

Wiley 1.0

Software Design Document

***Table of Contents***

[Wiley Product Overview 3](file:///C:\Users\klinder\Desktop\Proposals\FDA\Deliverables\150623_Wiley%20Software%20Design%20Document%20v3.docx#_Toc423947462)

[Software Architecture 4](file:///C:\Users\klinder\Desktop\Proposals\FDA\Deliverables\150623_Wiley%20Software%20Design%20Document%20v3.docx#_Toc423947463)

[Graphical User Interface 5](file:///C:\Users\klinder\Desktop\Proposals\FDA\Deliverables\150623_Wiley%20Software%20Design%20Document%20v3.docx#_Toc423947464)

[Data Services 8](file:///C:\Users\klinder\Desktop\Proposals\FDA\Deliverables\150623_Wiley%20Software%20Design%20Document%20v3.docx#_Toc423947465)

[Databases 10](file:///C:\Users\klinder\Desktop\Proposals\FDA\Deliverables\150623_Wiley%20Software%20Design%20Document%20v3.docx#_Toc423947466)

[Technical Trade Studies & Key Industry Standards 10](file:///C:\Users\klinder\Desktop\Proposals\FDA\Deliverables\150623_Wiley%20Software%20Design%20Document%20v3.docx#_Toc423947467)

[Glossary 13](#_Toc423947468)

[Appendix A 14](#_Toc423947469)

The Wiley product is a web-based food recall monitoring application that allows food industry professionals to discover trends in Food and Drug Administration (FDA) food recall data published at openFDA.gov.

# Wiley Product Overview

The web application is developed entirely using open source software and is deployed in an Orbis Amazon Web Services (AWS) environment. Wiley was developed using Agile development practices and adheres to the U.S. Digital Services Playbook.

Key Wiley software components include the following:

* **Graphical User Interface:** A collection of web-based search, graph and map applications integrated into several dynamic web pages. The search window allows the user to search for recall content based on Time, Location (by country and US state), Keyword and Food Group, from which the latter does not exist as a standard FDA selection choice but was developed using custom designed semantic web technology. The search results are displayed geo-spatially on a map and in a tabular with index card frames. The results gain further interpretation through the use of dynamic bar, line and donut graphs.
* **Data Services:** Wiley incorporates Application Programming Interfaces (APIs) created by Orbis and provided by the FDA, but it also utilizes custom built APIs and data services that implements semantic search technology in order to perform data mining fact extraction as well as continuous monitoring services to monitor end user query behaviors over time as well as a tabular view of query responses.
* **Databases:** Wiley utilizes an industry standard open source Resource Description Framework (RDF) Sesame database that is decoupled from the data services and web application to provide expandable, quick result sets for user interface. Wiley also access the openFDA database and a small open source database to store end user queries.

The expected audience for this document include software architects, system engineers, software developers, test personnel, GSA 18F program technical representatives, and management.

The software architecture, shown in Figure 1, highlights the core architecture elements – Graphical User Interface, Data Services and Databases. The approach incorporates nine (9) open source software components and numerous industry standards,

