FINAL REPORT

August 26, 2014

HHSF223201310098C

openFDA: A Pilot Research Project To Evaluate How Best To Make Datasets Available Via a Web Portal

Thomas Goetz
Iodine Inc.
34 Clyde Street
San Francisco, CA 94107

415-935-3463
thomas@iodine.com

Contents

Executive Summary

The openFDA project began with a mission to make it easier for the public - in particular software developers and researchers - to access, understand, and use public FDA data. Spearheaded by FDA's Chief Health Informatics Officer, Dr. Taha Kass-Hout, openFDA was launched to create easy access to public data and to highlights projects using these data in both the public and private sectors to further regulatory or scientific missions, educate the public, and save lives. In September 2013, Iodine began work as developer for the project, under a 12-month-long BAA research-and-development contract.

Working as a small team under a one-year deadline and with a federal government shutdown looming, we worked hard to identify the data, the technological processes, and the design approach that would result in a successful project. Thankfully, we benefited from a mission-driven team at FDA, and a responsive and cooperative agency overall.

We successfully engaged with various FDA centers to understand the public data available to us. We developed a process for building on this data, through harmonization techniques and documentation that would make it more useful to outside organizations. In particular, we used a taxonomy that the public would understand - Food, Drugs, and Devices - rather than an internal method that would be non-intuitive to outsiders.

After just eight months of evaluation and development, we successfully launched the open.FDA.gov website with the first API, offering up the FDA's drug adverse events data (referenced as FAERS internally) to outsiders. This one API combines (or harmonizes, in technical terminology) several FDA datasets to create a more useful and informative resource. Since that release, we have developed three more APIs, for recalls, labeling, and medical device adverse events. In just two and a half months, there have been more than 2.6 million API calls, and more than 30,000 unique visitors to the open.fda.gov site.

In addition, openFDA has served as a test bed for the FDA to try various technologies, including open source and Amazon AWS cloud resources. Elasticsearch in particular has been an excellent fit for FDA's datasets that contain free text and several new projects leveraging the technology internally at FDA are now being considered.

OpenFDA was conceived of and operated as a research and development project. In particular, the effort has been to learn about how best to develop and release data, and how to engage a community of users around that data. In these 12 months, we believe openFDA has successfully delivered on all fronts, while providing a foundation for a great deal more to learn and do.

Goals of openFDA

The OpenFDA initiative was launched in September 2013, as part of the larger Office of Informatics and Technology Innovation roadmap. Conceived under the leadership of Taha Kass-Hout, FDA's Chief Health Informatics Officer, openFDA began with the purpose of offering developers and researchers easy access to high-value FDA public data. The main goal has been to make it simple for an application, mobile, or web developer, or all stripes of researchers, to easily use data from FDA in their work.

As developers, our role in openFDA has been to leverage best practices in data science, cloud computing, community management and content management to create a flexible, robust platform that can serve both agency priorities and spur both citizens and the private sector to use openFDA datasets for innovation and experimentation.

Early on, we realized it would be essential to engage with the public without assuming any expert knowledge of the FDA's internal structures. Rather, we proposed to offer the public tools that mapped to a taxonomy they would intuitively understand for the FDA: food, drugs, and devices.

We also proposed to use open source software throughout, and to release all products as open source, with the hope of tapping the developer community in a dialogue about how openFDA could be developed and serve an ever-larger pool of needs and objectives.

Lastly, as an R&D project, the intention was to serve as a testing and development platform for the agency, to engage broadly with various agency partners, and to disseminate our approach and learnings widely within the agency.

Methodology

With just a year to develop, plan, and execute a multi-product/multi-platform project, we knew the clock was ticking as soon as the contract was signed. Accordingly, we adopted an iterative and centered methodology and set about conducting interviews with FDA stakeholders to identify potential datasets. We then set about identifying a handoff strategy with various contacts inside FDA, to ensure that the data was delivered in best-possible form. Then our engineering team set about working with the data and developing the API protocols.

Meanwhile, our design team got to work designing a clean interface and UX that would present the data in a useful, engaging, and approachable context. In addition, there was backend logistical work to be done in arranging for the hosting and serving of data, as well as external work in contacting developers and other potential interested outsiders, to ensure that upon release the APIs had been tested by real users with real use-cases.

These seven steps are elaborated upon below.

1. Choosing the datasets

Beginning in September 2013, the openFDA team met with several groups inside FDA, including teams from CDRH, CDER, CTP, CFSAN, and ORA. After explaining the goals and strategy behind openFDA, we discussed possible available datasets. Working with the FDA team, we then evaluated these datasets to identify those that A) had high value to the public, B) were in a publicly available form already, and C) might be improved upon by easier access and/or more intuitive API search capacity.

We fixed on eight datasets that offered promise. After working with the respective data owners inside FDA (see 2 below), we ended up eliminating two datasets and combining the other six into four discrete APIs.

2. Obtaining the datasets

Beginning in October 2013, we began working with various data owners inside FDA to better understand the data and the available formats. These conversations were largely

led by FDA's Steven Hubbard (a member of the openFDA team) and Sean Herron (a presidential innovation fellow who was on the openFDA team). In the course of these conversations, it became evident that some datasets were more ready than others, and that each would require a different approach to acquiring and working with. In general, the approach was to default towards the publicly available data, so as to ensure whatever appeared on openFDA is consistent with other public sources.

