Mapping Rails to Legacy Systems

- Devon Jones, Architect of Web Systems, Vonage
- Stephen Becker, Developer, Vonage

About Us

Here are the key players that made it a reality.

- **Devon Jones**
  Software Architect
- **F. Morgan Whitney**
  Senior Developer/Designer
- **Stephen Becker IV**
  Software Developer

Rails at Vonage

- Subscribe System
- Metrics Mole
- RESTful Services
- VFax
- Numerous other existing apps
- Multiple rails projects in development

Grand Strategy

In order to achieve our company's vision of changing the way the world communicates we must transform our legacy applications into agile systems composed of small programs with transparent interfaces.

Key Strategic Elements

- Change gradually & consistently
- Small programs & projects
- Transparent Interfaces

Mapping Rails to Legacy Systems :: Stephen Becker IV & Devon Jones

Small Projects Succeed

Of 13,522 surveyed projects

- 34% Unqualified successes
- 15% are abandoned
- 51% are challenged*

*Projects with cost overruns, time overruns, and projects not delivered with the right functionality to support the business.

From InfoWorld August 13, 2004
http://www.infoworld.com/article/04/08/13/33FEmyth5_1.html

Change gradually & consistently

- People fear change.
- Small projects are low risk projects.
- Adjust course rapidly.
- Show success early and often.

Small programs

- Break up existing business logic into atomic chunks which access individual resources or contain specific logic.
- Small scopes - create small simply services and applications that solve one problem well and play well with others.
- Interface flexibility - plan for adding more to your interface, consumers should always take what they need and ignore the rest

