/ejb-book/tr

Get Involved

- Active Community
- Contribute
  - Ideas on Forums or IRC
  - Feedback on trunk or Alpha releases
  - Bug Fixes
  - Enhancements
  - Documentation

FOLLOW US ON TWITTER

www.twitter.com/redhatsummit

TWEET ABOUT IT

#summitjbw

READ THE BLOG

http://summitblog.redhat.com/