# Software Architecture

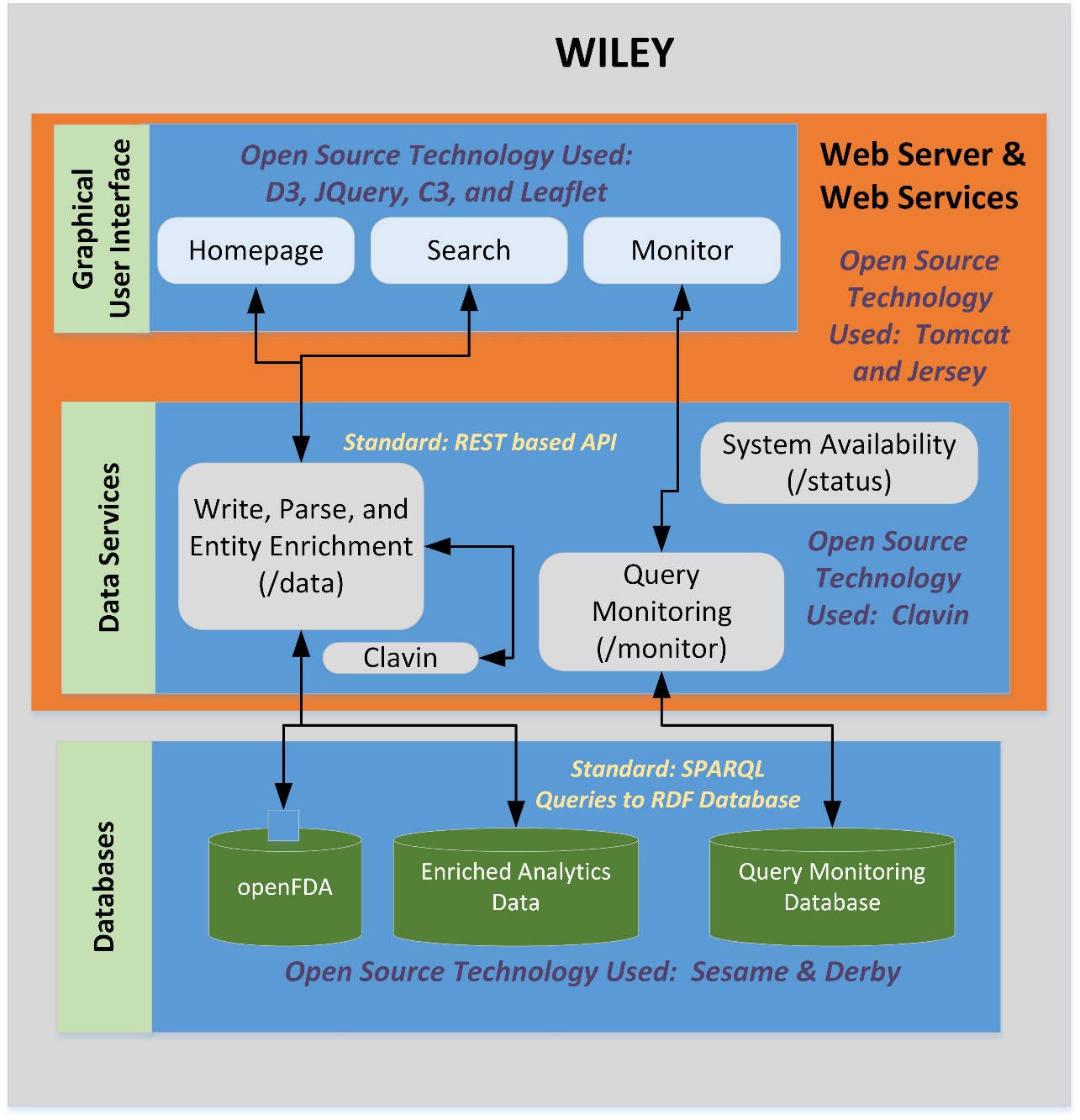


Figure 1: Wiley Software Architecture

**Technical Approach**

All of the project functionality and objectives are captured in a Project Charter. To achieve these goals, the Wiley design adopted three system architecture principles. First, Orbis products and solutions incorporate leading World Wide Web Consortium (W3C) IT industry standards to promote interoperability among applications and data. The W3C is the standards body responsible for many IT standards such as HyperText Markup Language (HTML) and eXtensible Markup Language (XML). By incorporating the adoption of industry standards, clients can affordably plan for future growth, and continue to adopt the most optimal solution available to meet evolving business needs.

Second, the solution will decouple the applications from the data. This approach is counter to many leading commercial products that provide “black box” products where the application only works over a proprietary database. By decoupling the components, the client can select applications a la carte, and modify only the elements needed for their business. Decoupling the solution at all levels (database, analytics and application) provides the client with the greatest amount of flexibility to tailor the solution to their individual needs, while simultaneously providing the lowest total cost of ownership over a five year period.

Finally, the third component to open architecture is the ability to use free and open source software (FOSS) and/or existing corporate IT products. This approach is critical because it allows many clients to reuse existing IT investments and/or retire expensive commercial products in favor of more modern freely available software. Whenever practical, FOSS is used to lower sustainment costs.

Wiley has a Graphical User Interface consisting of a three page dashboard configuration that includes a Home Page, semantic-driven Search Page and a Monitor Page that continually processes user queries. Both the Home Page and Monitor Page are dynamically driven to display query results as new user data is collected.

# Graphical User Interface

**Home Page**

The Home Page, shown in Figure 2, is the default page the end user will see upon opening the application. The Home Page includes an icon-driven Selection Menu, an Information Banner to provide an overview status report of the food recall database, a Line Graph to show the number of voluntary Food Recalls over time, a Bar Graph that shows a summation of the methods used by the FDA uses to notify the public about a Food recall, a Health Hazard donut graph to describes the percentage of recalls based on their relative degree of health hazard as defined by FDA (Class I, Class II and Class III) and a Recall Status donut graph to shows the status of Food recalls.

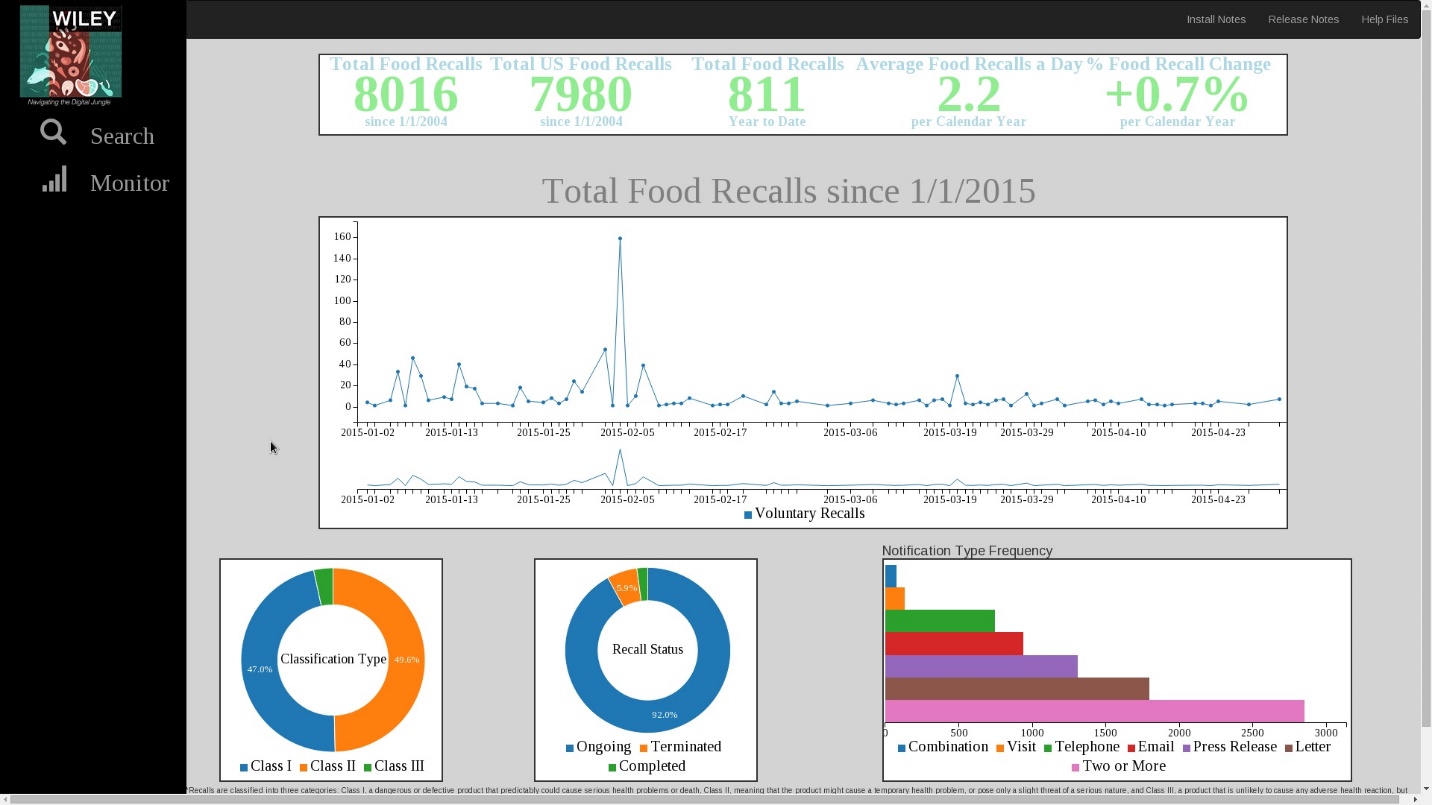


Figure 2: Wiley Home Page

First, the data used for the Home page is pulled directly from the FDA open data initiative (<https://open.fda.gov/>).

**Search Page**

The Search Page, shown in Figure 3, allows the end user to perform queries over the Food Recall database. The required search terms include Date Range and Location (state and country).

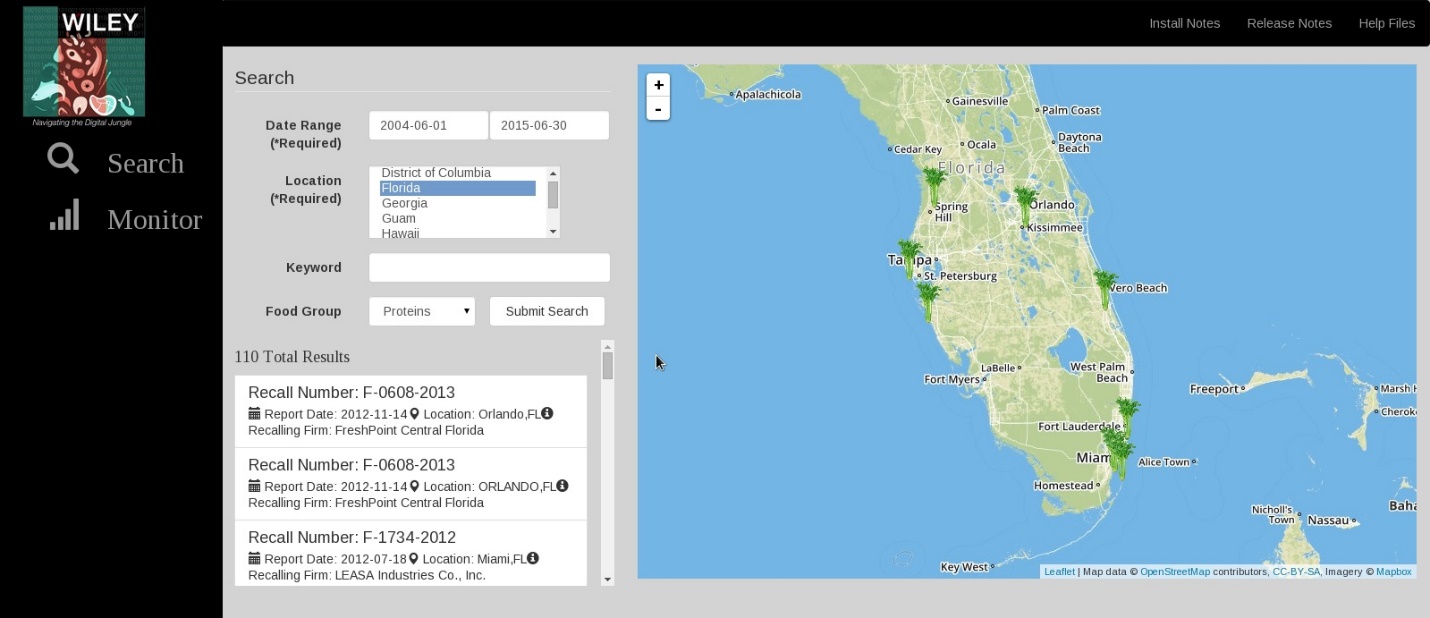


Figure 3: Search Page

The Location drop down menu allows a user to select more than one state location. The Date Range results in the end user seeing a pop-up calendar window to select the start and end date.

For Advanced Search, the user can type a keyword in the keyword search box. The search returns information found in any of the openFDA metadata fields (User ID, product description, Classification, etc.). The final end user selected search field is the Food Group drop down window which is an added feature for search over the FDA database. The end user selection of the Food Group returns results that are aligned to one of five (5) primary Food Groups – Fruits, Vegetables, Grains, Proteins, and Dairy. While not explicitly called out in the openFDA data, the Wiley data services uses semantic web technology to infer the food group based upon the description in the openFDA data.

Once the end user has selected the search criteria and selects the “Submit Search button”, the application displays the total number of records and a list of query results are displayed as index cards in order of the Recall ID with select metadata shown to the end user - product name, Recall initiation date and food description. When the user hovers over the index card, a full display of associated metadata is displayed. Finally, the location mentions in the search results are shown in a world map, indicated by a celery icon.

