A4M36ISS: Introduction

Jiří Pechanec
QA Engineer
September 17th, 2015
Agenda

- Goals
- Organizational details
- Introduction into system integration
  - Principles
  - Past/Present/Future
- Tools/Products used
Goals and Organization
About team

- Red Hat
- Middleware QE (JBoss)
Goals

• Introduce into system integration world
• Overview of SI open-source software
• Find future Red Hatters :-(
Organizational details

- 8 topics in 4 days
- Mostly theory followed by a lab
- Grading based upon a team project
- Materials on-line
  - https://developer.jboss.org/wiki/SystemIntegrationWithJBoss
Introduction into System Integration
Why?

- Organic growth of an enterprise
- Mergers and acquisitions
- New values created by combinations of existing products
- Incremental legacy application replacements
- Access internal data from public facing applications
Bible
Why?

• Organic growth of an enterprise
• Mergers and acquisitions
• New values created by combinations of existing products
• Incremental legacy application replacements
• Access internal data from public facing applications
Why?

- Organic growth of an enterprise
- Mergers and acquisitions
- New values created by combinations of existing products
- Incremental legacy application replacements
- Access internal data from public facing applications
Architectures

- Spaghetti
- Hub-and-spoke
- Bus
- Service Oriented Architecture
- Service Component Architecture
- Event-Driven Architecture
- Microservices
Spaghetti

• Ad-hoc integration
• No system
• Difficult to introduce a new system
• Almost impossible to do a change
• Requires modification of source code of integrated systems
Hub-and-spoke

- Every system speaks only to a central node
- Clients effectively decoupled
- Easy to add a new node
- Difficult to modify existing API
- Can have scalability issues
- Most useful application
  - Message Broker
(Enterprise Service) Bus

- Applications communicate via (virtual) bus
- Main features
  - Connectivity
  - Routing
  - Transformation
Service Oriented Architecture

- Everything is a service with defined contract
- Mostly associated with web services
  - SOAP
  - WSDL
  - UDDI
- Descriptive registry of services
- WS-* specifications

Fig 1. Service Oriented Architecture
Service Component Architecture

- Artifacts
  - Component
  - Properties
  - Implementation
- Composite
- Entry point/service
- Reference
- Wire
- Not too widely used
- Defined as OASIS standard
  - Assembly model, language bindings,…
- Strict interface description and matching
Event-Driven Architecture

- Services produces events and react to events
- Strongly decoupled
- Very scalable
- Event processing
  - Simple
  - Complex, Stream
- Difficult to understand
- Difficult to debug and monitor
Microservices

• SOA done right
• Lightweight
• Tries to avoid application container – just simple application
• Services easily replaceable
• Smart endpoints/dumb pipes
  • Is not there a risk of spaghetti?
• Perfect match for (Linux) containers - Docker
Product levels

- Integration frameworks
  - API to implement EIPs
  - Basic communication protocols
- Enterprise Service Bus
  - Standalone container
  - Managed deployments
  - Monitoring
- Integration Suite
  - BPM
  - BAM
Tools/Products used
Products used

- Apache Karaf
- Apache Camel
- JBoss Fuse
  - Fabric8
- JBoss SwitchYard
- JBoss A-MQ/ActiveMQ
- apiman
- Docker
Apache Karaf

- OSGi-based container
- Runtime for other products/projects
  - Hot deployment
  - Dynamic configuration
  - Centralized logging
  - Shell
  - JAAS integration
  - Blueprint DI
- Supports Apache Felix and Eclipse Equinox runtime
Apache Camel

- Integration framework
- Routing and mediation engine
- Configurable via
  - Spring/Blueprint XML
  - Java/Scala DSL
- Support for almost all EIP
- URI-based endpoint configuration
- Integrated test kit
JBoss Fuse

• Enterprise Service Bus

• Inside
  • Karaf
  • Camel
  • ActiveMQ
  • CXF

• Fabric8
  • Central management and provisioning of large-scale installations
    • Ssh
    • jclouds
    • OpenShift
JBoss SwitchYard

• SCA-related service development and integration framework
• Augmentation of plain Camel with declarative
  • Transformation
  • Validation
  • Policy
  • Security
  • Routing
• Integration with
  • jBPM
  • BPEL
  • Drools
JBoss A-MQ

- Standalone message broker
- Inside
  - Karaf
  - Apache ActiveMQ
- Multi-protocol
  - Openwire
  - AMQP
  - STOMP
  - MQTT
- Cluster, mesh and network of brokers
- Manageable by Fabric8
apiman

- Open-source API Management
- Web-based configuration and management
- Policy Engine
  - Embeddable
  - Java EE
  - Vert.x
- Under fast development
- Centralization
  - Security
  - Quotas
  - Metrics
Docker

- Lightweight virtualization
- Complete isolated filesystem for a set of processes
  - Same kernel used
- Layering and inheritance
- Image registry
- But be careful with security
  - Docker is about running random crap from the Internet, as root and expecting not to be hacked ;-)
Questions?

plus.google.com/+RedHat
linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHatNews