Integration Simplified

Rich Remington
rremington@Rich-Software.com
Topics

- Integration, SOA, and ESB
- What Mule ESB is (and isn’t)
- Mule Architecture & Components
- Configuration & Deployment
- Enterprise Integration Patterns
- Main Demo
- Wrap-up & questions
Goals

- Educate 1..* people about Mule ESB & how to simplify integration projects
- Give back to an awesome community
- Have a bit of fun
- Confirm “laziness” is a good thing!
  - At least some of the time…
- Convince you it’s safe, easy, and fun, to “get on the bus”
Background

- Who am I?
  - Mathematician wannabe turned software guy
  - A bit lazy

- The bucket of water story
  - Moral: don’t work harder than you have to - it’s okay to be lazy ;-)
3 R’s - Plan to be Lazy :-)

- **Reduce**
  - Code complexity & bloat

- **Refactor**
  - Move code to its proper place
    - Transformation, Routing, Service Component

- **Reuse**
  - Plan ahead for service/component reuse
  - Use pre-built components & solutions
Integration Projects

- Why we integrate
  - Users don't care where business functions and information reside
  - We don't always own or control all the pieces
    - intranet & external partners
  - Cheaper than building new solutions from scratch
  - Users want access to information from various devices - browsers, mobile phone, handhelds, ...
Integration Projects

- How we integrate
  - Set up meetings with business domain experts
  - Identify systems and processes
  - More meetings…
  - Map data flows then model
    - Patterns are extremely helpful
  - Yet more meetings…
  - Define services & message formats
  - Start coding away…

- But, then what happens?
Integration Projects

- Three things in life are certain:
  - Death
  - Taxes
  - Requirements will change often!

- Plan for change
  - Design & build service/object interfaces for reuse
  - Take advantage of loose coupling
  - Switch implementations or protocols as needed over time, but not the interface
Integration Checklist

Planning
- Identify Processes and Systems
- Create an Integration Profile
- Service Identification
- Map Data Flows
- Performance Requirements
- Security
- Redundancy Requirements
- QoS Requirements
- Define Message Formats

Implementation
- Service Interfaces
- Mule Topology
- Model Event Flows
- Fault Handling
- Data Transformations
- Test Use-cases
- Service Implementation
- Endpoint QoS, Security
- Functional Testing
Integration Approaches/Technologies

- Custom, point-to-point solutions
  - Information silos live on
- Service Oriented Architecture (SOA)
  - Enables loose coupling of services that are OS & programming language independent
  - Web services (SOAP, REST), RPC, CORBA, etc.
- Enterprise Service Bus (ESB)
  - Helps orchestrate & manage the chaos
  - Good place to centralize general services
  - Encourages best practices
The Bus

“There are going to be times when we can't wait for somebody. Now, you're either on the bus or off the bus. If you're on the bus, and you get left behind, then you'll find it again. If you're off the bus in the first place — then it won't make a damn.”

-- The Electric Kool-Aid Acid Test (1968)
Daily Double

- What are two of the 3 R’s?
- What was the moral of the “bucket of water” story?
- Name the Integration Approach I’m suggesting you use.
Audience Poll

- How many of you...
  - Need to integrate
  - Are using point-to-point integration solutions
  - Are doing SOA (SOAP, REST, RPC, ...)
  - Not doing SOA, but planning to
  - Are using an ESB
  - Not doing ESB, but planning to
Why use an ESB

- Integration Styles
  - Batch
  - File Transfer
  - Shared Database
  - Request/Reply
  - Messaging
    - Synchronous
    - Asynchronous

- ESB Advantages
  - Supports all styles
  - Bridges legacy systems
  - Modular architecture
  - Simple and flexible
  - Easy to test and maintain
  - Scalable
  - Can be a step towards SOA
Why use an ESB (before)
Why use an ESB (after)
What is Mule ESB?

“Mule is a lightweight integration platform and service container that allows you to quickly and easily connect your applications together. Mule provides a robust, secure and scalable platform to build enterprise applications offering an array of transports such as JMS, HTTP, Email, FTP, JDBC and many more. It also offers a rich set of features for web services, message routing, mediation, transformation and transaction management. Designed around the ESB (Enterprise Service Bus) concept.”

-- www.mulesource.org
What is Mule ESB?

“Mule is a lightweight integration platform and service container that allows you to quickly and easily connect your applications together. Mule provides a robust, secure and scalable platform to build enterprise applications offering an array of transports such as JMS, HTTP, Email, FTP, JDBC and many more. It also offers a rich set of features for web services, message routing, mediation, transformation and transaction management. Designed around the ESB (Enterprise Service Bus) concept.”