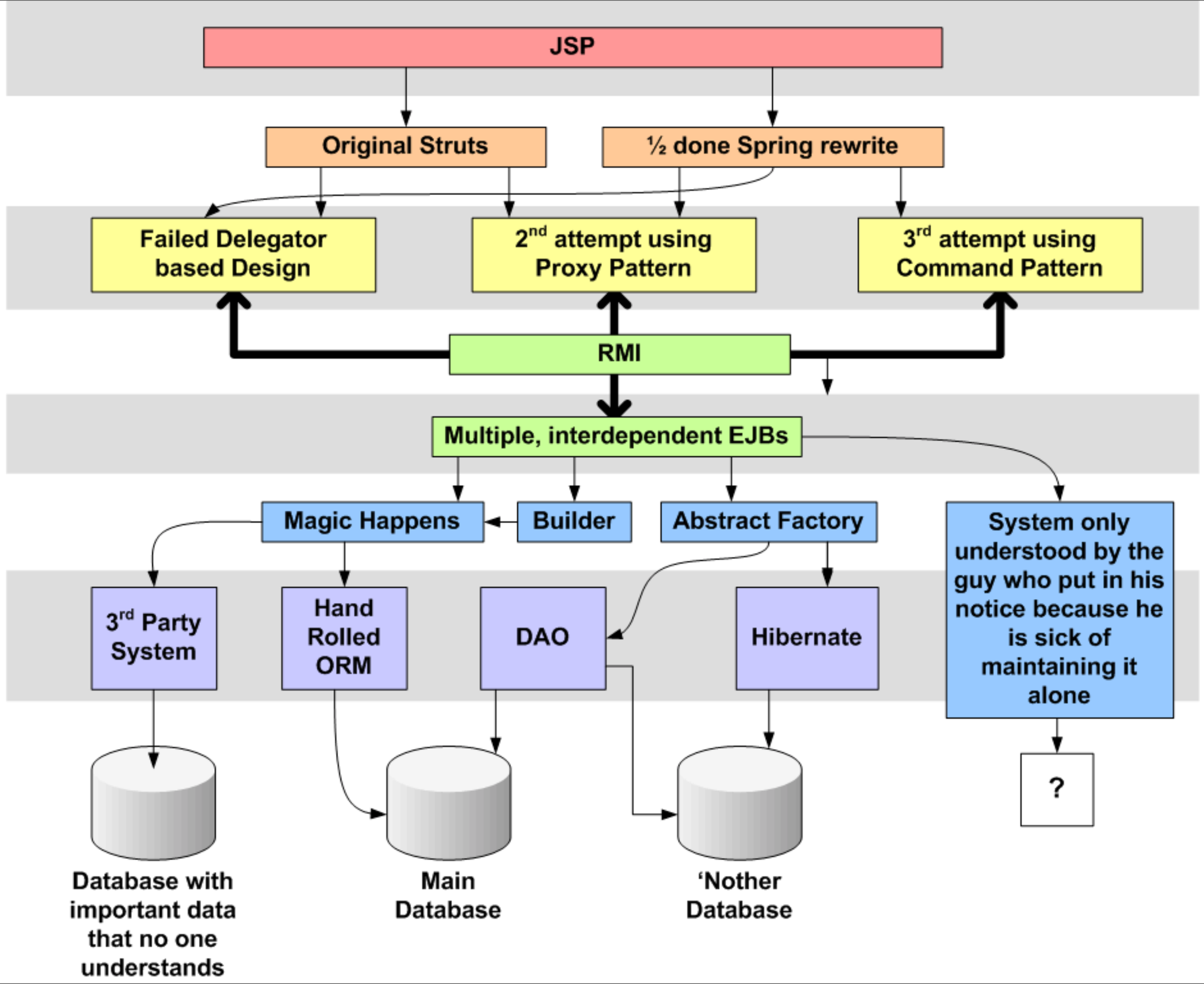
Transparent Interfaces

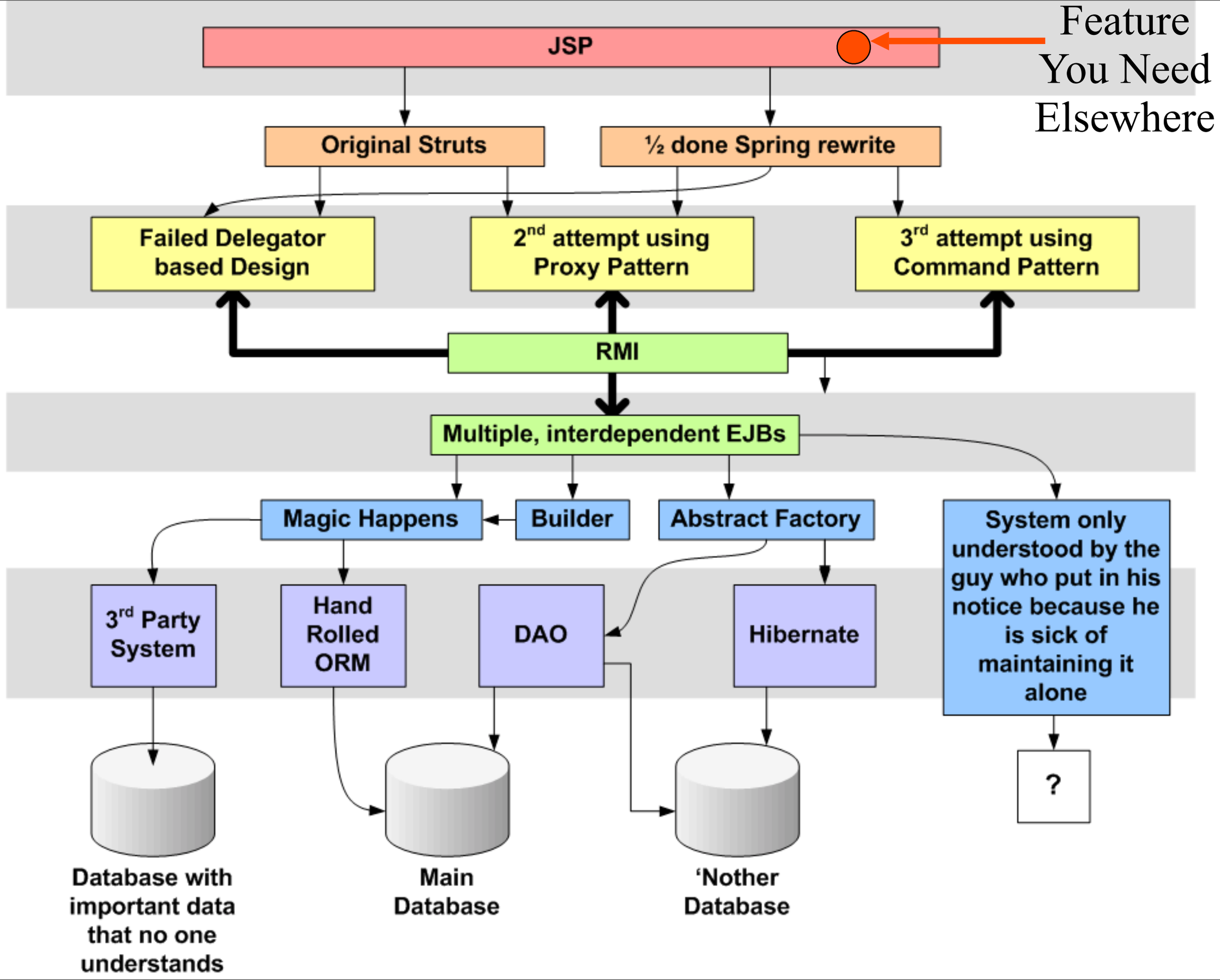
- Design for visibility to make consumption, inspection and debugging easy
- Interface inputs result in a predictable and dependable output
- Allows for easy re-use

Some specifics...

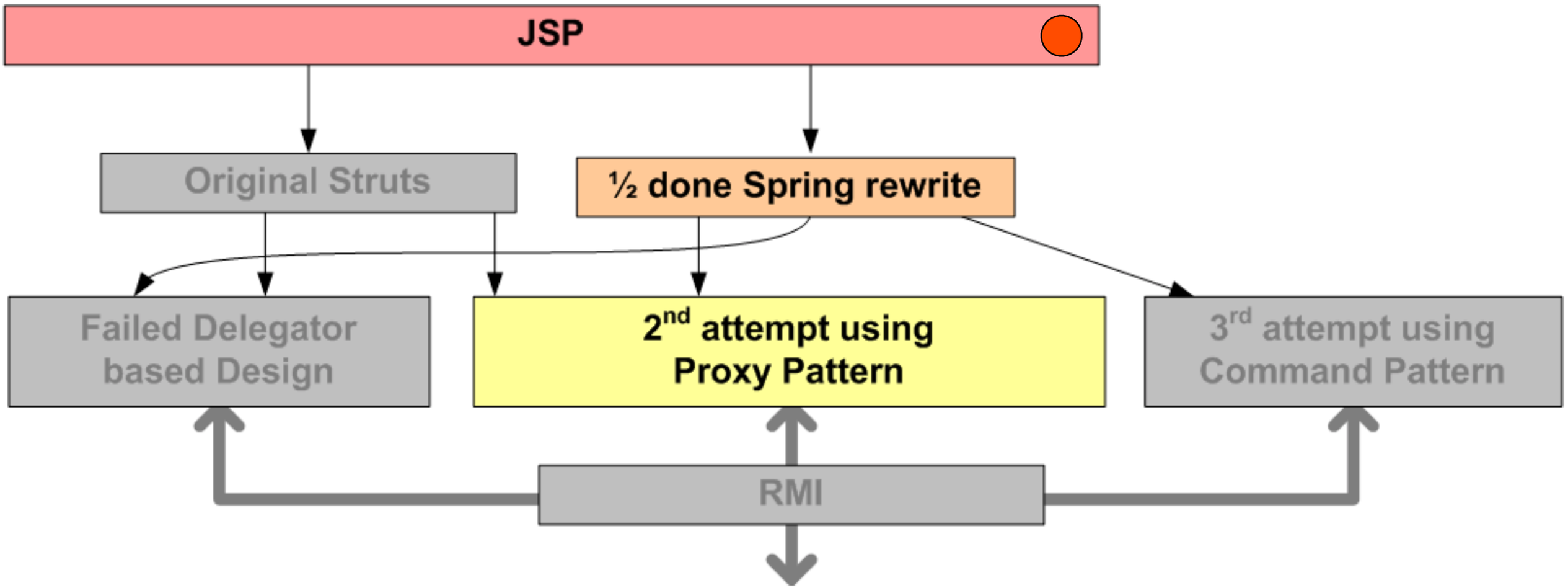
Ripping apart the system

- This is a strategy to eliminate code from a large EJB system
- Wrap features in simple RESTful services
- Create a functional test suite
- Ultimate goal is to turn off code in the legacy system while creating small services that implement your features