- The FAERS data were taken from the publicly available download. *http://www.fda.gov/Drugs/GuidanceComplianceRegulatoryInformation/Surveillance/AdverseDrugEffects/ucm083765.htm*

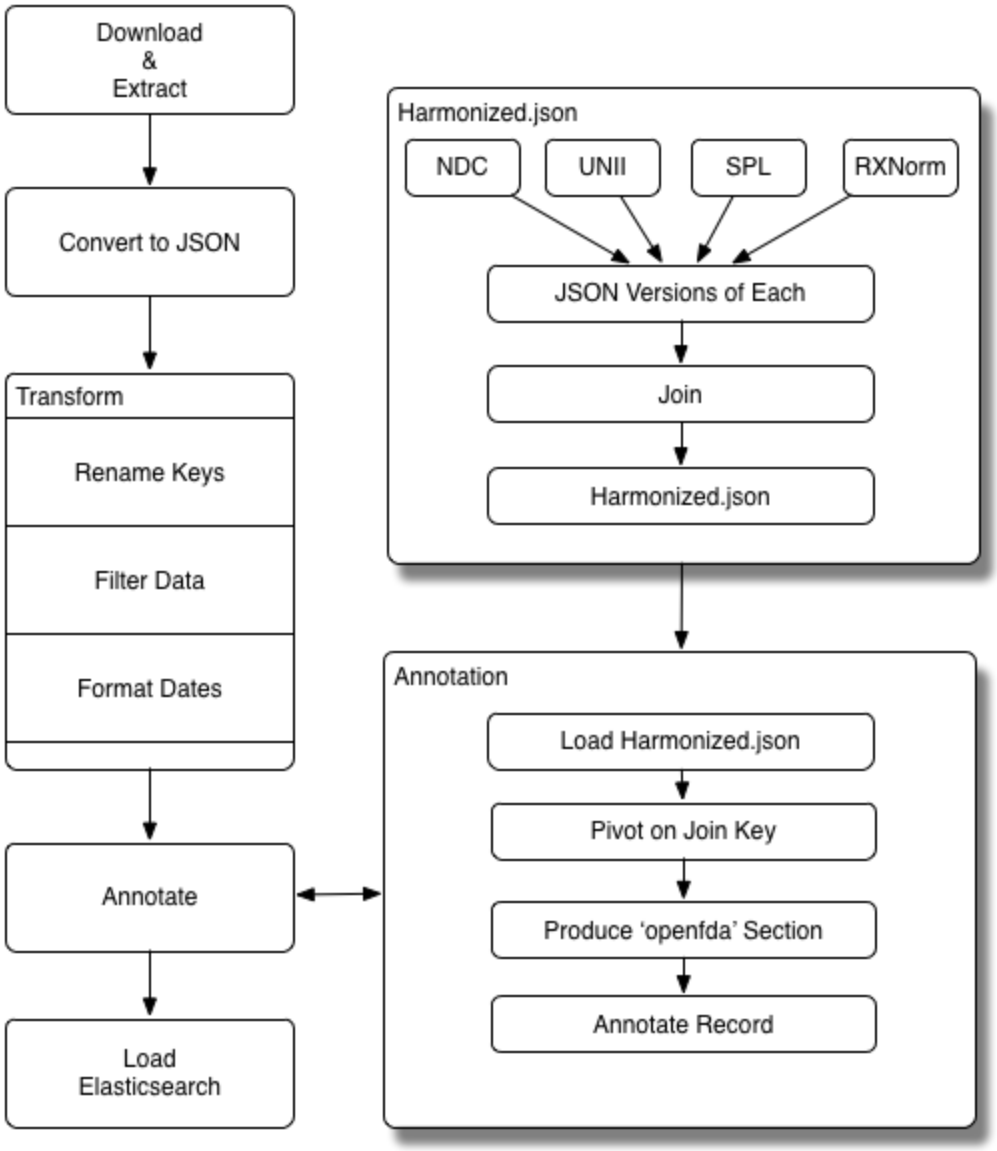
- The RES data were obtained using the publicly available HTML (we scraped the code) and public XML data (available starting June 2012). *http://www.fda.gov/Safety/Recalls/EnforcementReports/default.htm*

- The MAUDE data were obtained using the public FDA download source. *http://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/PostmarketRequirements/ReportingAdverseEvents/ucm127891.htm*

- The SPL data was obtained from the NLM's DailyMed download source. (Note that the FDA's SPL team is working on a method for making the SPL data available for download directly from FDA. When that is available it's possible to reorient the openFDA source to the FDA point.) *https://dailymed.nlm.nih.gov/dailymed/downloadLabels.cfm*

3. Harmonizing the datasets

Harmonization is a process of combining various datasets together so that they create an integrated, consistent and unambiguous resource. The purpose of harmonization is to bring complementary information together, in this case drug identifiers, so that it can be used to annotate another dataset. The harmonized data gets transformed into the 'openfda' section of any dataset for which there is a join key. The hope is these data can provide both a deeper insight into a single dataset and provide a common method of accessing drug data across APIs.



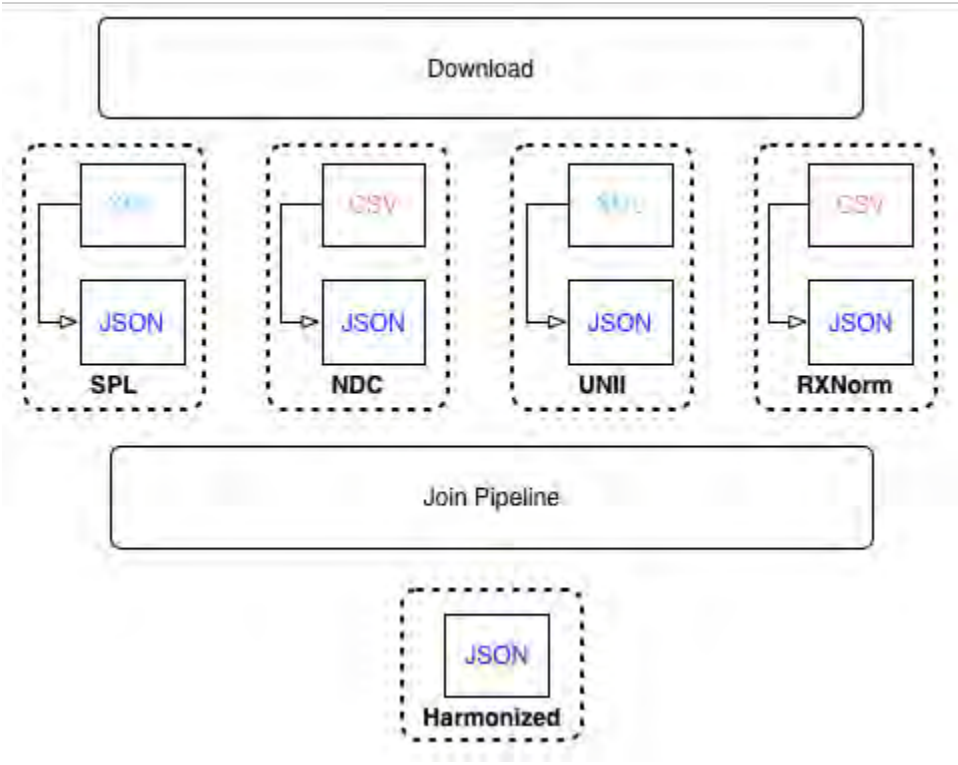
The harmonization processes is built upon four data sources:

A) NDC Product database file
B) Specific fields from the standard SPL download
C) SPL Pharmacological Class Indexing file
D) SPL RX Norm Mapping file

The harmonization process is much like all of the other pipeline processes; rather than loading Elasticsearch, it produces an intermediate dataset that is consumed by the annotation step within other pipelines. There are four major steps to the process:

1. Download all the raw data needed
2. Extract and convert each into its own JSON representation
3. Join each of the datasets together into a single representation
4. Write out that representation to a harmonized.json file

If there is a join key available within a dataset, then an annotation step is added to that dataset's pipeline, in which the harmonized data is loaded into memory, pivoted into the appropriate join key, reduced to a single record and then added to the dataset as an 'openfda' section. It is important to note that there may be many rows from the harmonized.json that apply to a record in a dataset. In these cases, the multiple rows are turned into a single row and the single values from each row become an element within a list in the 'openfda' section.



This harmonization process helps address the following issues for openFDA:

Issue 1: Inconsistent formats

Structured Product Labels are defined in an HL7 approved XML format. RES is available in XML currently and HTML historically. FAERS is available in XML currently and SGM historically and also requires use of ASCII files to properly filter the XML and

SGM files. NDC is available in XLS or CSV format. SPL indexing files are in XML format while SPL mapping files are in TXT format.

The harmonization effort converts all of these formats into JSON, so there is a standard way to access them via an API.

Issue 2: Inconsistent identifiers/Joins

SPL contains many structured identifiers but doesn't contain pharmacologic class (that's in the indexing files). FAERS has a structured application field and a mostly structured medicinal product field but doesn't contain any other drug identifiers like NDC. RES contains NDC and UPC but doesn't contain pharmacologic class or structured manufacturer information.

The harmonization effort joins together several structured FDA datasets, including SPL, SPL indexing files, SPL mapping files and NDC to form an annotation table that is then used to annotate records with additional identifiers not available in the native format. This enables API users to search different datasets with whichever drug identifier they'd like.

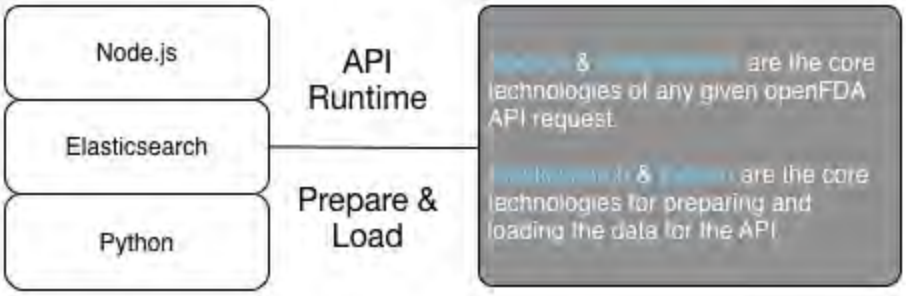
Issue 3: Free-text fields requiring structure

The recall datasets are largely free text but the free text typically contains NDC, drug name, manufacturer and other fields that would be useful in a structured format.

The harmonization effort builds regular-expression extractors for unique-looking strings like NDCs and UPCs, which can then be annotated via the process described in Issue 2.

4. Building the APIs

The APIs for openFDA were built using modern, open standards like REST APIs, JSON and Lucene query syntax and leveraging open source and cloud technologies. This stack includes:

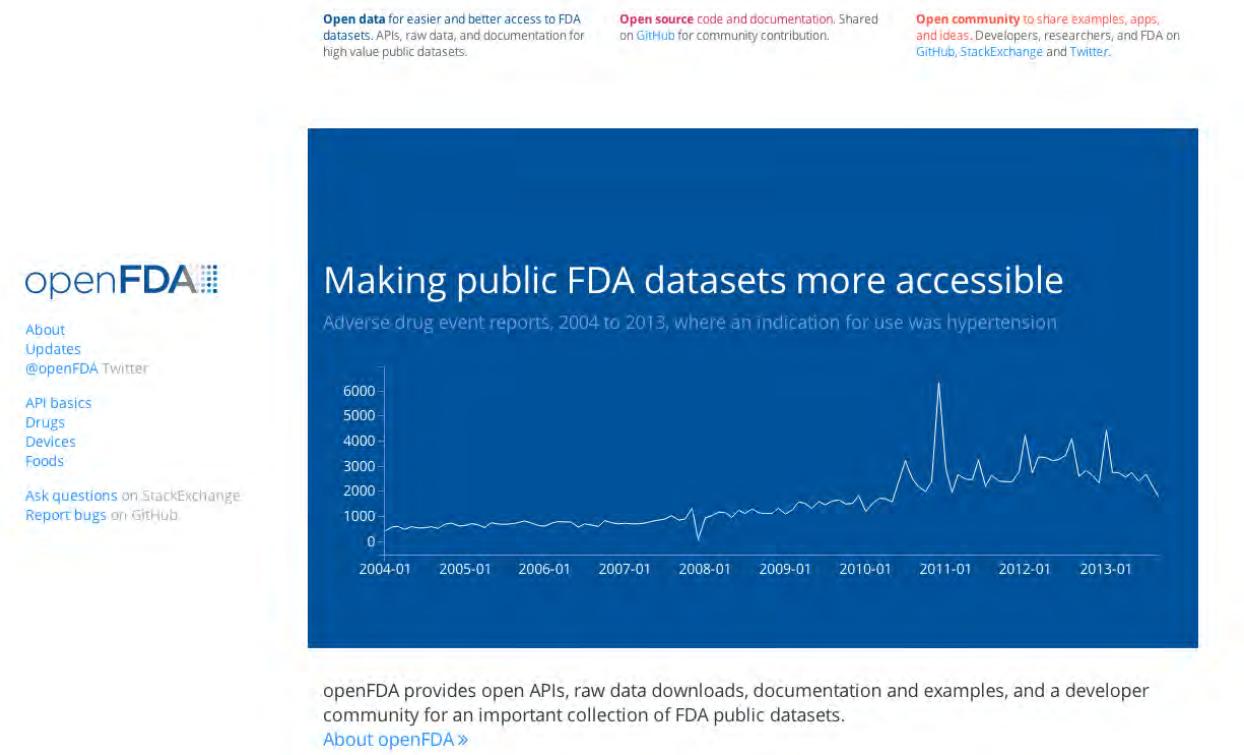


Elasticsearch: Elasticsearch is an excellent tool for building REST search APIs such as the ones employed by openFDA. Elasticsearch 1.0 was released in February of 2014 and 1.2 was released in late May 2014, a signal of the rapid development of this technology. Elasticsearch powers search for Github, Wikipedia, Asana and many other major companies and organizations. This also puts openFDA on the vanguard of implementing robust, scalable database technologies.

Node.js: Node has proven to be a great API server for openFDA. The event driven architecture combined with the fast V8 JS engine has enabled us to achieve well over 100 requests per second per process on Amazon EC2. Additionally, there are great libraries for connecting Node to Elasticsearch, including Elastic.js and Elasticsearch.js, both of which are employed for openFDA.

Python: We chose Python as the language for the data processing needs of openFDA, and have been very satisfied with this choice. JSON support is excellent due to the simplejson module and the open source Luigi pipeline system by Spotify enables us to create simple but powerful data pipelines for the transformation and loading of data into Elasticsearch. Finally, LevelDB (with Snappy compression) has excellent bindings in Python.

5. Designing open.fda.gov



The design of the open.fda.gov website draws on best practices in agile development, intuitive user experience, and data visualization.

The user experience (UX) for openFDA is based on the principle of providing one unified, simple presentation to users. We wanted to convey that the FDA is one cohesive agency, one that regulates drugs, food, and medical devices. By design, we wanted to not distract the user with the particulars of the FDA's internal organizational structure, or the idiosyncrasies unique to any one dataset.

We began by planning a lightweight, simple design that could be modified and improved with feedback. In other words, we didn't want to present one design as a *fait accompli* - we wanted to be able to respond to feedback, both from within FDA and outside the agency.

In turn, we wanted to create an interactive site that draws on real data, rather than presenting static information or simulations of data. We built the openFDA site using a combination of live, interactive programmer-friendly queries (which demonstrate to developers and researcher how to use the API) and visualizations and examples that help explain the nature of the data (what is available, what one can learn, and so on).

These were two sides of the same coin, one that demonstrates the power of the FDA data through clear explanation, context, and real demonstrations.

Importantly, we designed openFDA so that everything lives on one site: the data, API key generation, interactive examples, all necessary documentation to understand datasets, project updates, and so on. In other words, we didn't want to send users off the website for various purposes - with one great exception. Following best practices of the developer community, we leveraged StackExchange and GitHub - standard, community rich resources - that move the discussion around the openFDA project into the public, letting a community develop and allow learnings and insights to be spread among the community. The intent, in part, was to avoid having to answer a torrent of email directly, but also to use the resources that programmers are already familiar with and expect to be part of any well-conceived project.

Along the way, we made several specific design and technology choices that warrant mention.

- We built the site using a three-part information architecture of drugs, devices, and food. We believe this reflects the FDA's broad mandate to the public, and is also intuitive for the public to understand.

- We used a consistent appearance and format for all documentation and examples throughout the site, and built templates that facilitate addition of new datasets — might want to say that this echoes the harmonization effort in the data. This is effectively UX harmonization.

- We aimed to create as few pages as possible, to make it easier for users to find related information. For each API, we built no more than two pages.

- Where possible, we brought the documentation "in-house" and onto the openFDA site, rather than linked to and hosted elsewhere.

- We built interactive queries using real, live data drawn on the respective APIs. In addition, the API key signup happened in the context of the API information and documentation, making the path from interest to sign up much more streamlined.

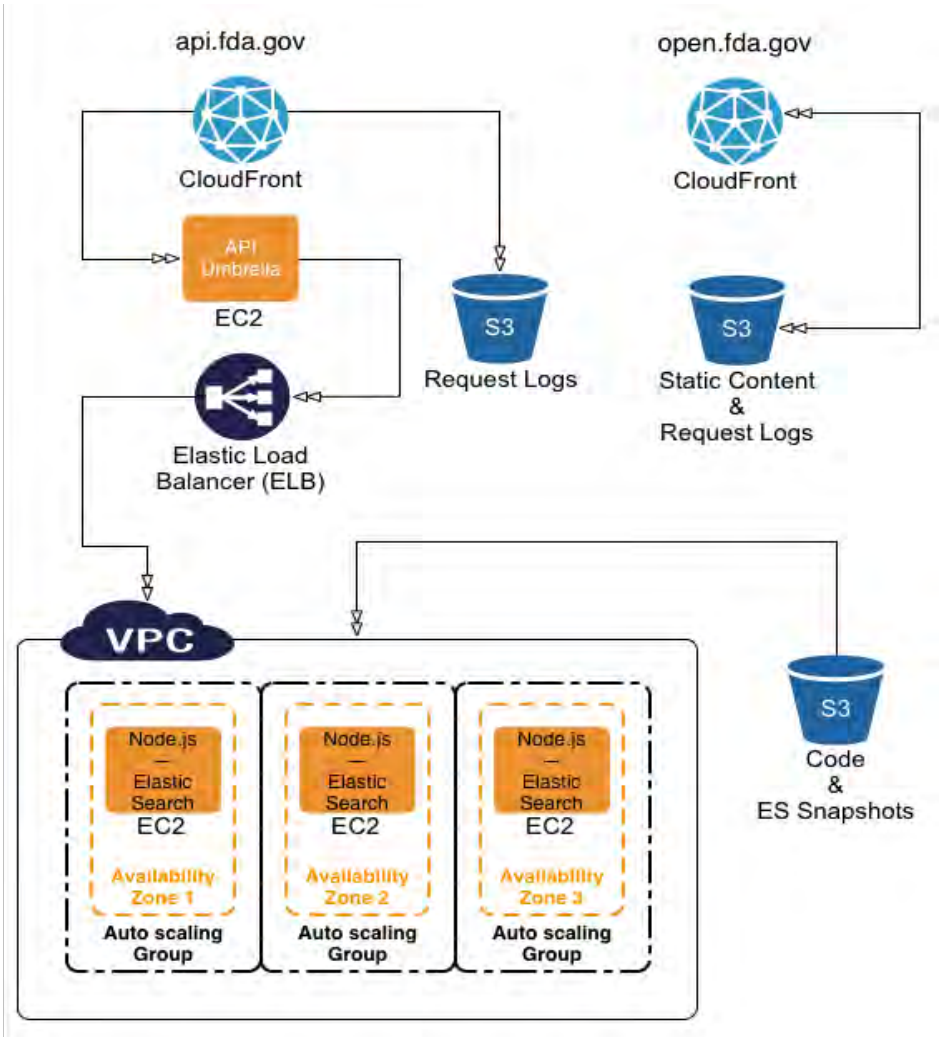
- Just as the visualizations change as queries are adjusted, it works the other way, too: The interactive queries change as the visualizations are manipulated. This is an innovative and powerful way to show both the macro and micro functionality of the data.

- openFDA is very much a customer of its own work. Any presentation of the data was created using the boilerplate API, without enhancement or modification.

- The entire site was built using open source technology, including Jekyll (which allows documentation to be written in markdown that anyone can edit, versus in more-complicated HTML), Bootstrap (which enables mobile use), Grunt (which optimizes our JavaScript and LESS/CSS), and D3 and C3 for data visualizations.

- Issues and feedback are managed on GitHub and StackExchange, as developers expect these tools (developers being an essential target audience for openFDA). GitHub, in particular, allows open-source contributors to submit documentation change requests directly, rather than by email.

6. Preparing the backend



OpenFDA makes use of the following technologies on Amazon Web Services (AWS):

EC2

OpenFDA currently utilizes the c3.4xl instance type, making use of the large memory capabilities and fast SSD instance storage for running Elasticsearch. Additionally, EC2 has excellent locality with S3 making Elasticsearch restores very fast. Finally, we make use of Amazon's multiple availability zones in US-EAST to provide fault tolerance against datacenter issues. Since launch on June 2, openFDA has had almost 100% availability as a result.

S3

Simple Storage Service (S3) is used for storing Elasticsearch snapshots, code releases and the open.fda.gov static content. By hosting the Elasticsearch snapshots on S3, we enable easier access to researchers and the press who commonly use S3 for transporting large files. By hosting the static content for open.fda.gov on S3, we minimize the maintenance burden of the site.

We evaluated the use of GitHub Pages for hosting the static content, as Jekyll templates also power GitHub Pages. However, there were several requirements that made GitHub Pages a poor fit, including the ability to stage a full site for review prior to setting live, the ability to host via SSL, and having raw level access to the request logs. Once it was determined that GitHub Pages was a poor fit, we decided to generate the static HTML locally using Jekyll and then push this content to S3. This setup meets all the requirements outlined above.

Cloudfront

Cloudfront is used as a CDN to reduce user latency; for caching to reduce load on the API servers; for SSL to protect API keys and the contents of the API requests; for logging in a standardized format to S3; and for protection against external attacks. Cloudfront is used to host both open.fda.gov and api.fda.gov.

Autoscaling

Autoscaling is used to ensure the openFDA EC2 instances are up and healthy. When an EC2 node goes down or is no longer healthy, autoscaling automatically starts a new

instance to replace the bad instance. In the future, we plan to make use of autoscaling load features to scale up or down depending on load on the API.

