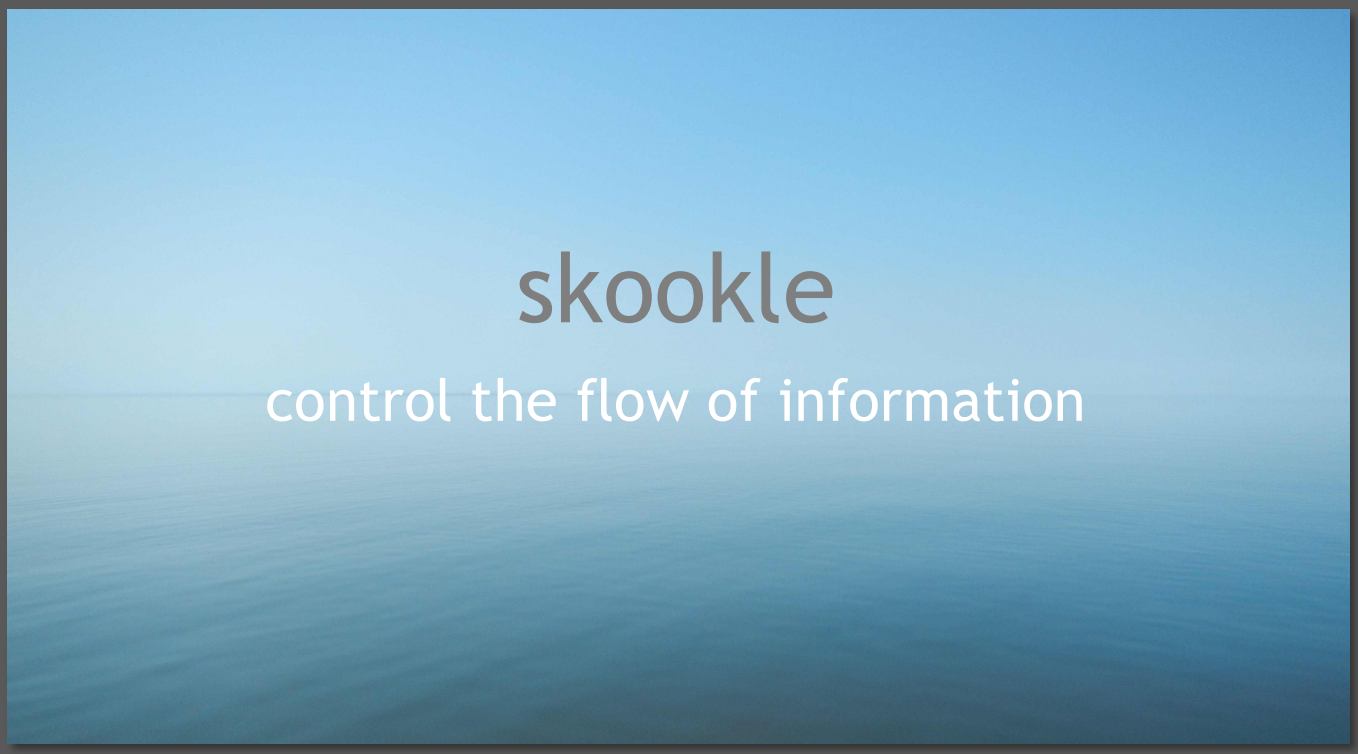
Brian O'Neill

1191 Randy Dr.
Pottstown, PA 19464
phone: 215.588.6024
email: bone@alumni.brown.edu



PHILLY ENTERPRISE HACKATHON 2013

The Need

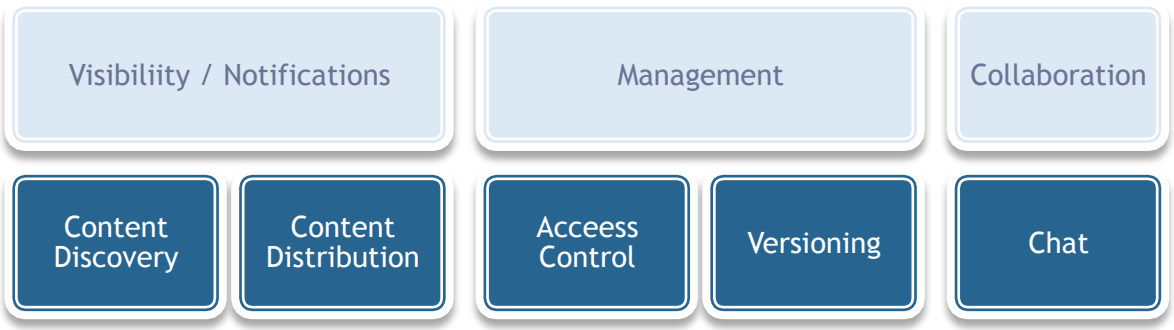
Content Management is critical to the success of any brand. Traditional content management systems focus on the artifact repository. They focus on workflows and taxonomies, enforcing rigid, often hierarchical, structures for storage and heavyweight processes around the artifacts. This often results in overly complex systems with cumbersome user experiences.