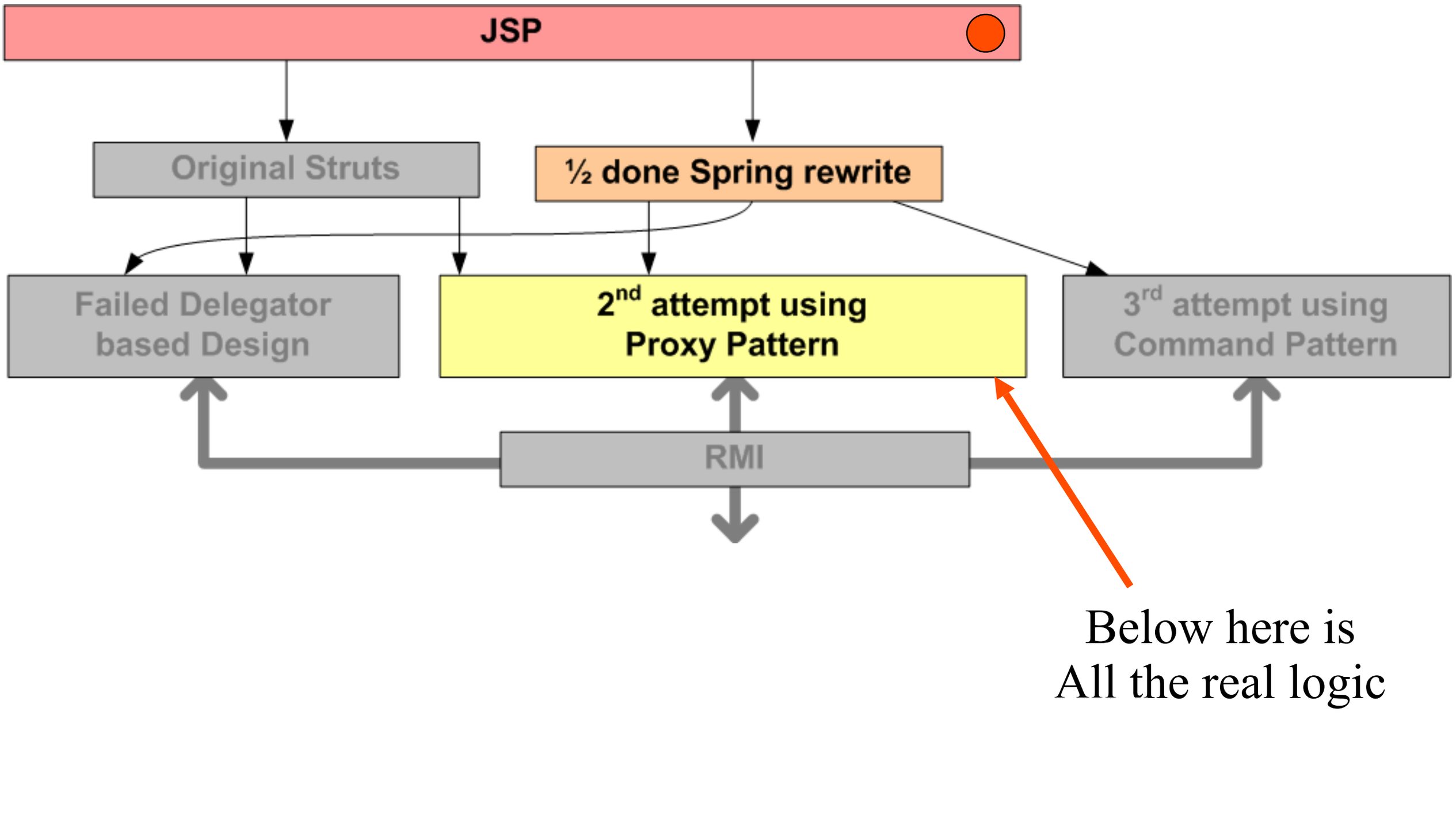

JSP
Original Struts
½ done Spring rewrite
Failed Delegator based Design
2nd attempt using Proxy Pattern
3rd attempt using Command Pattern
RMI
Multiple, interdependent EJBs
Magic Happens
Builder
Abstract Factory
System only understood by the guy who put in his notice because he is sick of maintaining it alone
3rd Party System
Hand Rolled ORM
DAO
Hibernate
?
Database with important data that no one understands
Main Database
'Nother Database


Feature You Need Elsewhere
JSP
Original Struts
½ done Spring rewrite
Failed Delegator based Design
2nd attempt using Proxy Pattern
3rd attempt using Command Pattern
RMI
Multiple, interdependent EJBs
Magic Happens
Builder
Abstract Factory
System only understood by the guy who put in his notice because he is sick of maintaining it alone
3rd Party System
Hand Rolled ORM
DAO
Hibernate
?
Database with important data that no one understands
Main Database
'Nother Database