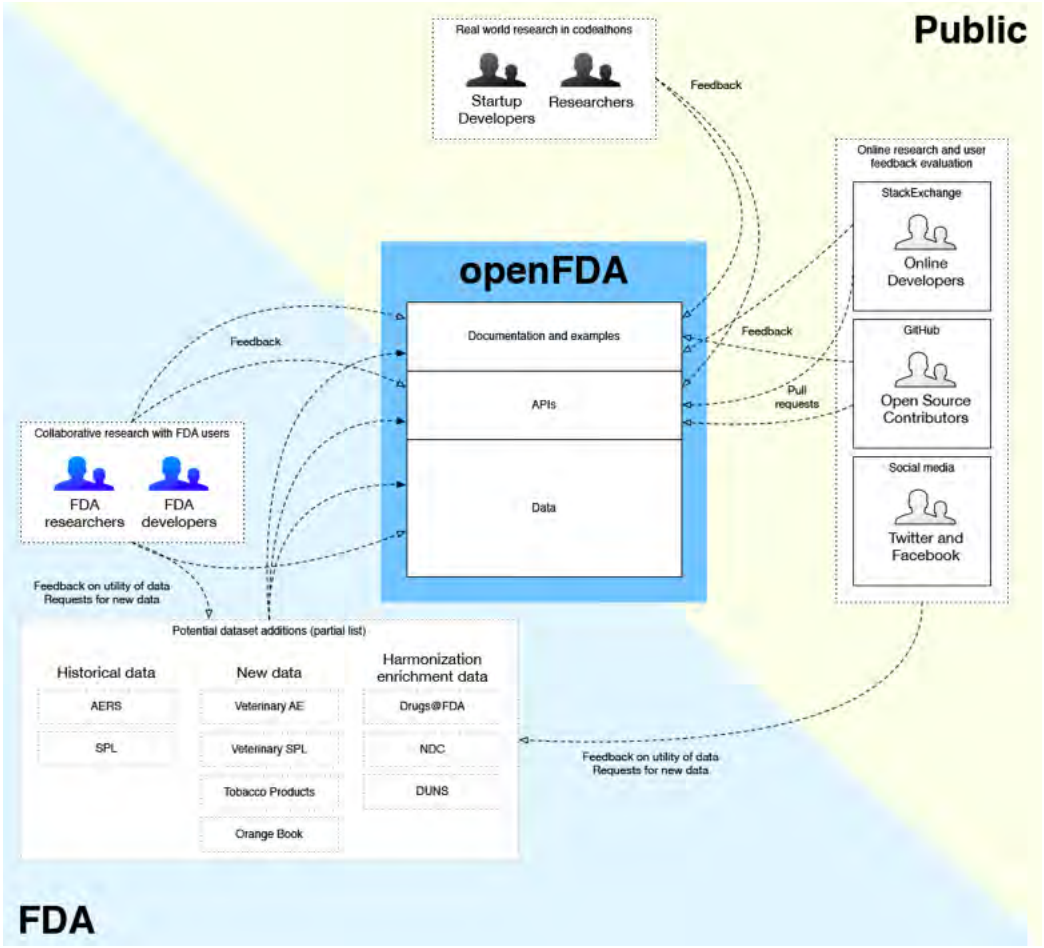
ELB

Elastic load balancers (ELB) are used to round robin HTTPS requests across the multiple API servers, across multiple availability zones. The ELB is where the SSL connection terminates as the ELB connects to these API servers within a VPC that's protected from external snooping. Finally, we use security groups with ELBs to limit access to the ELB to only api.data.gov (which provides quota services and is also hosted in US-EAST by GSA).

VPC

Virtual Private Cloud (VPC) is used so that no API server can be accessed directly from the external Internet. An SSH gateway server is used when access to the API servers is necessary. By using VPC in combination with the ELB, security groups and use of api.data.gov, this ssh port on this machine is the only publicly accessible port for all of openFDA.

7. Testing the APIs and community engagement



Following our iterative development approach, as the APIs became available for testing, we solicited input from a select list of “trusted testers.” This was a closed beta process with 67 individuals or organizations that had expressed interest in openFDA and were willing to contribute feedback before a formal launch.

Many of these beta testers worked with the API and half a dozen offered direct feedback at a Google Group set up for this purpose.

This trusted-tester review occurred distinct from an internal FDA review. Where the objective of an internal review was principally to ensure that the API was returning expected results and that the documentation regarding the dataset was in order, the trusted-tester review was meant as an early proxy for the demands and issues that might arise when the API was released to the public.

One especially useful round of feedback occurred in early May 2014, when a group of developers in the DC area was invited to an afternoon-long introduction session to the

FAERS API (the first API developed for openFDA). This turned out to be an essential part of the feedback and testing process, as we could see with our own eyes the frustration that some developers had in putting the API to use. As a result of this session, we radically revised the documentation plan around openFDA, and added an interactive visualization tool that let users more intuitively manipulate and query the API.

In addition, we have actively monitored the GitHub and StackExchange openFDA forums, responding to any requests for clarity or information. In general, we believe our outreach efforts for openFDA served the project well, insofar as they improved the product and reflected well on the project. The developer community responded positively to being brought into the process, regardless of whether any requested changes were made or not.

The outreach served the longer-term needs of openFDA as well. For the future of openFDA, we now have a roadmap of what users want, including linked data, access to the VAERS data, post-inspection reports, and other requests.

Challenges & Changes

In the process of building openFDA, we encountered a few challenges, requiring a few strategic changes. These are covered in the four categories below:

1. Adjustments to the dataset list

There were originally eight datasets selected for openFDA. As we began working with the FDA data owners, it became apparent that one dataset - the tobacco ingredients database - would not be completed in time for this contract. Another dataset - the CAERS data on adverse events from food products - was deemed not ready to release by FDA stakeholders (though we believe this dataset is still suitable for release in a subsequent phase of openFDA). Two datasets - the Drug Master Files and the Unique Ingredient Identifiers - were smaller and less compelling as separate dataset releases. The DMF data was deemed a poor fit for the project and the NDC data was substituted instead. With the approval of our FDA team members it was decided to combine the UNII data with other, more useful APIs (the FAERS and SPL APIs in particular).

2. Harmonization challenges

Some datasets needed more work than others and we've discovered that it's difficult to know all of these challenges in the planning phase.

For example, with FAERS we were aware of the challenges in joining adverse event reports with annotated information like NDC. We were not aware, however, that publicly available FAERS QDE downloads have problems in several SGM and XML files that prevent them from being parsed by a standard XML library. We've built transforms that run prior to the library so that the library can properly load these files.