**Monitor Page**

The Monitor Page, shown in Figure 4, provides continuous monitoring of queries to return a collection of profiles that are the result of monitoring other end users. Specifically, the Total queries since June 2015 and the corresponding Query time(ms), Year-to-date queries, and Average queries per day.

A line graph displays a list of queries over time. Finally, a tabular view shows the Top Ten Queries which includes the Query number product Name and User ID.

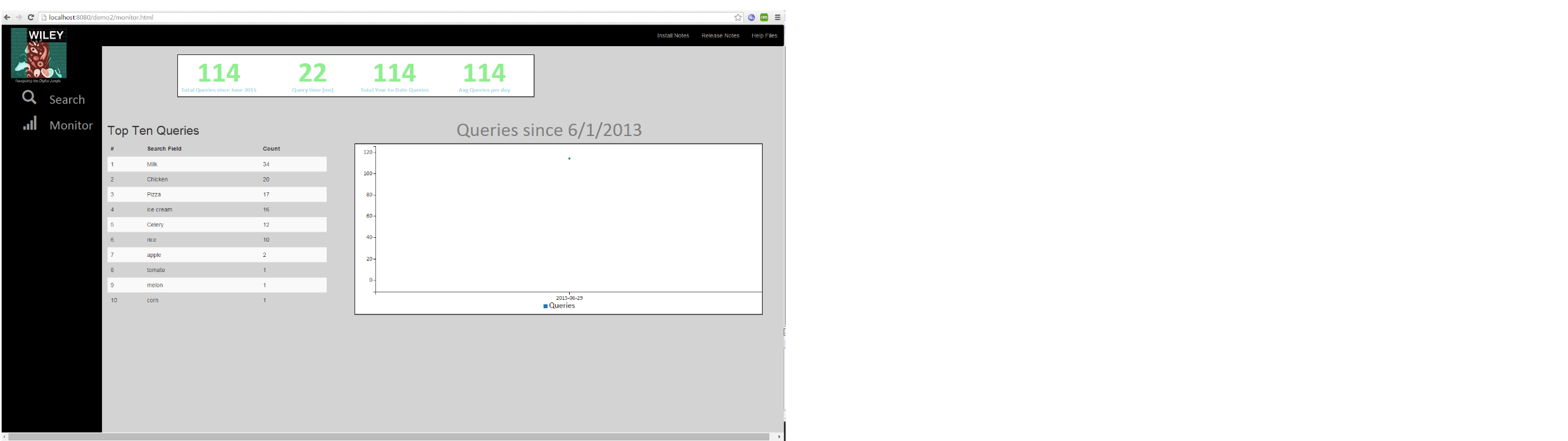


Figure 4: Monitor Page

# Data Services

Figure 3 highlights the numerous Data Services created for the GSA 18F project. The services report on System Availability, support Query Monitoring of the System and provide the overall capability to Write, Parse, and perform Entity Enrichment. All of the data services used the industry standard REST[[1]](#footnote-1) representation for the API format.

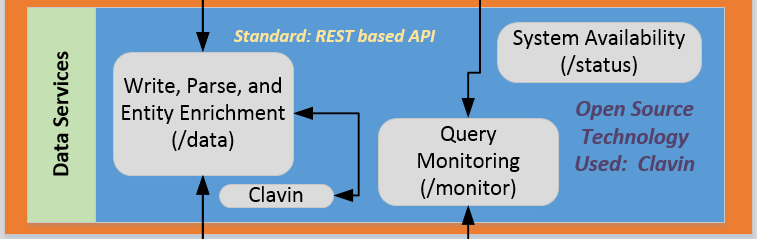


Figure 5: Wiley Data Services

Table 1 provides a complete list of the Wiley data services. Detailed descriptions of the data services are shown in Appendix A.