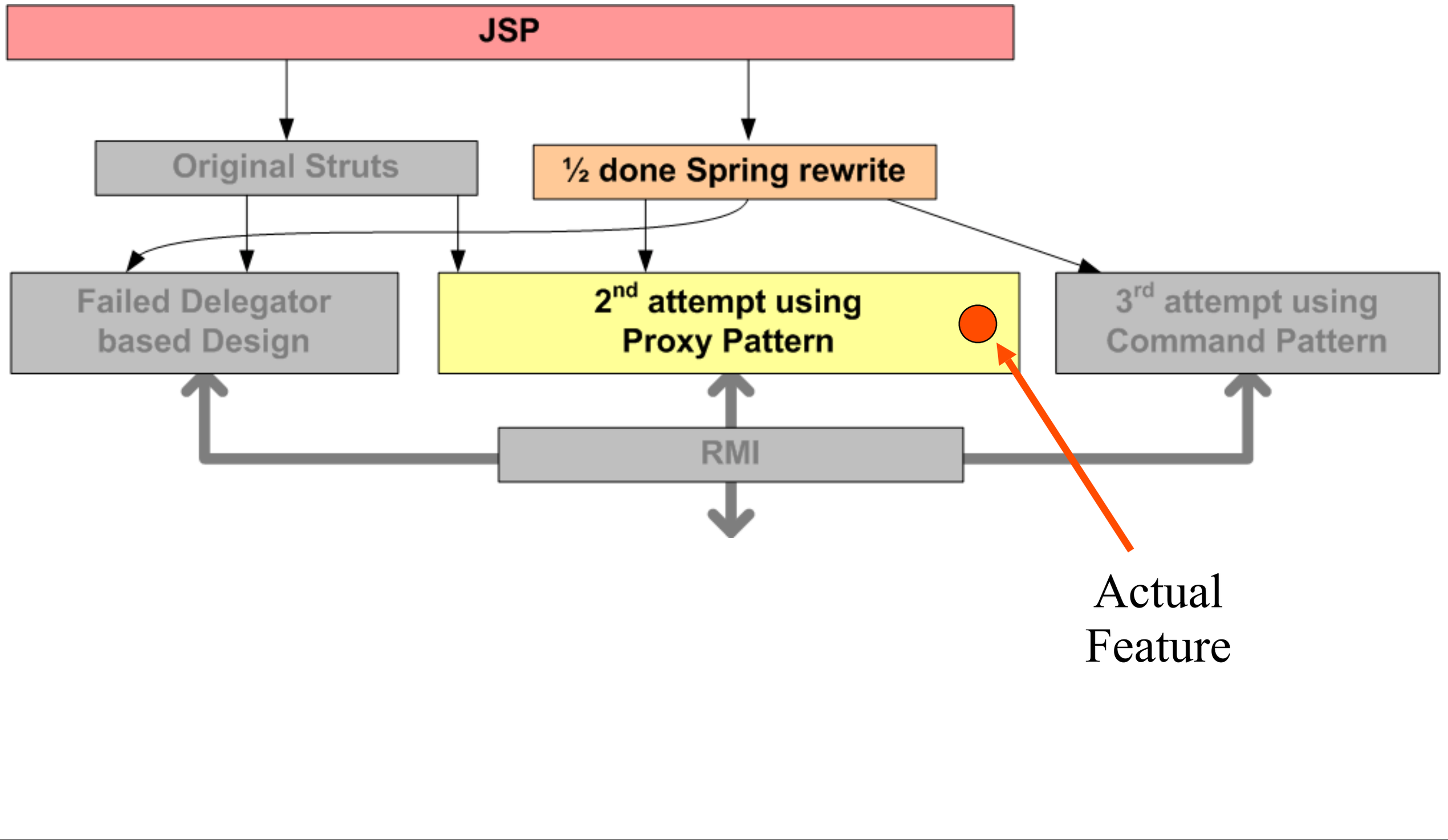
Identify a choke point for your feature



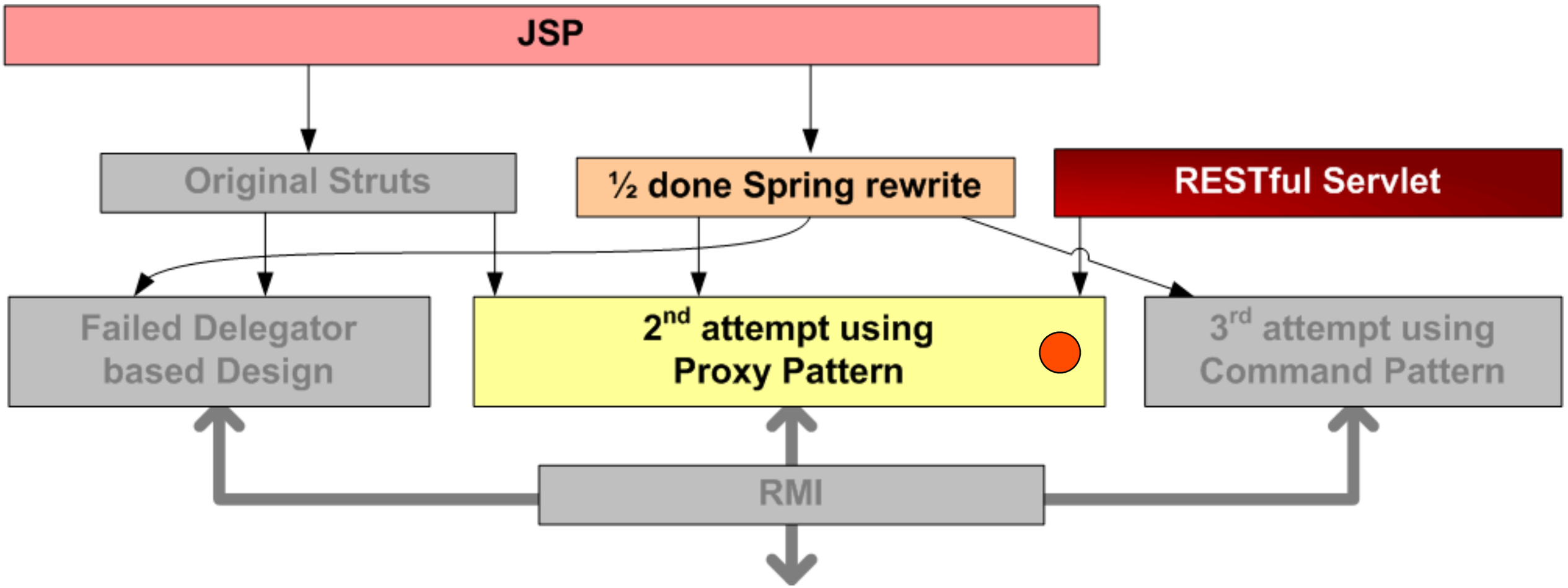
Identify a choke point for your feature



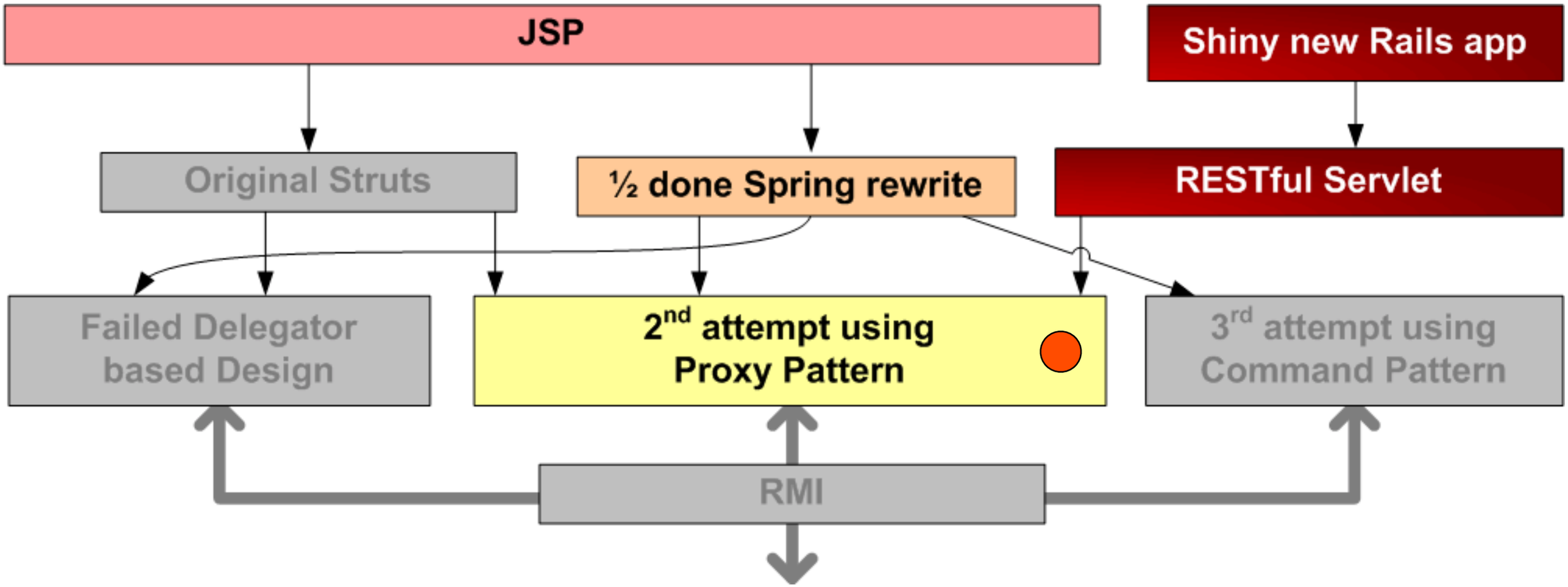
Slap a RESTful service on it



Slap a RESTful service on it

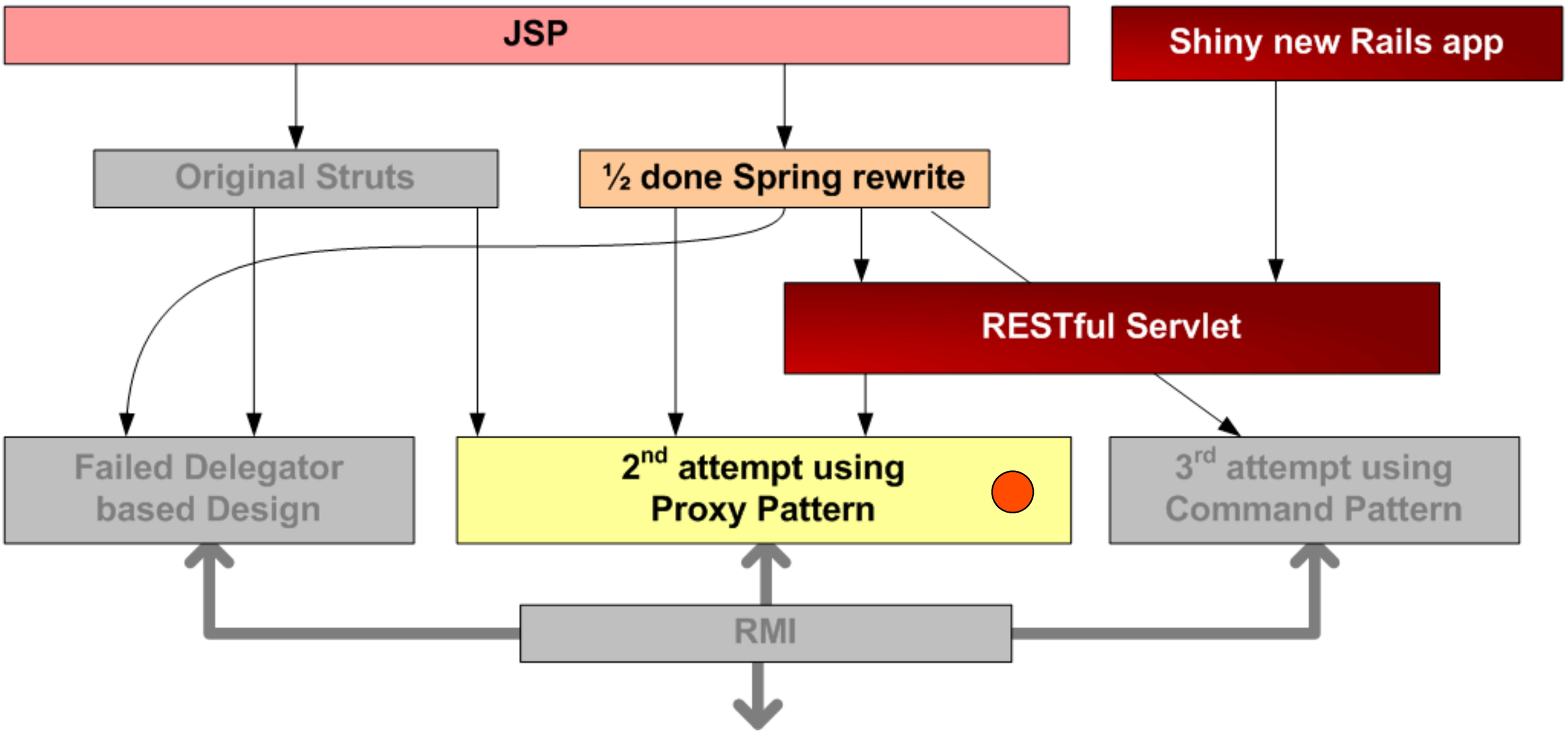


