

# Ruby on Rails Projects

## Dan Sharp – Sharp Internet Solutions

(970) 672-0337 (phone)  
dan@sharpinternetsolutions.com

(206) 337-1398 (fax)  
www.sharpinternetsolutions.com

### Purpose

This document highlights the Ruby on Rails projects that I have worked on over the past three years. The goal is to describe in sufficient detail the projects themselves and the technologies used in order to showcase my experience in Ruby on Rails and depth and breadth of my web application development expertise.

### Breadth of Experience

I have worked across all aspects of the Rails architecture for the past 3+ years. I have used and developed in Rails on Windows, Linux boxes and Macs and am comfortable with any of these three platforms for Rails development. I prefer OS X plus TextMate but am comfortable on all three platforms.

Most of the projects were individual undertakings, but I have also team-programmed on more than one project. I have used SVN and Git for source control and Capistrano for deployment. I have set up and managed deployment solutions both as stand-alone dedicated servers and via shared hosting services in combination with my web hosting company: Sharp Hosting. I have installed, configured and managed Apache + mod\_rails, Lighty and Mongrel clusters.

I have kept up with current Rails and Rails-related developments. I tend to stick to the latest stable release, primarily because I focus on project work using stable, deployable technologies. Thus, I do not live on Edge Rails and only adopt beta/edge technology/plugins when necessary.

I am not adverse to modifying or monkey-patching source code, either in Rails or in plugins, when necessary. Some patches were done in older Rails versions to support features before they became available. I have enough software architecture experience from previous employment that I am able to see and build “the big picture” just as well as diving into the nitty-gritty code. I enjoy both aspects of software development: architecture/design and development/implementation.

I also have experience taking an idea and building a web-based business around it. Two of the projects developed into companies (one S-corp and one LLC). These involved partners and business decisions and aspects beyond simple code development. I am comfortable working with marketing, business and sales people in regards to technology projects/solutions. I work well on my own or on a team. I am not a manager, however. I like to “get my hands dirty”.

The one area, I admit, where my experience is less than desired is in the area of testing. All of these projects have undergone extensive testing, but usually “live-style” testing, meaning: trying everything out in the browser. I have written some unit and functional tests and have started to learn some of the testing frameworks and ideas such as TDD/BDD. I am definitely for testing and hope/plan to learn more and apply more in this area for future projects and Rails ventures.

### Projects

The projects I have been involved with that utilized Ruby on Rails technologies are listed here. Greater detail for each project can be found below.

- **adsOOH LLC** – a digital signage / Out-Of-Home marketing services company
- **Xprove, Inc.** – a post-production review and approve services company
- **The FORC** – a community site for Regulatory Insurance law
- **The Peterson Page** – a members-only job posting community
- **RV Focus Law** – a document management site for the RV industry
- **TookABook** – a book sharing service
- **Cheers!** – a helps/service management project

### Access

There are varying levels of demo/live access to these sites. Some have public-facing pages. Some require administrative access. Some have staging/demo servers. Some have screencasts. And some never launched. If you are interested in further details from any of these projects, let me know and we can arrange access or code samples.

# *adsOOH LLC – a digital signage / Out-Of-Home marketing services company*

www.adsOOH.com

adsOOH LLC

## **Project/Company Description**

adsOOH LLC is a web-based middle-ware company conceived in early 2008 to offer services to the digital Out-Of-Home (OOH) industry. In short, adsOOH allows digital signage networks to sell their advertising “inventory” (on-screen ad-space + run-time slots) to local businesses, an opportunity previously ignored as too cost prohibitive. adsOOH provides a branded web portal for selling and managing video commercial ad buys. It involves an extensive amount of programming and leverages many of the latest Rails technologies.

adsOOH is a three-tier system, offering unique and secured access to (1) networks, for managing their inventory and sales, (2) producers, for managing video production and sales, and (3) customers, for managing their ad purchases and video deliveries. Customers access the system via branded, subdomain-managed “portals”, one for each digital signage network.

A full-featured demo site showcasing all three tiers has been developed. These run as live transactions except for (a) no actual payments are made... they are passed through as test transactions, and (b) no actual emails are sent... they are stored and displayed via a custom email “previewer” mechanism.