Additionally, traditional content management systems focus on *internal* content. That is content generated by the enterprise, ignoring the increasing amount of content generated by external sources. As customers continue to use social media sites for brand research, the external content is increasingly valuable. Tapping into that stream of content and incorporating it into a management system is critical. It allows brand managers to echo the positive sentiment, and directly address any negative content.

The Approach

Skookle takes a different approach from traditional content management systems. It focuses on the *flow of the information*, enabling users with simple, fundamental primitives for content management. It removes the hard constraints of rigid taxonomies and processes, and seamlessly integrates content from multiple sources, while enabling seamless collaboration.

Skookle leans heavily on today's cutting edge technology to provide:



The User Experience

Sometimes it is best to imitate, rather than create. Tools like Dropbox, and sites like Twitter and Youtube have found ways to address the least common denominator, allowing people with limited technical background to collaborate with others in very sophisticated ways. Skookle blends the concepts from these sites and others to deliver a simple user interface with simple but powerful concepts.

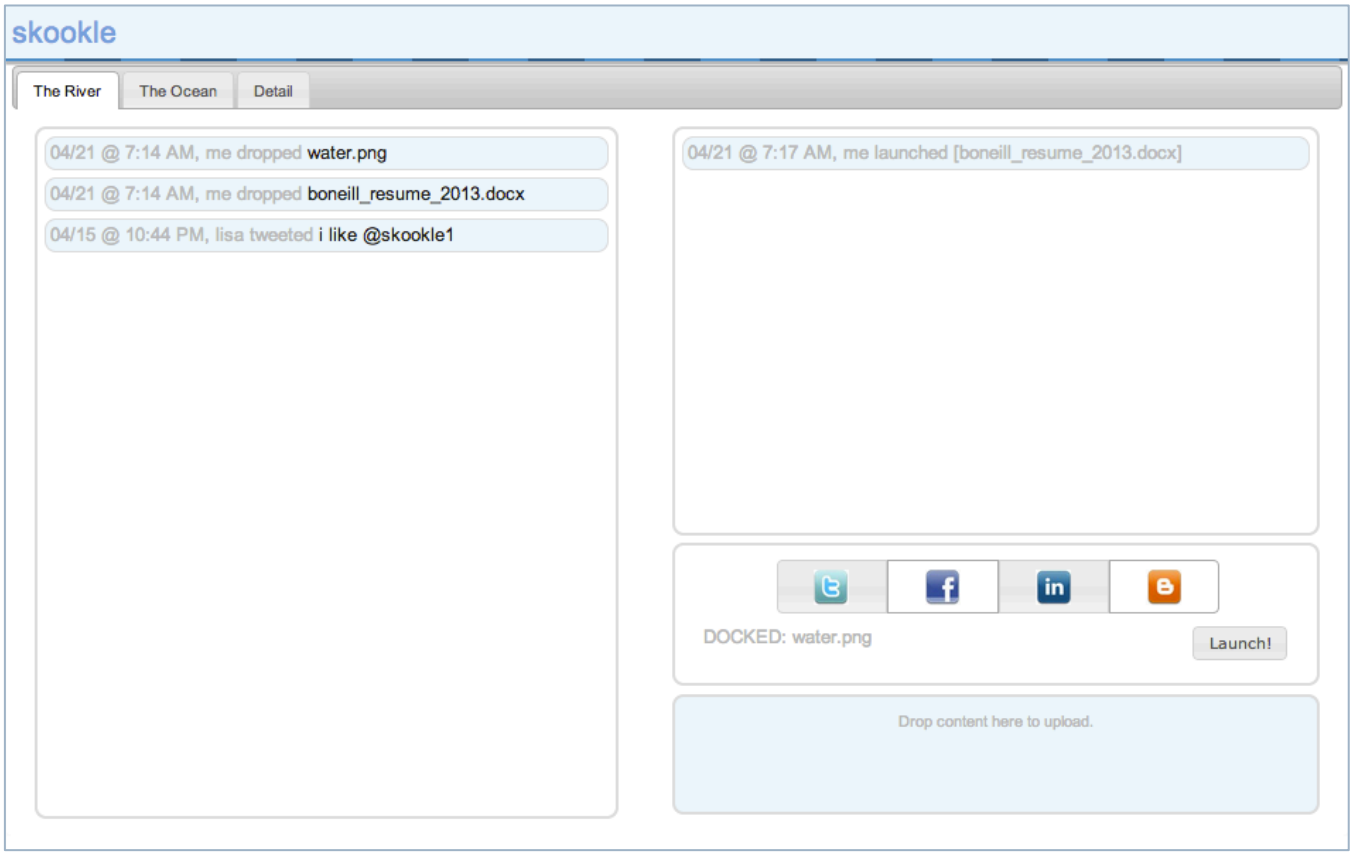
Skookle uses a river metaphor. (Skookle, like Schuylkill, get it? =)

An online screencast is available here that demonstrates the user experience. The screens are also described below.

The River

The left-hand side of the main screen displays events and content happening "upstream" from the user. Events may come from anywhere. The prototype includes a twitter stream. Whenever @skookle1 is mentioned on twitter, that data is received and added to the upstream event stream. Then, every user using Skookle CMS would see what was said.

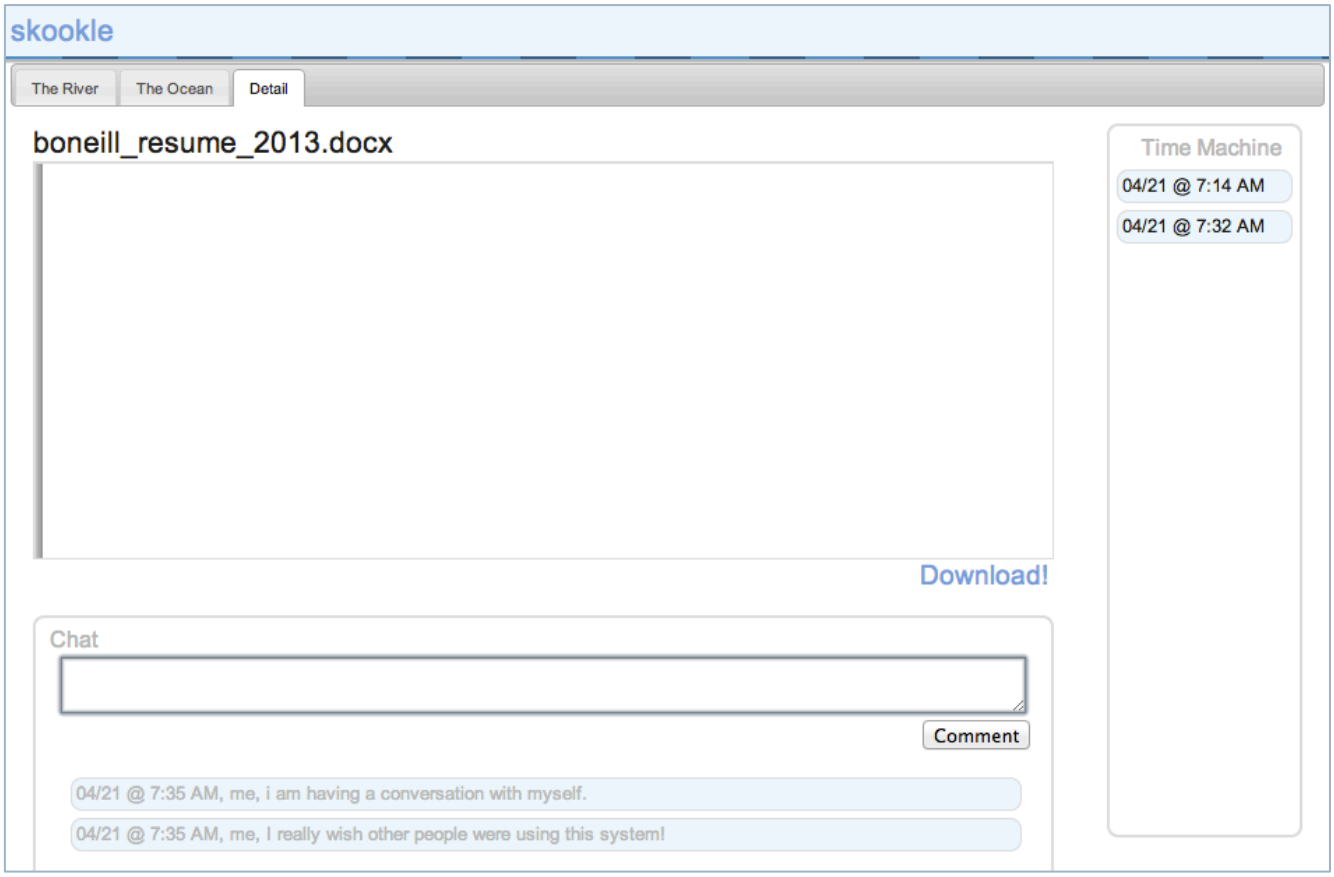
Below is a screenshot of the main screen:



Similarly, the right-hand side of the screen is downstream. Users can contribute content, and launch it downstream. They do that simply by dragging files onto the box in the bottom right. That immediately uploads the file to the server, and "docks" the content for launch. The box switches to a text box that allows the user to type in a quick description. (just like a tweet) Then the user can select which downstream channels should receive this content. In the prototype, when the user cicks the *Launch* button, Skookle tweets a link to the content out through Twitter.

Content Details

Once a piece of content is dropped into the repository, all users see that the document was dropped via their upstream window. They can then click on that link which takes them to the actual content. The content screen imitates a Youtube screen:

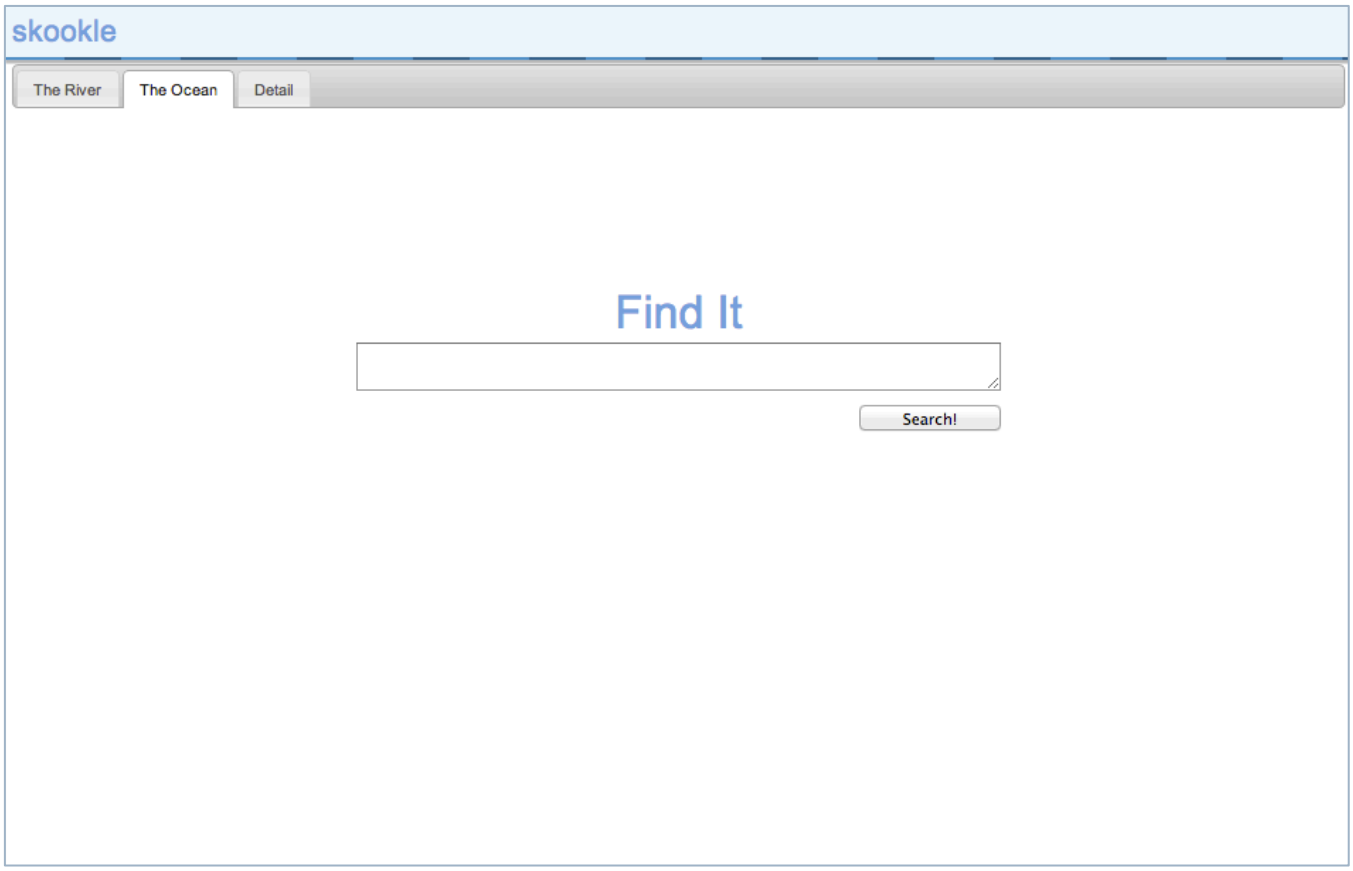


The top left quadrant contains a preview of the content. (not quite working in the prototype) Since often a user only wants to read through the content (and not edit it), it helps to have an embedded preview. Below the content is a persisted chat. Much like users can comment on videos on Youtube, Skookle users can comment on content.

On the right-hand side of the screen is the *Time Machine*. Skookle never deletes content. Each new drop is simply a new version. Clicking on any version in the *Time Machine* takes you back to that version of the content. Eventually, this screen would be enhanced with other metadata such as: number of views, last accessed date, etc.

The Ocean

Extending the river metaphor perhaps a bit too far, the screen that allows users to search for content is called *The Ocean*. In the prototype, the search screen is simple, providing only a single text box for the user:



Eventually this screen will also include a dashboard of the most active documents, tag clouds, and other visualizations that quickly summarize the content activity inside and outside of the enterprise walls. (e.g. a heat map of activity by geospatial coordinates would be valuable)

Hash Tags for Categorization and Access Control!

For those unfamiliar with hash tags, they are simple words added to tweets that categorize messages. Instead of a hierarchical folder structure for content, Skookle leverages a hash tag approach to categorize and filter content. Although it is not complete in the prototype, users would be able to filter their upstream events using hash tags.
(for more information on hash tags see: https://support.twitter.com/articles/49309-using-hashtags-on-twitter)

Additionally, Skookle leverages hash tags for access control. A system administrator would assign the hash tags to which a user has access. This is a powerful mechanism for fine-grained access control. (e.g. #exec might be added to content that only executive management should read)

Use Cases

The Skookle Content Management System provides the same fundamental functions as a traditional content management system, but it also incorporates additional functionality that allows brand managers, marketing, and sales to monitor and incorporate external content into their activities.

Brand Manager:

A brand manager needs to stay tapped into the activities of many organizations. From the main River screen, a brand manager can monitor all the events and activities from the various organizations:

- As marketing materials evolve, the brand manager will be able to see new versions of materials, and see the evolution of those materials through the time machine.
- Using simple conventions, such as hash tags for compliance (#compliance) and region (#europe), a brand manager would be able to quickly assess the state of compliance artifacts in a specific global region!
- Similarly, since skookle is designed with integration in mind, any system might generate events/content. An expense management system might flag a practitioner and/or sales organization if expenses are violating federal compliance laws. (e.g. Sunshine Act)

Marketing:

Marketing is continually looking for valuable use cases and events in the ecosystem. They want positive sentiment to reverberate through all marketing channels, and they want to directly address any growing negative sentiment.

- Marketing would monitor the upstream panel for events of interest. Those events might include positive or negative experiences with the product (e.g. tweets or blog entries, or posts in industry forums). Marketing could then respond to negative sentiment at its source, or re-tweet/echo positive sentiment across additional channels.
- Not limited to Marketing, but in general, users would evolve their content via Skookle. They would leverage the chat mechanism to collaborate on the content, *dropping* new versions into the system whenever there was a significant change in the version.