Slap a RESTful service on it

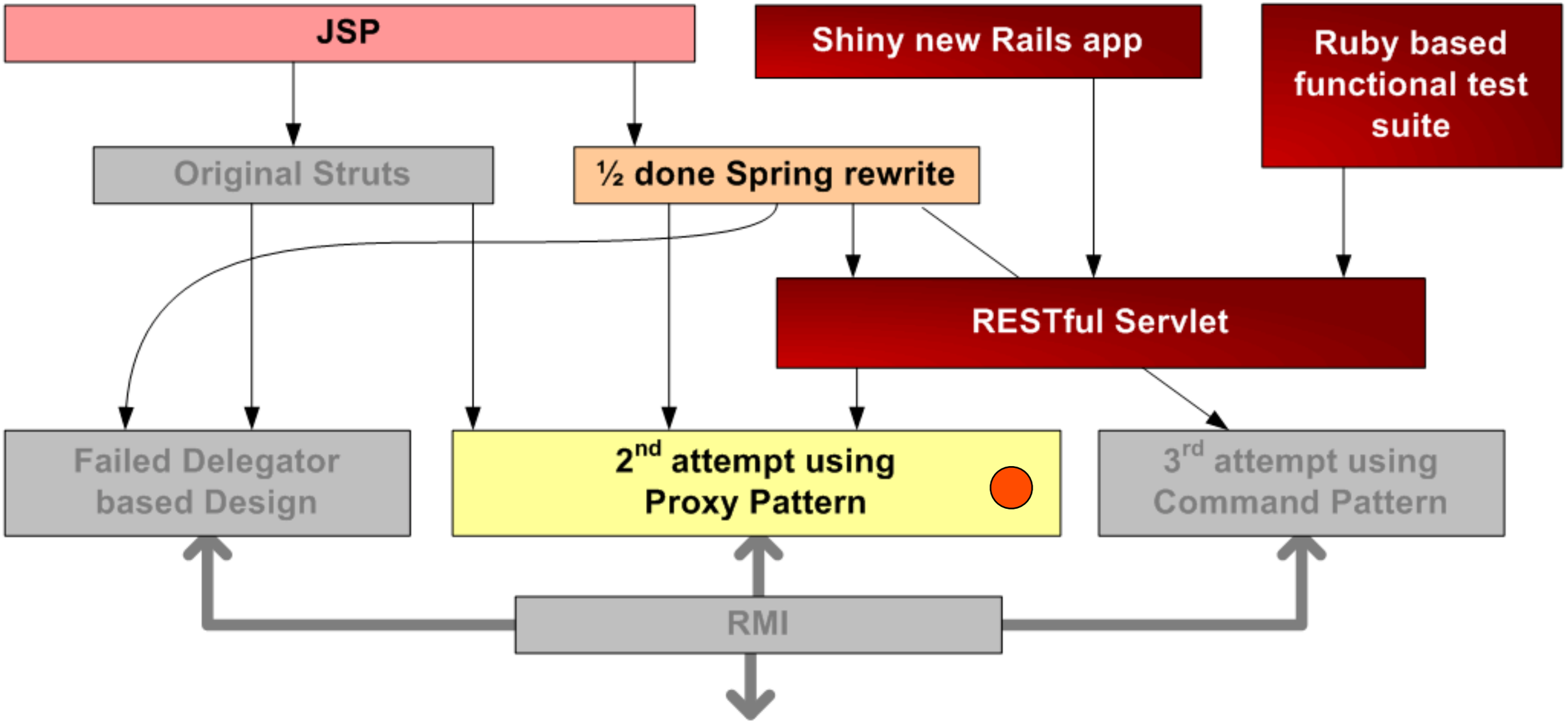


PORT

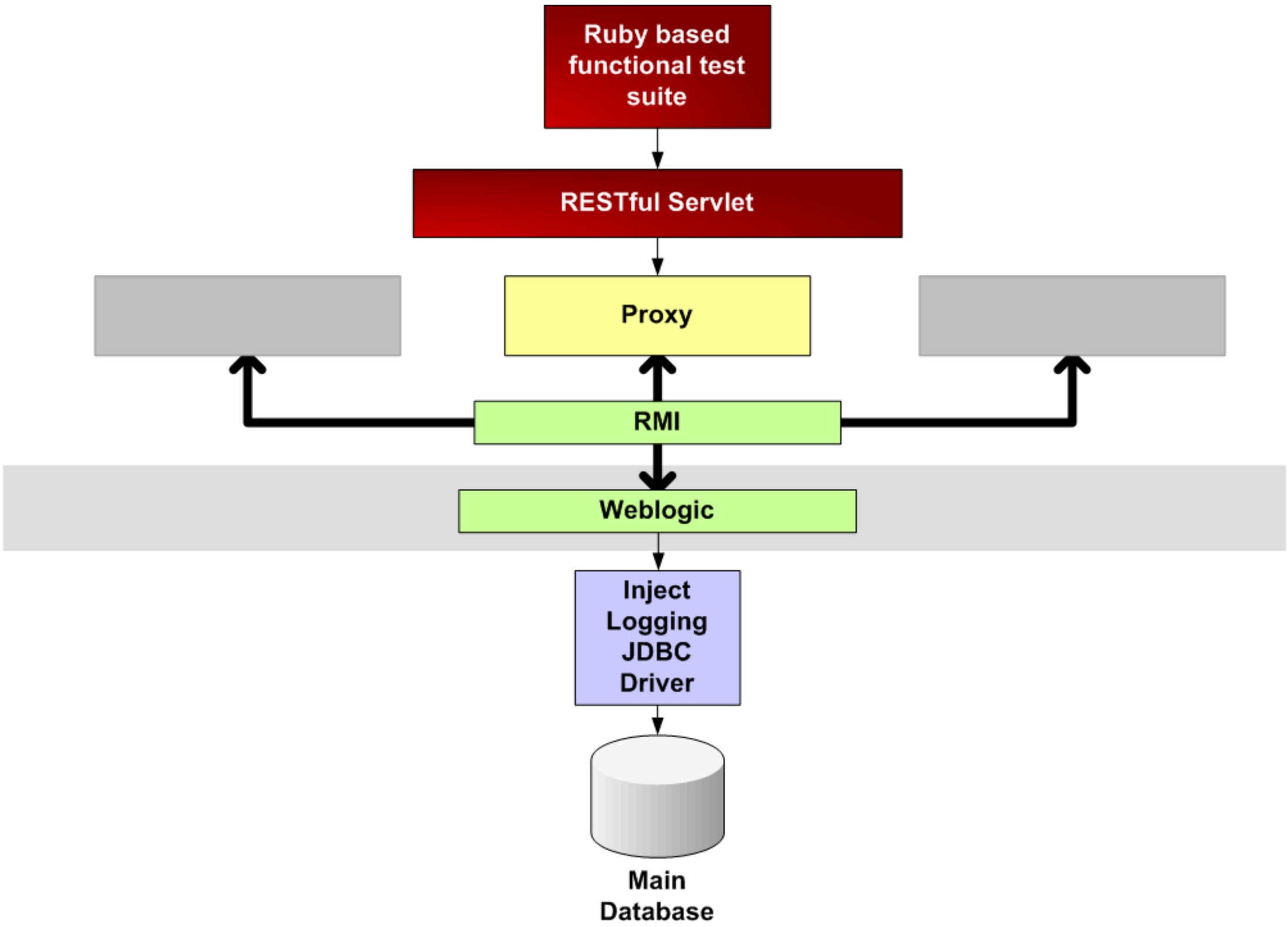
Port the old front end t o use the service