## **Technologies Used**

Built and deployed using the latest Rails at the time and recently upgraded to Rails 2.2.2 (with 2.3.x upgrade coming very soon). Source control is handled via SVN originally but upgraded in mid-2008 to Git using a remote Git repository. Deployment via Capistrano 2.x to shared hosting server running Apache 2.x and mod\_rails + MySQL 5.0.x. Developed using TextMate on OS X and collaborated via Skype. Some Rails and supporting technologies leveraged for this project:

- Rails 2.2.2 and beyond
- MySQL 5.0.x
- Apache 2.x + mod\_rails on shared hosting (for demo site)
- SVN + Git for source control
- Capistrano 2.x for deployment
- Screencasts created via SnapzPro & IShowU and QuickTime Pro
- PHP uploader + mod\_rewrite for large-file uploading with status
- Active Scaffold integration and customization for administrative toolkit
- Active Merchant integration and customization via BrainTree for Merchant Processing
- User management via restful\_authentication + role\_requirement
- Subdomain management for fixed + custom subdomains via subdomain-fu
- Video conversion to flash video via spawn + ffmpeg conversion script
- SSL support for payment pages via ssl\_requirement
- Google Maps integration with custom JavaScript + JSON for custom on-the-fly maps
- State machine via AASM gem
- Dynamic filtering mechanism – networks can create filterable categories on-the-fly
- Unit + Functional tests + custom bootstrap system
- Saas, Haml and jQuery used in some places
- Custom form builders
- Extensive use of Partials and Subtemplates across all tiers
- Quad-tiered namespaced routes: Network, Producer, Customer, Administrator

## **Challenges/Innovations**

The adsOOH system has, by far, been the most complex and extensive Rails project I’ve undertaken to date. It has it all: multiple tiers of access, user management, financial transactions (via Active Merchant), Google Maps integration, Ajax, video upload + conversion, jQuery, Saas + Haml, messaging, file attachments, bank transactions, credit card transactions, reporting, process flow management via state machine, user levels + roles, dynamic filters, searching, sorting, pagination, and more. Nearly 7,500 lines of code across 120 classes!

# *Xprove, Inc. – a post-production review and approve services company*

www.xprove.com

Xprove, Inc.

## **Project Description**

Xprove, Inc. is a web-based services company launched in 2006 to provide online video review and approve services to post-production companies. It was the first Rails project undertaken and began prior to Rails 1.0. Xprove was built as a team of three using shared programming practices across diverse geographies. The application was launched on a dedicated server and continues to service hundreds of clients to this day.

## **Technologies Used**

Built and deployed initially using Rails 1.0 and later 1.2.x. Developed using TextMate on OS X and collaborated via SVN and Skype. Some Rails and supporting technologies leveraged for this project:

- Rails 1.2.x
- MySQL 4.1.x
- Lighttpd (Lighty) deployment
- Web uploader integration for large-file uploading
- SVN for source control
- Capistrano 1.x for deployment
- Prototype
- Custom Merchant API integration for Credit Card support (reference storage only)
- Custom Mime Type support
- Layered layouts
- Custom cron scripts
- Ruby shell scrips for administrative tasks

## **Challenges/Innovations**

Xprove was built before and as Rails as a framework was just beginning to emerge into the “public eye”. There were a number of implementation challenges due to the newness of the technology. We built dynamic subdomain support before Rails officially included it using custom route management and filters. We integrated a PHP-based uploader with hand-shake integration between Lighty and Apache. We created and integrated custom mime-type support before Rails fully supported mime-types. We built a custom merchant API to interface in real-time with our merchant account provider, offering credit card purchases and reference-based account storage (no CC numbers stored).

# The FORC – a community site for Regulatory Insurance law

www.forc.org

The Federation of Regulatory Counsel, Inc.

## Project Description

The Federation of Regulatory Counsel, Inc. (FORC) hired me to take their existing Joomla!-based site and completely rebuild it to support a much greater set of document management features. The first pass was completed in 2007 using early Rails technology. In late 2008-early 2009, the FORC again contracted me to upgrade a number of features of the site. This description includes the latest round of development.

The FORC site provides public access to Journals, which include multiple articles by multiple authors, Alerts, which include short “blurbs” organized by state/region, Events, such as upcoming board meetings or special seminars, and Member search/display using a graphical map of the US + member lists. People can sign up for various mailing lists to be able to receive the journals, alerts and event notices via email.

The internal management portion of the website is called “The Toolkit”. This provides access for FORC members to create, manage and publish articles, journals, alerts and events. There is a member management component whereby members with appropriate access rights can manage the member list. Recently, a discussion forum based off of Beast was integrated for member discussions and all-member notices. The email system was improved to include HTML-styled emails. The Events system was upgraded to include file attachments and iCal (.ICS) integration.

## Technologies Used

Built and deployed using Rails 1.0.x originally and redeployed using Rails 2.2.2. Developed using TextMate on OS X. Source control managed via git and remote git repository. Deployed using Capistrano 2.x to a shared staging Apache server with mod\_rails. Some Rails and supporting technologies leveraged for this project:

- Rails 2.2.2
- MySQL 5.0.x
- Apache 2.x + mod\_rails
- Capistrano 2.x
- Prototype + jQuery + Ajax
- Custom modifications of ar\_mailer + spawn for large-volume email distribution
- File attachments via attachment\_fu
- Graphical calendar implementation via calendar\_date\_select
- ICS/iCal implementation via icalendar plugin
- Custom Paginator implementation (this was pre-will\_paginate)
- User authentication via custom authentication (this was pre-acts\_as\_authenticated)
- TinyMCE integration for “fancy text areas”
- Email notification of signups
- Uploaded file management & storage
- Beast integration and customization with tiered access levels
- Custom TinyMCE alteration to support in-line references converted to end-notes
- Complex access control and user levels
- Dynamic list creation, alteration and ordering
- Auto-PDF generation using HTMLDOC and custom layouts/mime types

## Challenges/Innovations

This was a site that involved retrofitting technology improvements around an existing visual design. The conversion of Joomla! pages to Rails-delivered pages involved some headaches and challenges. This project involved a lot of new and interesting Rails technologies, each with their own challenges. The two biggest challenges were: (1) building a TinyMCE plugin to support end-note references. Authors could select text in an article, flag it as an end-note via the TinyMCE button, and upon publishing of the article, the reference text would be stripped and auto-organized as end-notes. Various people have access to the publish functionality, allowing them to mark Journals, Alerts and Events as “published”, thus making them visible on the public site as well as sending them to the appropriate mailing list. One particular challenge was taking the original project code, built on Rails 1.0.x and upgrading it to Rails 2.2.2 throughout the codebase.

# *The Peterson Page – a members-only job posting community*

www.thepetersonpage.com

The Peterson Page

## **Project Description**

The law firm Baker & Daniels LLP desired to create a members-only community for former members of the administration of Mayor Bart Peterson. The design was retrofitted from Baker & Daniels site design templates. The site provides user profiles, user listing, job postings, admin features and multi-format views, including PDF, Spreadsheet (CSV), and full-vs-compact listings. Future expansion plans include event/calendar management, forums, custom email templates, and more. The admin features allow for management of user profiles and job postings by admin-flagged users.

## **Technologies Used**

Built and deployed using Rails 2.0.2. Developed using TextMate on OS X. Source control managed via Git and deployed using Capistrano 2.0 to a shared Apache server running mod\_rails at Sharp Hosting. Some Rails and supporting technologies leveraged for this project:

- Rails 2.0.2
- MySQL 5.0.x
- Apache 2.x + mod\_rails
- Capistrano 2.x
- Git + remote git server
- PDF auto-generation via RailsPDF
- CSV auto-generation via FasterCSV
- Pagination via will\_paginate
- Custom formats for PDF, CSV, and “full list” displays
- Ferret and acts\_as\_ferret for search functionality
- User authentication via modified restful\_authentication plugin
- Email via Observers
- Custom testing with data loader for bootstrapping data
- TinyMCE integration for “fancy text areas”

## **Challenges/Innovations**

The existing design had some challenges in that it leveraged some very convoluted CSS styling and did not have good multi-browser/multi-platform support. The other challenge was getting PDF and CSV generation to work well. I chose RailsPDF for PDF generation and FasterCSV for CSV generation.

## *RV Focus Law – a document management site for the RV industry*

### **Project Description**

The law firm Baker & Daniels LLP hired me to upgrade their document management site for the RV industry: [www.rvfocustlaw.com](http://www.rvfocustlaw.com). This site offers a searchable/filterable list of documents to members. The prior site had been implemented using Joomla! but was too cumbersome for people to use effectively. I proposed rebuilding the site in Rails and mapping a design similar to the design (at the time) of the Baker & Daniels website. The RV document management site never launched due to internal management issues at Baker & Daniels. However, the project was completed and the site remains working to this day on my staging server.

This project involved a heavy amount of Prototype and custom JavaScript/Ajax work. The searching, filtering, and displaying of documents, organized by category and state, is handled via Ajax as much as possible. A user can multi-select categories and/or states to select documents that match filter criteria. Users can sign up to request access to the system. Administrators can manage pending user accounts and approve or deny them access. Documents can be inserted into the system either as uploaded documents or using TinyMCE to enter document content.