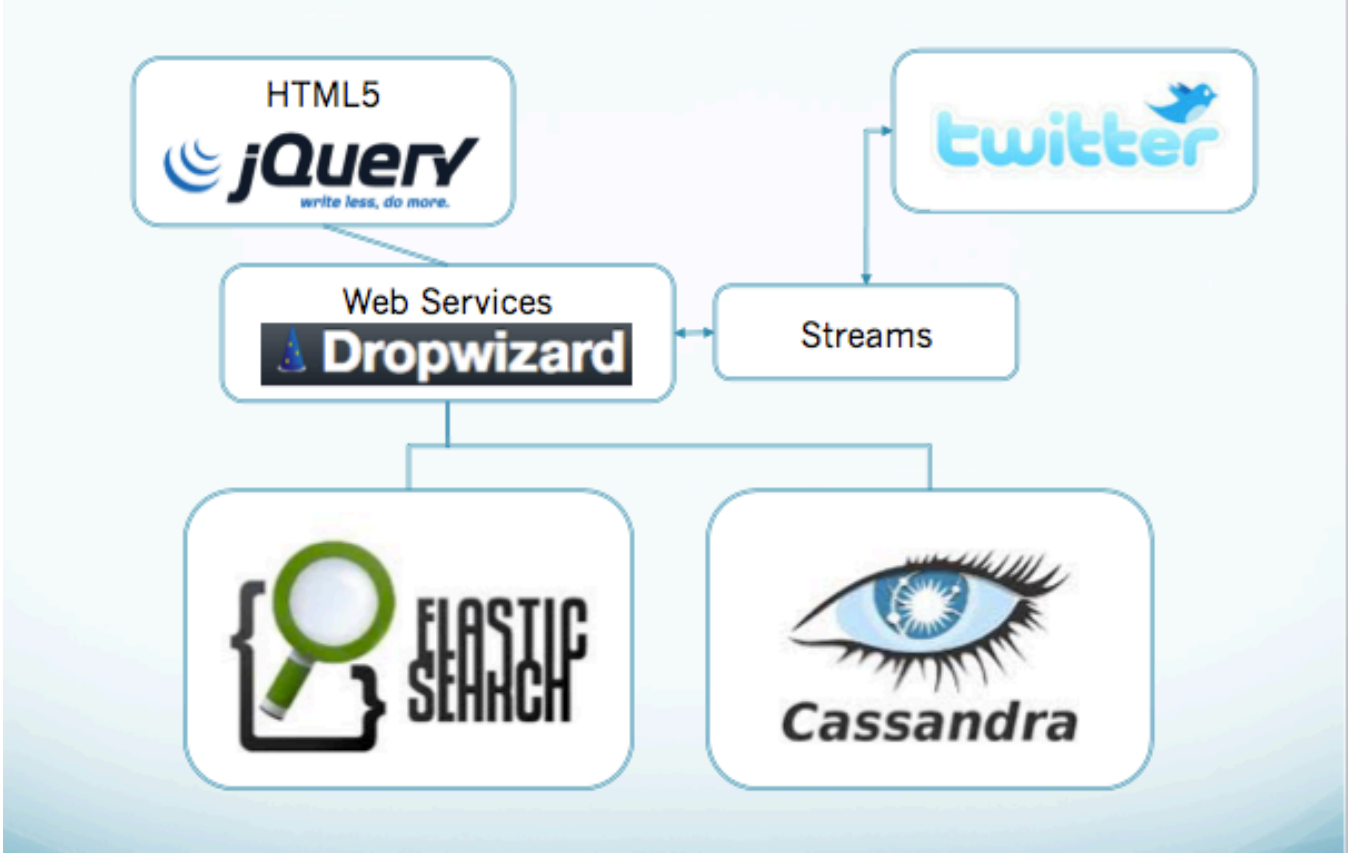
Sales:

Sales organizations are trying to determine where to apply resources.

- The sales manager, responsible for sales representative allocation, might monitor the geospatial heat-map within their territory to address sentiment in their area.
- Furthermore, Skookle can be integrated with prescriber eligibility systems that can indicate whether or not a specific practitioner is transitioning to a competitor's drug. Indications might be tagged with #switch, and sales teams could filter and search on that hash tag to find practitioners that are switching, and arrange one-on-ones with them.