-- www.mulesource.org
## What is Mule ESB?

| Service creation and hosting | • Expose transport end-points, EJBs, Spring beans, and POJOs as modular and reusable services  
|                            | • Host services as a lightweight service container |
| Service mediation           | • Separate business logic from messaging  
|                            | • Shield service components from message formats and protocols  
|                            | • Enable location-independent service calls |
| Message routing             | • Route messages based on content and complex rules  
|                            | • Filter, aggregate and re-sequence in-bound messages |
| Data transformation         | • Exchange data across applications with varying data formats  
|                            | • Transform data as needed, based on need  
|                            | • Format messages across heterogeneous transport protocols |
Mule: Lightweight and Adaptive

Adaptive integration

- Standards-based and vendor-neutral
- Seamless integration with existing infrastructure
- Plug-in architecture allows for building block approach
- Use with any application server or standalone

Development simplicity

- Simplified programming model
- Ease of use through configuration and control
- Rapid development and deployment through code reuse

Open source advantage

- No expensive up-front license commitments
- Access to source code allows customization and lowers risk
- Vibrant developer community delivering best practices and modules
What Mule ESB isn’t

- It does not place restrictions on your environment
- It is not message queuing, it uses other (JMS capable) products
  - And, it does not require message queues
- It is not complex
- It does not need to be “the grand master of all integration endpoints across the enterprise”
  - Think big, start small
  - It plays nicely with other ESBs
Why use Mule ESB

- Decouples Business Logic
- Location Transparency
- Transport Protocol Conversion
- Message Transformation
- Message Routing
- Message Enhancement
- Reliability (Transactions)
- Security
- Scalability
Why choose Mule

- No prescribed message format
  - XML, CSV, Binary, Streams, Record, Java Objects
  - Mix and match
- Zero code intrusion
  - Mule does not impose an API on service objects
  - Objects are fully portable
- Existing objects can be managed
  - POJOs, IoC Objects, EJB Session Beans, Remote Objects
  - REST & Web Services
- Easy to test
  - Mule can be run easily from a JUnit test case
  - Framework provides a Test compatibility kit
  - Scales down as well as up
Mule ESB Architecture

- Architecture is based on concepts discussed in Enterprise Integration Patterns
Mule Components

- **Endpoint**
  - Channel, MessageReceiver, Connector

- **Transformer**
  - Message transformation / enhancement

- **Router**
  - Message flow control (inbound/outbound)

- **Service Component**
  - Your integration logic lives here
Mule Configuration

- Spring based
- mule-config.xml (default)
- Multiple config files using import
Daily Double

- Name two reasons to use Mule ESB
- Name one thing Mule ESB isn’t
- Name two Mule Components
Mule Examples (mini-demo)

- Download Mule
  - http://www.mulesource.org/display/MULE/Download

- Configure the environment
  - MULE_HOME, JAVA_HOME

- echo example
  - options 1 & 3, show config
Mule Deployment

- Standalone
- Embed in servlet container
- JCA 1.5 resource adapter
Mule Deployment Topologies

- Enterprise Service Bus

Diagram showing various service bus components and their relationships.
Other Deployment Topologies

Mule also supports:
- pipeline
- peer network
- client/server
- hub-and-spoke, etc.

Combine them to make an ESN
Enterprise Integration Patterns

- **Message Filter**
  - Service component receives specific message types

- **Message Translator**
  - Transformer: change format, tag names, etc.

- **Content Enricher (Message Enhancement)**
  - Add additional data to message

- **Content-based Router**
  - Route message based on contained data

- **Messaging Bridge**
  - Send message between different protocols or messaging systems
Enterprise Integration Patterns (cont.)

- **Content Filter**
  - Shrink message to fewer data items

- **Polling Consumer**
  - File directory, email, FTP

- **Scatter-Gather**
  - Send to multiple endpoints / aggregate replies

- **Splitter**
  - Batches, Lists, Collections, etc.

- **Wire Tap**
  - Inspect messages during flight (logging, monitoring, etc.)
Main Demo

- Set the stage [see diagrams]
  - Describe scenario
  - Configuration
- Show demo
- Show code
  - Transformer(s)
  - Service Component(s)
Main Demo - Scenario
Other MuleSource Goodies

- **MuleForge**
  - Mule extensions, mostly provided by community
  - Mule IDE, modules, tools, transports, etc.

- **Mule Galaxy™**
  - SOA governance platform with an integrated registry/repository

- **Mule ESB Enterprise**
  - Subscription based support that includes
    - Mule HQ™: systems monitoring / management
    - Premium connectors (e.g. IBM Websphere MQ)
    - Online knowledgebase & support links
The Bus

“The bus came by and I got on, that's when it all began…”

-- "The Other One" by the Grateful Dead

So, get on the bus!
Audience Poll

- How many of you…
  - learned something about Mule ESB?
  - learned something you didn’t know about integration patterns?
  - had a bit of fun?
  - will think more about how to be lazy?
Summary

- Integration Projects
- What Mule ESB is (and isn’t)
- Mule Architecture, Components, Deployment
- Enterprise Integration Patterns
- Easy to implement integration scenarios
- Plan to be lazy :-)

All content copyright © 2009, Rich Software Inc; portions copyright © 2009, MuleSource Inc. All rights reserved.
Double Jeopardy

- Send me a use-case for Mule ESB within your organization using the contact form on the RSI website (www.rich-software.com/main/contact)
- Put “Mule ESB use-case” in the subject
- The use-case must be received within 48 hours of this talk to be eligible for a prize
- *I will buy two lucky people a free lunch that includes up to two hours of free business analysis on your use-case and/or mentoring on Mule ESB. And, you will receive one of the Manning books!*
Resources

- **Mule ESB:**
  - http://www.mulesource.org (Community)
  - http://www.mulesource.com (Enterprise)

- **Open Source ESBs In Action**
  - http://www.esbinaction.com

- **Enterprise Integration Patterns**
  - http://www.enterpriseintegrationpatterns.com

- **Mule In Action (July 2009)**
  - http://www.manning.com/dossot
Thank you!

Questions?