### **Technologies Used**

Built and deployed using Rails 2.0.2. Developed using TextMate on OS X. Source control managed via SVN and shared across 2-person team. Deployed using Capistrano 1.x to a shared staging Apache server calling FastCGI at Sharp Hosting. Some Rails and supporting technologies leveraged for this project:

- Rails 2.0.2
- MySQL 4.1.x
- Apache 1.3.x + FastCGI
- Capistrano 1.x
- Prototype + Ajax
- Nested Layouts plugin
- Custom Paginator implementation (this was pre-will\_paginate)
- User authentication via acts\_as\_authenticated
- TinyMCE integration for “fancy text areas”
- Email notification of signups
- Uploaded file management & storage

### **Challenges/Innovations**

As with The Peterson Page, there were some challenges in using an existing design. Furthermore, the user interface for selecting states and/or categories required extensive Ajax work and testing to make it work right. We implemented the system using three user classes: users, admins and superadmins. The users could browse documents. The admins could manage documents and manage user accounts. The superadmins could convert a user to an admin, as well as delete/modify admin accounts. At the time, the best user management plugin was “acts\_as\_authenticated”. We used this plugin, with some modifications.

# TookABook – a book sharing service

www.tookabook.com

TookABook

## Project Description

TookABook is an idea spawned from some online forum discussions regarding book sharing. In a particular forum that I frequent, there were some threads where people suggested the possibility of posting books they had and people in the forum could request them. There was a lot of enthusiasm for the concept but no real sense of how it would work.

So I thought about a web-service for book sharing. Combining social-networking with book distribution and a very modest profit model. Basically, the system would be branded for each “community” and within that community, members could post book lists and wish lists. When a match (book-to-wish-list) is made, the requestor pays a modest fee to cover shipping costs and a very small commission (to TookABook). The owner of the book then ships the book (or books) to the requestor. Upon receipt, the money is credited to the owner to reimburse them for shipping costs.

Effectively, this becomes a business model not unlike Amazon’s used book sellers except (a) it’s within tighter communities and (b) it’s not profit-based for the book owners... it’s sharing-based. It’s the geographically-disperse version of letting your friends browse your bookshelves and say “that one looks interesting... can I borrow that?”

To date, the service has not launched, primarily because my time was taken with other ventures and the development is not complete. However, I still have high hopes for this service, in spite of a few other similar offerings already out there.

## Technologies Used

Built and deployed using Rails 2.0.x. Developed using TextMate on OS X. Source control managed manually. Some Rails and supporting technologies leveraged for this project:

- Rails 2.0.x
- MySQL 5.0.x
- Prototype + Ajax
- Subdomain support for “communities”
- User management via restful\_authentication
- Transition states handled via AASM
- Amazon/ECS integration

## Challenges/Innovations

Most of the challenge of this project revolved around modeling the interactions between book owners and book requestors. For example, how should the system handle the transition of the book from requested, to request acknowledged to book sent to book received and such. And when should the financial transactions take place? I also spent some time integrating the Amazon/ECS API for ASIN lookups. This way, book information is immediately accessible simply by entering the title and author or the ISBN number.

# *Cheers! – a helps/service management project*

(no website)

Cheers!

## **Project Description**

Cheers! Is another idea for a web-based service that I began to develop. My wife started a service ministry within our church whereby she gathered people willing to help in a variety of skills/tasks and paired them with people in the church or community in need of help. As the ministry took off, it became more difficult to manage the data set of “helpers” with “helpees”. What if there were a web-based application to handle the data management and the needs-pairing? Thus was born the Cheers! Project. As with TookABook, this is a project that received significant development interest but has yet to be completed and launched.

The system I built started as a deep customization of Active Scaffold. I learned all the ins-and-outs of the AS APIs and how to properly set up model interactions. I built a nice administrative interface that allows a “helps manager” enter helpers, enter helps requests and match helpers to requests based upon dynamic categories.

Ultimately, the vision of the project is that this system could be portal-customized for non-profit organizations and used to manage the full spectrum of a “helps” ministry, including monetary donations. Using Active Scaffold as a starting point allowed me to quickly figure out the objects and interactions within the system.

To date, the service has not launched, primarily because my time was taken with other ventures and the development is not complete. As with TookABook, this idea still has great potential, perhaps as an ASP model of a branded portal-like service offering to churches and other service-oriented organizations.

## **Technologies Used**

Built and deployed using Rails 2.0.x. Developed using TextMate on OS X. Source control managed manually. Some Rails and supporting technologies leveraged for this project:

- Rails 2.0.x
- MySQL 5.0.x
- ActiveSupport
- Prototype + Ajax
- Subdomain support for “portals”
- User management via restful\_authentication
- ActiveCalendar + jsCalendar

## **Challenges/Innovations**

By far, the biggest challenge was understanding ActiveSupport. It is a very robust technology for model interaction and “administrative-style” UIs. The API at the time was in active development and not as well-documented as I would have liked. Thus, there was a lot of source browsing, forum discussions and trail-and-error to get AS working as I envisioned. AS is also very Ajax-heavy. I found a number of tools to debug Ajax/JavaScript issues.