We also weren't aware that because the API spans multiple quarters, FAERS requires a duplicate case report filtering process that involves indexing tens of gigabytes of data. We employed LevelDB to perform this filtering and it performs quite well. It's worth noting that this was discovered a month prior to launch due to communication challenges with the FAERS team. While we had met with many different members of

the team, we were not connected with the data owner of the QDEs until early May. Even after the filtering issue was identified with his help, the solution for historic (SGM-based) records was not apparent and had to be researched by him. In the future, we should connect with the technical data owners as quickly as possible and proactively solicit for information on transformation and filtering required by the dataset.

With RES, we were aware that much of the data was only encoded in free text fields and that we'd need to build regular expression based extractors to extract identifiers such as NDC and UPC. We are also aware that the publicly available enforcement reports were only available in HTML format prior to June 2012. We were expecting, however, to obtain these data in XML format from the RES team. However, due to concerns around PII and lack of resources in the RES team, we instead needed to build HTML scrapers for these historic HTML files and because these historic HTML files were created by hand using the FDA CMS, many different scraping heuristics had to be employed to handle the many different ways that data was entered.

With SPL, we were aware that the bulk downloads were only available from Dailymed. We were expecting, however, to obtain data from the SPL team directly in an S3 bucket that would be updated daily. Due to challenges navigating the security at FDA, the bucket has only recently been made available. As a stopgap, we implemented code for downloading these datasets from Dailymed automatically and checking their integrity as the NLM FTP servers can sometimes return bad data.

3. APIs and downloads

At the outset, the objective of openFDA was to develop both APIs and full downloads for respective datasets. After further review, however, and in consultation with the FDA team, it was decided to hold off on offering complete downloads for every openFDA dataset. The reasoning was three-fold: first, because a more useful, feature-rich API can create an ongoing relationship with developers and users, we wanted to avoid having downloads cannibalize users who might do one-off visits rather than sustained connections through an API (for instance, when people download instead of use the API, we lose the ability to see which fields in the dataset they find useful). Secondly, the infrastructure for consistent downloads was not yet complete by the time we started API releases. Third, we wanted to minimize confusion. After the FAERS API launch, many thought that this may change the process of release of QDEs; had we hosted a mirror of these files, people could have been even more confused.

Finally, the APIs are a foundation to providing specialized download tools for end users. These specialized tools could serve downloads of specific subsets of data (for example all SPL files from a particular manufacturer), in formats not natively supported by the dataset (like JSON or XLS) and to have filtering that is not part of the native download.

One app developed on top of openFDA by the community (www.researchae.com) offers that download capability. We believe this further validates our approach to keep focusing on the APIs primarily and evaluate individual S3 downloads (such as requests from AP) on a case-by-case basis.

Our recommendation for the download strategy in the future is to continue to encourage developers to use the APIs. As part of this effort, simple example download servers could be built for SPL, FAERS and other datasets where there have been requests for downloads. For datasets that are included in future APIs for which there are no current public downloads, a simple strategy for hosting on S3 similar to the one proposed in earlier reports could easily be employed.

That said, it might be appropriate to prioritize a capacity for data downloads, should openFDA continue.

4. Outreach to developer community

Community has been a priority for openFDA, for both philosophical and pragmatic reasons. Philosophically, it is a cornerstone that openFDA exists to serve a community of developers and researchers, and that it is only through direct contact with this community that we can best serve their needs. Pragmatically, openFDA is a scrappy project that lacks resources to actively monitor feedback boards and/or to serve individual requests or queries. An engaged community could help provide some of these resources and guide new users to success.

That said, any nascent community requires nurturing and contact. Our priority thus far with openFDA has necessarily been on creating useful APIs that serve the needs of outsiders. Though engagement with developers has helped inform the product, it has been more difficult to maintain a sustained level of successful engagement, as we've been busy pushing out new APIs and documentation. In the future, openFDA should consider emphasizing community outreach as a primary objective, rather than as an episodic means to an end.

Results

Nearly 12 months after it began, and just three months after the release of the first dataset, there are various metrics by which we can consider the success of openFDA.

Number of datasets: After reviewing dozens of potential publicly available FDA datasets, openFDA released four APIs, connected to highly valued, high impact datasets that capture the range of the FDA's mandate to protect and improve the public's health.

Number of API calls: In just three months, there have been more than 2.6 million API calls for various openFDA datasets. Considering that all of these data were publicly available in some form, at least, this statistic testifies to the appeal of a clear, well presented, and compelling resource for FDA data.

Developer uses: OpenFDA APIs are being actively used by Novartis, Social Health Insights, Reuters, Big Data Lens, and other downstream organizations for various purposes.

Developer contributions: Seven open source contributions to the open.fda.gov website (primarily improved documentation and bug fixes) have been made by three external developers. Two open source contributions (bug fixes) to the core openFDA project have been made by two external developers. Both of these contributions were supervised and admitted by the core openFDA team.

Queries: At present, there have been at least 27 questions submitted - and responded to by openFDA team members and external developers - on StackExchange (under the openFDA tag). At three months in, this counts as a promising level of engagement; many queries are from developers who are seeking guidance in how to build upon openFDA data in their own products and companies. In addition, we have had inquiries from Reuters, the Associated Press, Wells Fargo, and other organizations that are eager to put the openFDA data to work. Developer interest in enhancements and new datasets has been recorded and discussed by the community on GitHub. Nearly 100 developers have "favorited" the openFDA project on GitHub.

Internal FDA interest: We are happy to report there have been several groups inside FDA who, having seen the promise and potential of openFDA, are eager to work with

the project to bring new datasets to the public. In this, we believe that openFDA serves as a potential way for groups to get things done that, for budgetary reasons or technological capacity, was beyond their means previously. We hope that these leads can be developed as openFDA builds moving forward.

To date, we have used Google Analytics to measure usage of documentation and API Umbrella and custom scripts over Cloudfront logs to measure usage of the API. Key metrics for the documentation include total site visitors, average time on site and percentage returning visitors. Key metrics for the API include API calls by end point, average API latency and top API users. In the future, we believe it would be worthwhile to measure API usage at the field level. This would enable analysis like how commonly used is the openFDA section in each endpoint and how often are specific fields in each dataset accessed.