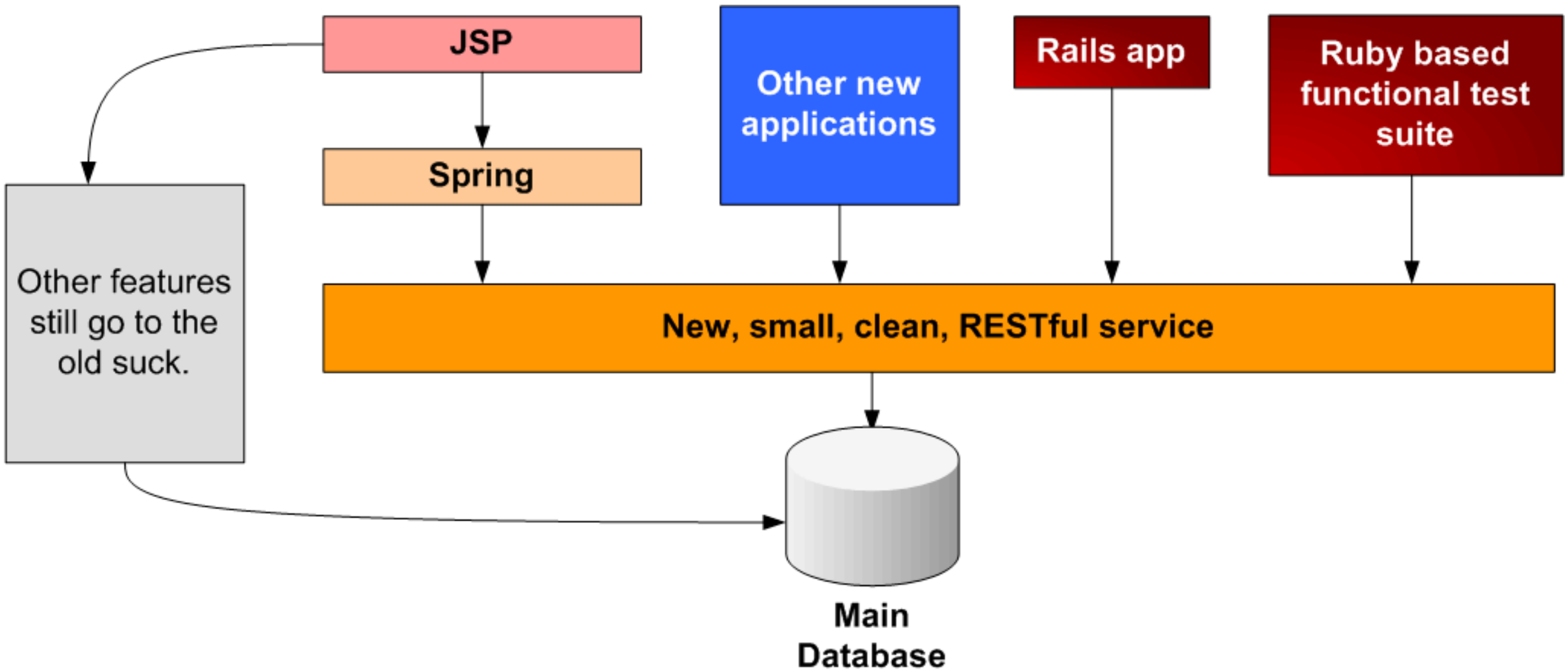
Write functional tests against the service