The Architecture

The prototype is built with enterprise scalability and integration baked in. The high-level architecture is shown below:



Web Services

The front-end is a very lightweight single-page web application. It leverages HTML5 and jquery to provide a slick user experience. All the communication with the server is done asynchronously using lightweight web services developed using DropWizard. The services themselves are entirely stateless, which allows the system to scale horizontally. Additionally, since all the business logic is encapsulated in web services, additional user interfaces might be developed. (e.g. Native iOS application)

Storage

The documents and data are stored in Cassandra, which is a scalable distributed database easily configured to replicate data across hosts and data centers. This provides fault tolerance at the storage layer for high-availability and disaster recovery.

For some performance metrics on Cassandra see:
http://techblog.netflix.com/2011/11/benchmarking-cassandra-scalability-on.html

Search / Indexing

Along side Cassandra, Elastic Search indexes the data. This provides all the fuzzy search capabilities to which users are accustomed. (Misspellings, edit distance, etc.) Like Cassandra, Elastic Search is fully distributed. Indexes are spread across multiple machines providing fault-tolerance and redundancy in the case of a failure.

Content Opacity

The system makes very few assumptions about the content being entered. Except for the embedded preview, content is simply an opaque set of bytes. This means that anything, quite literally, can be dropped into the repository. Users can drag and drop urls, or drag events from upstream, to downstream, to launch them through additional channels.

Integration

Furthermore, using links as content, other content management systems can be integrated easily. For example, from a systems perspective Sharepoint is no different from Twitter. It generates events around content, and would be an upstream and downstream channel just like Twitter. Using this approach, Skookle can integrate other Content Management Systems, providing a central information hub across the disparate content sources.

Production Readiness Plan

To evolve the prototype into a production ready system would require the following work items:

Item	Description	Estimate
Authentication and Authorization via LDAP	Assuming most organizations will want to tie this into their existing Active Directory or LDAP server, the platform will need to accommodate LDAP.	40-80 hr.
Implement hash tags filtering for upstream content.	Although the user can include hash tags in their description, which are immediately searchable, they need to be able to filter their upstream events by hash tag.	20-40 hr.
Polish of the User Interface	The user interface needs to be polished. It is largely functional, but it may even need to be themed per implementation dependent on client requirements.	40-80 hr.
Additional Streams	Twitter is a simple integration. For the system to realize its potential value, additional systems would need integration. Ideally, a partnership with a company like SimpleReach would be established, which would discover and provide the relevant content.	20-160 hrs.
Cloud Deployment (?)	The system is ready for cloud deployment, but time did not permit the use of AWS for deployment. Alternatively, the system could be deployed on premise in customer datacenters eliminating the need for this line item.	0-80 hrs.
Content Preview	The content preview screen is not fully operational. It is largely dependent on the file type. The thought was to implement a PDF conversion (PPT->PDF, DOC->PDF). Then, integrating a PDF viewer into the browser should be a manageable task.	40-160 hrs.
Test Automation	To support additional rollouts, testing should be automated.	20-40 hrs.
Access Control	Once the system is integrated with LDAP, the admin screens need development, which will constrain which tags a user can assign to content, and what content they	80-160 hrs.
Cross-Browser Testing and Development	The prototype was only tested with Chrome. To deploy into production cross-browser compatibility needs to be tested and fixed where necessary.	20-80 hrs.
	TOTAL	280-880 hrs.

Why I should win?

I live at the intersection of passion, creativity, and execution. I love this stuff. I wrote my first program when I was seven. By the time I was fourteen, I was running a bulletin board system (BBS) out of my bedroom. I've been doing this ever since. You will not meet anyone with more passion for technology.

Professionally, I've done everything from startups to Fortune 500 companies, working in Artificial Intelligence and Natural Language Processing, Voice over IP, Web Services, and most recently Big Data. I understand what it takes to deploy large enterprise scale software, and I have a track record for delivering. (For more information on me see: http://about.me/boneill42)

Presently, I am the Chief Architect for Health Market Science, where I focus primarily on employing Big Data technologies to address the problems of Master Data Management (MDM) for the healthcare space. That is not too dissimilar from the challenges of content management. Our MDM platform is capable of ingesting thousands of data feeds from government organizations, Internet harvesting, and other private data sources. The system then correlates data across those feeds and consolidates the (often contradictory) information, into a single masterfile for the healthcare market.