Enterprise Rails on Ruby

Steven Beales

Part 1

- Enterprise meet Ruby on Rails ...
- Enterprise Rails on Ruby
  - Ruby on Rails framework extended with Ruby to provide ‘missing’ enterprise features.
  - Best tool for building an enterprise service-oriented architecture?

Enterprise Management Issues

- Are we ready for a new language and development style?
- Is our team ready?
- Is our company ready?
- Do we have the right projects?

Right Developer Attitude is Crucial

- Agile teams *get* Ruby on Rails sooner than traditional ones
- Ruby on Rails increases productivity and sheer joy of development
- Unenthusiastic or mediocre developers won't *get it*

Rails vs the Enterprise

- The Rails framework
  - simplicity
  - antithesis of over-complex 'enterprisey' frameworks.
- David Heinemeier Hansson
  - “Rails will do what it does, and will not complicate itself to support things it doesn't like.”
  - “Enterprise will follow legacy to become a common insult among software creators and users.”
  - “By the end of 2006, it will be written that enterprise means bulky, expensive, dated, and golf.”

Enterprise-class software (ECS)

- ECS are the behemoths that everybody thinks power Big Companies.
  - 10 million transactions a minute
  - span five different databases
  - three different hardware platforms

- "You can't use Ruby for enterprise (ECS) applications.
  - It doesn't scale, nobody understands it, and it is slow.
  - No ecosystem or vendor behind it.
  - No first spectacular failure - means all Ruby apps will be failures."
    - *Prevailing Java/.NET opinion ...*

- Transactional ECS is a good fit for J2EE/.NET
  - For the small percentage of applications (<1%) that need distributed transactions/exceptions ...

Software that runs the enterprise (STRTE)

- But for everything else ...
  - Ruby on Rails is a good fit ...
  - Downsizing J2EE/.NET is not such a good fit.

- Software that runs the enterprise
  - Internal applications/portals that make each department work
  - Reports for CxOs
  - Code that transforms Excel spreadsheets into a data warehouse etc

- STRTE has to adapt to multiple pressures placed upon it every day
  - changing whims of rotating CxOs
  - needs of newly acquired subsidiaries
  - changing nature of the hardware components
  - any number of other, rapidly-changing variables

Simplicity vs Complexity

- “Using the full might of a J2EE stack to write a small stand- alone application is using a sledgehammer to crack a nut. But I keep hearing the sound of nuts being pulverized as developers seem to think that using anything other than J2EE is somehow unprofessional.” Dave Thomas

- “Enterprise Java, has grown into a complex behemoth that consists of layer upon layer of complexity.” Dave Geary (author of Java Server Faces book).

- “I’d rather write in a language that let’s me focus on the application, and which lets me express myself clearly and effectively.” Dave Thomas

- A better algorithm will easily gain back any marginal performance hit taken for using a slower language.

- Less time coding infrastructure means more emphasis on clean design and elegance in all aspects of the application.

Ruby is glue that does not set

- Ruby is a glue language. Like Perl, Ruby works and plays incredibly well with external data, resources, and programs.
- But, there's a major difference. Over time, Ruby programs stay malleable and easy to change. Ruby is the glue that doesn't set.
- As we move forward, more and more of our applications will be less like one-man bands and more like orchestra conductors.

The Ruby Way

- "Everything should be as simple as possible, but no simpler." Einstein
- "Perfection is achieved, not when there is nothing left to add, but when there is nothing left to take away." de St. Exupery
- We can't avoid complexity, but we can bury it out of sight. This is the old "black box" principle; a black box performs a complex task, but it possesses simplicity on the outside.
- In Ruby we see simplicity embodied from the programmer's viewpoint (array.indexes and array.indices both valid).
- When you say something in a small language, it comes out big. When you say something in a big language, it comes out small. "Ontogeny recapitulates Phylogeny"
- "Ruby arose from the human urge to create things that are useful and beautiful. Programs written in Ruby should spring from that same source. That, to me, is the essence of the Ruby Way." Hal Fulton

Part II

- Ruby on Rails meet Enterprise ... advanced Ruby techniques to make Ruby on Rails more ‘enterprisey’.