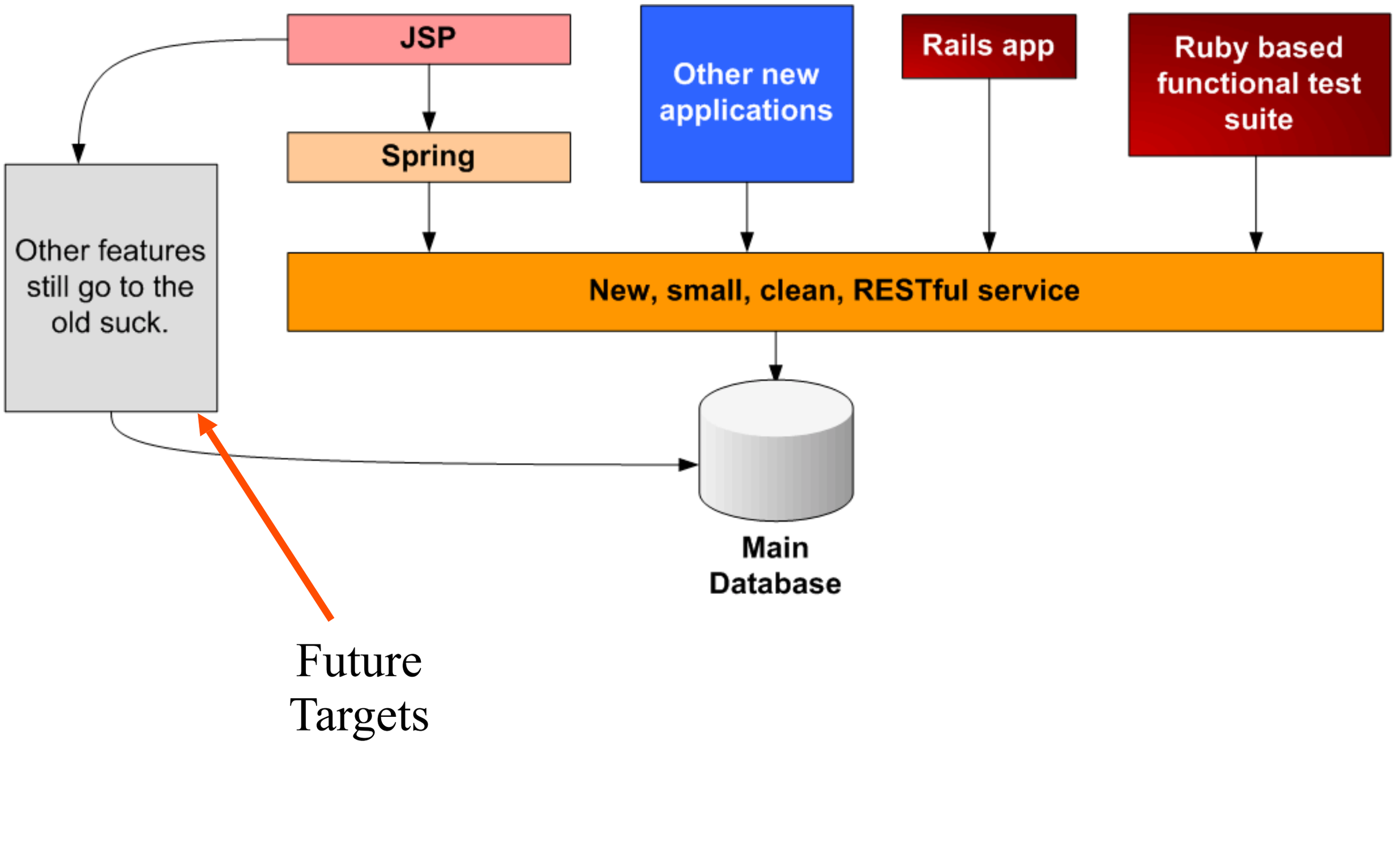
Use the functional test suite and JDBC logging to map the possible endpoints



Build a clean service to encapsulate the logic



Build a clean service to encapsulate the logic



Rails Tactics For Legacy

- Problems:
  - How to cope with bad database design
  - What you can do when your rails app needs application state from the legacy app

Active Record in Legacy Databases

- Database Degradation
- Documentation
- A non standard database schema
  - 8 different ways to represent false
  - Date as a varchar

one.rb

```
include LegacyAr
class One<ActiveRecord::Base
  LegacyAr.csv_column :bad
  LegacyAr.date_column :bad_date2, “%Y-%m-%d %H:%M:%s”
end
```

legacy_ar.rb

```
module LegacyAr
    def self.csv_column(att_name)
        att_name=att_name.to_s
        code_block = <<-end_of_block
            def #{att_name}
                @attributes["#{att_name}"]||=""
                @attributes["#{att_name}"].split(',')
            end
            def #{att_name}=(array_data)
                raise "Size error" if @attributes["#{att_name}"]
                   && array_data.size!=#{att_name}.size
                @attributes["#{att_name}"]=array_data.join(",")
            end
        end_of_block
        eval code_block
    end
end
```

legacy_ar.rb

```
module LegacyAr
     def self.date_column(att_name,date_str_format)
          att_name=att_name.to_s
          code_block = <<-end_of_block
          def #{att_name}
               return Date.strptime(@attributes["#{att_name}"],"#
                 {date_str_format}") if  @attributes["#{att_name}"]
          end

          def #{att_name}=(date)
               raise "not a date" unless date.kind_of?(Date)
               @attributes["#{att_name}"]=date.strftime("#{date_str_format}")
          end
          end_of_block
          eval code_block

     end
end
```

ark.rb

```
has_many :llamas,
         :foreign_key => "animal_id",
         :conditions  => "date_birth <= sysdate and status = 'Q'"
has_many :shrews,
         :foreign_key => "animal_id",
         :finder_sql  => 'SELECT s.* from shrews s, occupation o ' +
                         'WHERE s.occupation_id = o.occupation_id ' +
                         'AND s.status = \'HUNGRY\' ' +
                         'AND occupation = \'Mortician\''
has_many :ducks,
         :class_name => "Duck",
         :finder_sql => 'SELECT d.* from ducks d ' +
                        'WHERE status = \'FAT\' ' +
                        'AND flock = #{self.number} ' +
                        'AND TYPE=\'Mallard\' ' +
                        'AND orientation LIKE \'south\''
alias_column "square_footage" => "cubits"
#unicorn breaks rails
exclude_column_name "unicorn"
```

LLAMA

LLAMA

http://www.sxc.hu/photo/651432

User Interface Integration

- Transparent proxy
  - Noodle
  - Transproxy and Squid
  - Single Sign-On Keys

Did It Work?

- Quick time to market
- Maintainability
- Testing
- Reduced line code

Repurposing Old Data

- Maps
- Metric
- Project Tracking
- Testing

Where does Rails fit?

- Rapid prototyping
- Mission typical tasks
- Cost analysis
  - Development time vs. hardware

Is it really ready?

- Greater focus on Rails in Apache
- Ruby VM
  - Jruby, Yarv, Parrot,....
- User commented documents
- Cookie-less sessions
- Composite Keys
  - http://compositekeys.rubyforge.org

Questions?

Fin.

Questions?

Resources Du Jour

- Resources can also be found via this del.icio.us account: http://del.icio.us/vonage_railsconf_2007
- Noodle Proxy
  - http://noodle.tigris.org/
- Transproxy
  - http://transproxy.sourceforge.net/
- Squid
  - http://www.squid-cache.org/