Conclusion

In the nearly 12 months since openFDA began, it has progressed from a white-board vision to a hub of innovation inside the FDA. Under the leadership of the Office of Informatics and Technology Innovation and Taha Kass-Hout, FDA's Chief Health Informatics Officer, openFDA has emerged as a new way for the agency to engage with the public, and offers new opportunities to bring FDA's data-rich resources to a wider audience with the potential for wider impact on the public's health.

Fortunately, the project's status as a research-and-development project has allowed the openFDA team to pursue a flexible, iterative development approach. We have emphasized direct contact and feedback with both internal stakeholders and external data-users, and that feedback has allowed us to optimize our plans along the way.

Significantly, it has also offered the FDA an opportunity to learn about and work with new technologies and tools, including open source software and cloud computing. These tools have demonstrated their aptitude for the FDA's needs, even in a security intensive environment. The result has been a lean and rapid development and testing environment that may serve the agency as it plans for its technological future.

OpenFDA has demonstrated that there is an appetite inside the FDA to release its data in more useful, more intuitive forms such as APIs. It has also demonstrated an external appetite for FDA data beyond the expected audience of regulated industries. The response to the first releases has been remarkably, if not entirely, positive, and has been taken as a sign that the FDA is on the vanguard of innovation in the federal government.

In all, openFDA serves as a new chapter in the agency's engagement with the public. It demonstrates how a sincere effort to release data in a useful form, and to engage with potential users in how to improve the form and context around that data, can yield dividends and aligns with the agency's mandate to serve the public's health.

This 12-month effort has spawned a great deal of promise. There is more to do with the openFDA initiative, but we have been honored to serve the FDA in helping accomplish so much in just 12 months time.

Glossary

API: Application Programming Interface, a set of instructions and/or standards that make it possible to access a website, software application, or dataset.

ASCII: American Standard Code for Information Interchange, a standard code for representing English characters as numbers.

C3: A JavaScript library for creating web-based, interactive data visualizations, built on top of the D3 library.

CDER: Center for Drug Evaluation and Research, a division of the Food and Drug Administration.

CDN: A Content Delivery Network is a system of distributed servers that deliver content to its users based upon the user's geographic location.

CDRH: Center for Device and Radiological Health, a division of the Food and Drug Administration.

CFSAN: Center for Food Safety and Applied Nutrition, a division of the Food and Drug Administration.

CSS: Cascading Style Sheets is a feature added to the HTML standard providing the developer with more control over how the pages are displayed.

CSV: Comma Separated Value File is a common approach to sharing data where each row within a file contains values that are separated by commas; it is typical for each row to contain the same number of commas.

CTP: Center for Tobacco Products, a division of the Food and Drug Administration.

D3: Data Driven Documents, a JavaScript library for manipulating documents based on data; helps bring data to life when used in combination with HTML and CSS.

FARES: FDA Adverse Event Reporting System, an FDA system for collecting and processing adverse events with respect to drugs.

Grunt: A popular automation tool within the JavaScript ecosystem, typically used in build and deploy processes.

HL7: Health Level Seven, refers to a set of international standards for transfer of clinical and administrative data between Hospital information systems.

HTTP: Hypertext Transfer Protocol, an application protocol for exchanging information the World Wide Web.

HTTPS: Hypertext Transfer Protocol with SSL Encryption, an encrypted version of HTTP, providing secure information exchange on the World Wide Web.

Jekyll: A tool for automatically converting markdown to HTML, typically used as a lightweight Content Management System (CMS).

JS: JavaScript, a popular programming language that's built into all the major web browsers and used to make web pages interactive.

JSON: JavaScript Object Notation, a simple way of representing data that is both human and machine-readable.

LESS: Less is a CSS pre-processor, meaning that it extends the CSS language.

LevelDB: An open source, on-disk key-value store written by Google that is used to store and retrieve data.

Lucene: Open source information retrieval library that is part of the Apache Foundation, providing a unique and effective approach to reverse-word indexing.

MAUDE: Manufacturer and User Facility Device Experience, an FDA system for collecting and processing adverse events with respect to devices.

Markdown: A text-to-html conversion tool for web page authors.

NDC: National Drug Code, a unique product identifier used in the United States for drugs intended for human use.

NLM: National Library of Medicine, established in 1836, it continues to be a leader in providing the world with biomedical information.

ORA: Office of Regulatory Affairs, the lead office of the FDA for all agency field activities.

RES: Recall Enterprise System, an FDA system for collecting and processing recall reports for food, drug, device, biologic and veterinary products.

REST: Representational State Transfer, an architectural style of designing API's that typically relies on the ubiquitous HTTP in order to exchange data; it is known for its simplicity and being lightweight.

SGM: Standard Generalized Markup (Language), a way of embedding data descriptions within the data file; it can be viewed as a predecessor to XML and it is no longer widely used.

SPL: Structured Product Labels, a document markup standard approved by Health Level Seven (HL7) and adopted by FDA as a mechanism for exchanging product and facility information.

SSH: Secure Shell, a cryptographic network protocol for secure data communication, typically used with command line tools.

SSL: Secure Socket Layer, a protocol created by Netscape to ensure secure transactions between web servers and browsers.

TXT: Text File, a typical file extension given to files that contain data that are either space or tab delimited.

UPC: Universal Product Code, a unique way of identifying products in several countries, including the United States, United Kingdom, Canada, et al.

UX: User Experience

V8: Google's open source, high performance JavaScript engine.

VAERS: Vaccine Adverse Event Reporting System, an FDA system for collecting and processing adverse event reports with respect to vaccines.

XLS: Microsoft Excel File Extension.

XML: Extensible Markup Language, a generalized way of embedding data descriptions within the data file in an attempt to make the data self-describing and independent of how it was created.