Service-oriented Architectures

- The vision of service oriented architectures - IT as loosely coupled on demand services that are semantically interoperable.
- Ruby's niche is quickly writing small decoupled servers/clients to glue together existing applications.
- Rails' niche is quickly writing new web applications.
- Though it's not supposed to matter what language you write your web services in, some languages (Ruby) are more suited than others.
- Together Ruby and Rails form a micro-architecture that grows organically through enterprise from bottom up.

Enterprise Patterns in Ruby on Rails ...

- Domain Logic Patterns: Domain Model
- Data Source Architectural Patterns: Active Record
- Object-Relational Behavioral Patterns: Unit of Work, Identity Map, Lazy Load
- Object-Relational Structural Patterns: Identity Field, Foreign Key Mapping, Association Table Mapping, Single Table Inheritance
- Object-Relational Metadata Mapping Patterns: Metadata Mapping, Query Object, Repository
- Web Presentation Patterns: Model View Controller, Front Controller, Template View
- Session State Patterns: Client Session State, Server Session State, Database Session State
- Base Patterns: Layer Supertype, Special Case, Plugin

Distributed Ruby (DRb)

- It is a library that allows you to send and receive messages from remote Ruby objects via TCP/IP.

- DRb allows Ruby programs to communicate with each other on the same machine or over a network. DRb uses remote method invocation (RMI) to pass commands and data between processes as though they were local.

- Use DRb to add long running background processes to Rails.

- On a conference panel in 2003 someone asked Robert C. Martin about the future of SOAP. After listening to the answers of his panelists, Uncle Bob replied: "I'd rather use a socket."

Complete DRb Server On 1 Slide

```
require 'drb'
class ProprietaryServer
  def mask(cc_num)
    fields = cc_num.split("-")
    return "XXXX-XXXX-XXXX-#{fields[3]}"
  end
  def transform(cc_num)
    return cc_num.reverse
  end
end
server = ProprietaryServer.new
DRb.start_service("druby://localhost:9000", server)
DRb.thread.join
```

- DRb Client On <1 Slide (plus my cat)

  - require 'drb'
  - DRb.start_service
  - obj = DRbObject.new(nil, "druby://localhost:9000")
  - cc_num = "5438-0166-8187-9942"
  - puts(obj.mask(cc_num))
  - puts(obj.transform(cc_num))

DRb Tips

- Security
  - Access Lists
    - acl = ACL.new(%w(deny all
    - allow 192.168.1.*
    - allow localhost))
    - DRb.install_acl(acl)
  - Disable Remote Evaluation
    - $SAFE = 1 # disable eval() and friends
- Concurrency
  - Mutex.new.synchronize do ...
- Rinda - http://segment7.net/projects/ruby/drb/rinda/ringserver.html. Automatic discovery of DRb servers and clients.

Multiple Databases

- By default, all Rails models get access to the default connection.
  - But make a connection from one of your model classes (by calling establish_connection( ))
  - Connection will be available from that class and any of its children but not from ActiveRecord::Base

- class External < ActiveRecord::Base
  - self.abstract_class = true
  - establish_connection :legacy
  - #add legacy entry to database.yml
- end

- class InLegacyDatabase < External
- end

Legacy Database - Basic

- ActiveRecord::Base.table_name_prefix = "wp_" #wp_comment
- ActiveRecord::Base.primary_key_prefix_type = :table_name_with_underscore #comment_id
- ActiveRecord::Base.pluralize_table_names = false
- set_table_name for individual table names outside the rails convention
- set_primary_key for individual tables whose primary key was not called id
- find_by_sql for arbitrary complex queries or tables with no primary key
-
- before_validation_on_create for non-auto-generated primary keys
- set_column_prefix to prefix atributes #patient_first_name, patient_last_name

Legacy Databases - Advanced

- Composite Keys
  - http://compositekeys.rubyforge.org/
- Data Mapper
  - rBatis (https://svn.apache.org/repos/asf/ibatis/trunk/rb/)
- Foreign Keys
  - http://wiki.rubyonrails.org/rails/pages/

Composed of - Legacy mapping

- Here we map person_address_street, person_address_city etc fields to a separate address object ...

