Grails at LinkedIn

Alex Vauthey, Brian Guan
Monday May 5th 2008

Linkedin Overview

- Started in 2003
- 22M+ members, over 1.2M new per month
- Average member 41 years old, over $100K in household income
- 250+ employees, mostly in Silicon Valley HQ, Mountain View, CA
- Still private and more importantly - Profitable!

Engineering & Technologies

- 80+ Software Engineers
- Main site
  - built on Java, Tomcat, Oracle/MySQL
  - our own MVC framework, and customized Spring IoC
- Internal backend services
  - Java based via Spring Remoting, JMS
- Corporate Solutions
  - Grails + Java
- Mobile, Browser Plugins, Facebook apps
  - Ruby on Rails
- Partner API
  - Java based REST/HTTP services

Grails Projects/Products

LinkedIn

- Corporate Solutions
  - Private Portals for Corporate Customers
    - Recruiter
    - Research
  - Typical project requires short development cycle for premium customers with focused needs
- Engineering Services (Internal)
  - Continuous Build/Test Automation add-on to Hudson

Why Grails?

LinkedIn

- Needed a more productive webapp framework
- Rapid prototyping to demonstrate feasibility and productivity gain
- Benefit vs. risk analysis
  - Grails uses established open-source libraries
  - Easy to cut-down on the amount of Groovy
  - Integration with java
  - Open source !
  - LinkedIn Internal Spring expertise

Linkedin System Architecture

Complex eco-system

Many Many components!

- Web Apps (JSP/Servlet, Spring MVC, RoR, Grails)
- RPC Servers (APIs, Network graph, Search, business rules,...)
- Async Queues (tracking, real time statistics servers, communication infrastructure...)
- Data Access Services and database replicas
- Scheduled batch jobs

About 170 physical servers (and growing), all load balanced with an instant fail-over scheme

Key Findings

- Grails is productive for New web-app development...
- But bringing Grails into complex existing eco-system
  - First project spent significant effort on "Integration"
  - Subsequent projects saw productivity gain
- Productivity Gains from...
  - Model, View, Controllers, Taglibs, Command objects easy to create/maintain
  - Hot redeploy of MVC + Services works most of the time which helps rapid iteration
  - Books, online references, tutorials, examples exists to help train team members
  - Pure Business Logic as Grails Services/Domain Objects
    - Easy to create/maintain
    - ...but hard to extract for reuse by non Grails modules

Key Findings, cont.

- Grails based project is able to reuse many existing Linkedin Java assets
  - Java Business Logic Services, DAO, Value Objects…
  - Handle LinkedIn custom session as a New Session Scope for services/controller
  - Integrate LinkedIn custom Authentication and Single-Sign-On with Filters
- Having creator (Graeme Rocher) and key committer (Peter Ledbrook) in the early phase embedded helped tremendously
  - Training team new to Grails
  - Rapid bug fixes/enhancements incorporation into Grails 1.0

Challenges + Solutions (1)

- Integration into LinkedIn Eco-System (solutions extracted into LinkedIn Grails Plugin)
  - Code Integration - Grails Extension for...
    - Linkedin custom Spring instead of basic Spring
    - Linkedin custom Session instead of HTTP Session
    - Linkedin custom Auth/SSO
  - Build Integration -
    - Final product is a standalone WAR file
    - Grails build system assumes top level project within source tree and defaults to building within a project
    - Our project is a module within Linkedin's codebase and the centralized build convention expects a centralized build destination outside
    - Grails inject dynamic methods into Classes at build time so building a production WAR file still requires Grails runtime
    - Custom Ant scripts and Grails build-time event handlers to customize build and deployment

Challenges + Solutions (2)

- IDE - replace Eclipse with IntelliJ + JetGroovy Plugin
- Database - Linkedin convention does not match Grails defaults
  - GORM DSL for Naming, custom data type like XMLType...
- Test - fit Grails testing mechanisms into Linkedin's…
  - Integration Test (Junit based) - Entry point is controller actions
  - Unit Test (Junit based) - dependencies w. Groovy duck typing/EasyMock
  - GUI Test (HtmlUnit)
- Deployment - use LinkedIn centralized config external to Grails
- Cross Team Re-factoring
  - Groovy code invisible to other Java developers, system wide refactoring could lead to bugs during Tests, or worse, Runtime
  - Introduce Java based centralized glue service for integration
- Dynamic Language is new to team of Java Engineers
  - Groovy looks like Java, but dynamic nature makes errors hard to catch at compile time, and re-factoring hard to automate
  - More test coverage is required to catch silly errors

Conclusion

- For our usage, Grails is a lot more productive than current crop of mainstream Java Web-app Frameworks
- Grails can work in an Enterprise environment
- Grails can play as tightly integrated component in a Java ecosystem

- BTW...
  - Enjoy building with Grails? Cutting edge technologies?
  - Enjoy building features used by tens of millions of Professionals?
  - … come join Linkedin!