Table : Wiley Data Services

|  |  |
| --- | --- |
| **Service Name** | **Service Description** |
| /Data/writeToFile | Pulls openFDA from the Food API and writes to a flat file for processing. |
| /data/parsingJsonDataValue | Pulls from openFDA and stores into the Sesame Database for analytics to process the openFDA data |
| /data/update/coordinate | Semantically enriches the openFDA named location data with a translation services that provides the latitude and longitude from Clavin[[2]](#footnote-2) for each named location in the openFDA data. |
| /data/update/foodgroups | Semantically enriches the enforcement report with food group category based on the product description. |
| /data/searchQuery | Provides the search parameters based upon end user selection. |
| /status | Monitors the health and overall system availability. |
| /monitor/init | Initializes the monitoring database |
| /monitor/query | monitors user queries |
| /monitor/queryCount | total number of queries made to Wiley |
| /monitor/summary | providing the queries per day and average query time |
| /monitor/topten | a list of the top ten keywords that were searched for, and the associated count |
| /monitor/querytimes | Provides the count of queries listed by date |

The Wiley application is pulling data from three databases, represented in Figure 6. The databases are pulled from open source software (e.g. Sesame) or as part of accessing the open source site openFDA. The databases are accessed via industry standard queries.

# Databases

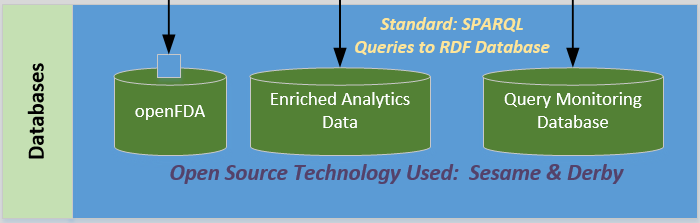


Figure 6: Wiley Databases:

The Wiley application required several key technical decisions regarding the underlying database and the technology selected to build the application and infrastructure. Three primary factors led to the selection of the components. First, as a developer of Big Data applications, the data from <https://open.fda.gov/> was analyzed to determine the size of the corpus. The number of records drives the selection of the underlying technology. With slightly more than eight (8) thousand records, the decision was made to select an RDF based database NOT a NoSQL database. This was because the number of records is not large enough to justify the overhead and processing requirements of a NoSQL infrastructure. Orbis clients processing tens of millions of records consider a NoSQL infrastructure for applications and analytics. Due to the small number, a RDF databases appeared more appropriate.

# Technical Trade Studies & Key Industry Standards

Once the database type was selected, the particular database is driven by the number of RDF statements that are derived from the source data. After analyze the data, an approximate 150 thousand statements were identified. As a result of this small size, the team selected the open source Sesame as the underlying database.

Second, the team needed to select the Web Service container and had to decide between JBoss and Tomcat. The requirement to deploy the application in an Infrastructure as a Service (IaaS) environment, the team selected Amazon Web Services (AWS). As an official trusted Industry Technology Partner, Orbis has a significant amount of experience deploying to AWS environments. As a result, the team decided to leverage numerous existing internal tools for rapid deployment and as a result, Tomcat was chosen.

Finally, the GSA 18F project has a requirement for a minimum of five (5) open source components. Incorporating open source software is a core Orbis architecture design principle so the design team was not concerned. However, the number of open source components had to be tracked to assure compliance. Table 2 highlights the notable open source components used to build the Wiley application.

Table : Wiley Open Source Software

|  |  |
| --- | --- |
| Open Source Technology | Purpose |
| JQuery | User Interface development |
| Sesame | Data Store for openFDA data |
| Jersey | Web services development |
| Leaflet | Creates map of lat/long values |
| Clavin | Extracts lat/long values from (city, state) metadata |
| Tomcat | Web server |
| Derby | Data Store for query monitoring data |
| D3 | Graph visualization |
| C3 | Graph visualization |

**Standards**

The Wiley solution incorporates key World Wide Web Consortium (W3C) IT industry standards to promote interoperability among applications and data. These standards – Resource Description Framework (RDF)[[3]](#footnote-3), Web Ontology Language (OWL)[[4]](#footnote-4), Rules Interchange Format (RIF), SPARQL, and others – provide the specifications for a schema-agnostic approach to rapidly integrating and transforming data. Orbis leverages many of the standards to promote data service and application interoperability (e.g. RESTful services) as well as standards to promote data interoperability (e.g. OWL and RDF).

**Resource Description Framework Facts**

* A schema-less format designed by W3C for representing relationships across objects in the Web
* Handles Web scale data volumes
* Subject-object-predicate model for structured & unstructured data
* Captures implicit & explicit relationships - key to integrating Customer data sources
* Can incorporate logic to aid end user decisions

# Glossary

**Metadata** - a set of data that describes and gives information about other data.

**Natural language processing** - field of computer science, artificial intelligence, and linguistics concerned with the interactions between computers and human (natural) languages.

**Web-based applications** - An application in which all or some parts of the software are downloaded from the Web each time it is run. It may refer to browser-based apps that run within the user's Web browser, or to "rich client" desktop apps that do not use a browser or to mobile apps that access the Web for additional information.

**Web-service** - a method of communication between two electronic devices over a network. It is a software function provided at a network address over the web with the service always on as in the concept of utility computing.

# Appendix A

Detailed description of data services.

|  |  |
| --- | --- |
| **URL** | **monitor/init** |
| **Method** | ***GET*** |
| **Query Params** | n/a | *n/a* |
| **Mime Type** | n/a | *n/a* |
| **Response** | **Example:** **Code:** 200  **Content:** init complete. |
| **Sample Call** | *<Just a sample call to your endpoint in a runnable format ($.ajax call or a curl request) - this makes life easier and more predictable.>* |
| **Notes** | *This is to be run as the last step of install to begin monitoring the system.* |

|  |  |
| --- | --- |
| **URL** | **monitor/query** |
| **Method** | ***Post*** |
| **Query Params** | n/a | *n/a* |
| **Mime Type** | application/json | *n/a* |
| **Response** | **Example:** **Code:** 200 |
| **Sample Call** | $.ajax({  type: 'POST',  data: JSON.stringify({"searchField":searchword,"responseTime":responseTime, "location":locationtarget}),  contentType: "application/json",  dataType: "json",  url: 'http://localhost:8080/dataIngestion/rest/monitor/query',  }) |
| **Notes** | *This records a query as having been completed*. |

|  |  |
| --- | --- |
| **URL** | **monitor/querycount** |
| **Method** | ***GET*** |
| **Query Params** | n/a | *n/a* |
| **Mime Type** | text/plain |
| **Returns** | **Example:350**  A count of the total number of queries completed. |
| **Success Response** | **Example:** **Code:** 200  **Content:** 350 |
| **Sample Call** | *<Just a sample call to your endpoint in a runnable format ($.ajax call or a curl request) - this makes life easier and more predictable.>*  var query\_stats = (function(){  var query\_stats = null;  $.ajax({  'async': false,  url: 'http://localhost:8080/dataIngestion/rest/monitor/queryCount' }  ).done(function(data) {  query\_stats = data;  }); |
| **Notes** | *<This is where all uncertainties, commentary, discussion etc. can go. I recommend timestamping and identifying oneself when leaving comments here.>* |

|  |  |
| --- | --- |
| **URL** | **monitor/summary** |
| **Method** | ***GET*** |
| **Query Params** | n/a | *n/a* |
| **Mime Type** | application/json |
| **Returns** | A Summary object with all of the summary data that is being monitored. |
| **Success Response** | **Example:** **Code:** 200  **Content:** {"QueriesSince": 15, "A\_V\_G\_QueryTime": 574292, "Y\_T\_D\_Queries": 150, "QueriesPerDay": 37, "YearlyChangeInQueries": null } |
| **Sample Call** | *<Just a sample call to your endpoint in a runnable format ($.ajax call or a curl request) - this makes life easier and more predictable.>*  $.ajax({  'async': false,  url: 'http://localhost:8080/dataIngestion/rest/monitor/summary',  }); |
| **Notes** | *<This is where all uncertainties, commentary, discussion etc. can go. I recommend timestamping and identifying oneself when leaving comments here.>* |
|  |  |

|  |  |
| --- | --- |
| **URL** | *Monitor/topten* |
| **Method** | *GET* |
| **Query Params** | n/a | *n/a* |
| **Mime Type** | application/json |
| **Returns** | A list consisting of the top queries by keyword. |
| **Success Response** | **Example:** **Code:** 200  **Content:** [{"Date":null,"Location":null,"SearchField":"Milk","Count":34,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"","Count":23,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"Chicken","Count":20,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"Pizza","Count":17,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"ice cream","Count":16,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"rice","Count":12,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"Celery","Count":12,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"orange","Count":4,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"dessert","Count":2,"ResponseTime":null},{"Date":null,"Location":null,"SearchField":"corn","Count":2,"ResponseTime":null}] |
| **Sample Call** | *<Just a sample call to your endpoint in a runnable format ($.ajax call or a curl request) - this makes life easier and more predictable.>*  $.ajax({  'async': false,  url: 'http://localhost:8080/dataIngestion/rest/monitor/topten'  }) |
| **Notes** | *<This is where all uncertainties, commentary, discussion etc. can go. I recommend timestamping and identifying oneself when leaving comments here.>* |