- class Person < ActiveRecord::Base
- composed_of :address, :mapping => [%w(address_street street),
- %w(address_city city),
- %w(address_state state),
- %w(address_country country)]
- end

Composed of - Make dumb data smart

- Here we make a custom class from letter_grade (A .. F) for our students ... this class sorts C+ below B- etc

- class CourseRecord < ActiveRecord::Base
- composed_of :grade,
- :class_name => 'Grade',
- :mapping => [%w(letter_grade letter_grade)]
- end
- class Grade
- include Comparable
- attr_accessor :letter_grade
- SORT_ORDER = ["f", "d", "c", "b", "a"].inject({}) {|h, letter|
- h.update "#{letter}-" => h.size
- h.update letter => h.size
- h.update "#{letter}+" => h.size }
- def <=>(other)
- SORT_ORDER[letter_grade.downcase] <=> SORT_ORDER[other.letter_grade.downcase]
- end

Web Services

```
require 'soap/wsdlDriver'
 def googletest
   yourkey = 'YOUR GOOGLE DEVELOPER KEY'          # Your Google dev key
   @yourquery = 'SEARCH TEXT'                     # Search value
   XSD::Charset.encoding = 'UTF8'                 # Set encoding
   wsdlfile = "http://api.google.com/GoogleSearch.wsdl"   # WSDL location
   driver = SOAP::WSDLDriverFactory.new(wsdlfile).create_rpc_driver
   @result = driver.doGoogleSearch(yourkey, @yourquery,  0, 3, false, '', false, '', '')
 end
```

- wsdl2ruby - Ruby magic for web services ...

Domain Specific Languages

- Rails examples
  - 3.years + 13.days + 2.hours
  - 4.months.from_now.next_week.monday
  - has_one :department, :foreign_key => "department_number"
- Ruby metaprogramming (another talk)
- Like Lisp - we build language up to problem - not problem down to meet language.

Enterprise messaging

- Enterprise Rails needs
  - A competent Ruby-based message-queueing system based on the most recent standards and security practices.
  - Support the emerging AMQP protocol
  - Interoperability with new and established Java-based solutions." Francis Cianforcca

- Francis Cianforcca wrote http://rubyforge.org/projects/reliable-msg/ and is currently writing the above.

- ACTIVEMQ + STOMP

  - http://rubyforge.org/projects/stomp/

XML - REXML

- def echo()
- doc = REXML::Document.new File.new("echo.xml")
- response = {
- "api_key" =>
- doc.elements["parameters"].elements["api_key"].text,
- "action" =>
- doc.elements["parameters"].elements["action"].text,
- "controller" =>
- doc.elements["parameters"].elements["controller"].text
- }
- end

- <?xml version='1.0' encoding='UTF-8'?>
- <parameters>
- <controller>service</controller>
- <api_key>1234</api_key>
- <action>echo</action>
- </parameters>

- The each_element_with_attribute method is called like this:
- • xml.each_element_with_attribute (key [,value, max,
- name]) {|Element| ... }

- xml.each_element_with_text([text, max, name]) {
- |Element| . . . }.

LDAP, RDF

- ActiveLDAP - http://dataspill.org/pages/projects/ruby-activeldap
- ActiveRDF - http://www.activerdf.org/

Java

- JRuby is an implementation of a Ruby interpreter in Java.
  - It's useful for embedding Ruby in Java and vice versa.
- rjb (Ruby Java Bridge) uses JNI to access Java objects in Ruby programs.
  - Easy to use for simple cases.

- Next month's talk

Closing thoughts

- “I always knew one day Smalltalk would replace Java. I just didn’t know it would be called Ruby.”
- Kent Beck