|  |  |
| --- | --- |
| **URL** | *monitor/querytimes* |
| **Method** | *GET* |
| **Query Params** | n/a |
| **Mime Type** | application/json |
| **Returns** | A list the count of queries done by date |
| **Success Response** | **Example:** **Code:** 200  **Content:** [{"Date":"2015-06-30","Location":null,"SearchField":null,"Count":120,"ResponseTime":null},{"Date":"2015-07-01","Location":null,"SearchField":null,"Count":15,"ResponseTime":null},{"Date":"2015-07-02","Location":null,"SearchField":null,"Count":10,"ResponseTime":null},{"Date":"2015-07-04","Location":null,"SearchField":null,"Count":5,"ResponseTime":null}] |
| **Sample Call** | *<Just a sample call to your endpoint in a runnable format ($.ajax call or a curl request) - this makes life easier and more predictable.>*  $.ajax({  'async': false,  url: 'http://localhost:8080/dataIngestion/rest/monitor/querytimes',  }) |
| **Notes** | *<This is where all uncertainties, commentary, discussion etc. can go. I recommend timestamping and identifying oneself when leaving comments here.>* |

|  |  |
| --- | --- |
| **URL** | *data/searchQuery* |
| **Method** | *GET* |
| **Query Params** | bngDateRng  endDateRng  loc  keywordSearch  foodGroup | *n/a* |
| **Mime Type** | application/json |
| **Returns** | A list of all food recalls matching the query. |
| **Success Response** | **Example:**  **Content:** {  "head" : {  "vars" : [  "recallNumber",  "reportDate",  "eventId",  "recallingFirm",  "status",  "location",  "latitude",  "longitude",  "foodGroup",  "classification",  "recallInitiationDate",  "productDescription",  "productQty",  "codeInfo",  "distPattern",  "recallReason",  "voluntaryMandated",  "notification"  ]  },  "results" : {  "bindings" : [  {  "recallNumber" : {  "type" : "literal",  "value" : "F-2317-2012"  },  "productQty" : {  "type" : "literal",  "value" : "xx"  },  "location" : {  "type" : "literal",  "value" : "Fremont,CA"  },  "status" : {  "type" : "literal",  "value" : "Ongoing"  },  "productDescription" : {  "type" : "literal",  "value" : "Licorice Root, WHL;\nDistributed by San Francisco Herb and Natural Food Co.\nFremont, Ca."  },  "foodGroup" : {  "type" : "literal",  "value" : ""  },  "codeInfo" : {  "type" : "literal",  "value" : "Lot 120019."  },  "recallingFirm" : {  "type" : "literal",  "value" : "San Francisco Herb & Natural Food Company"  },  "distPattern" : {  "type" : "literal",  "value" : "Nationwide and Canada."  },  "voluntaryMandated" : {  "type" : "literal",  "value" : "Voluntary: Firm Initiated"  },  "eventId" : {  "type" : "literal",  "value" : "62737"  },  "recallInitiationDate" : {  "datatype" : "http://www.w3.org/2001/XMLSchema#dateTime",  "type" : "literal",  "value" : "2012-07-26T00:00:00.000Z"  },  "classification" : {  "type" : "literal",  "value" : "Class II"  },  "notification" : {  "type" : "literal",  "value" : "Press Release"  },  "recallReason" : {  "type" : "literal",  "value" : "CA Department of Public Health inspectors found a pervasive infestation of mice in the facility."  },  "longitude" : {  "datatype" : "http://www.w3.org/2001/XMLSchema#double",  "type" : "literal",  "value" : "0.0"  },  "reportDate" : {  "datatype" : "http://www.w3.org/2001/XMLSchema#dateTime",  "type" : "literal",  "value" : "2012-09-26T00:00:00.000Z"  },  "latitude" : {  "datatype" : "http://www.w3.org/2001/XMLSchema#double",  "type" : "literal",  "value" : "0.0"  }  }  ]  }  } |
| **Sample Call** | *<Just a sample call to your endpoint in a runnable format ($.ajax call or a curl request) - this makes life easier and more predictable.>*  TODO |
| **Notes** | *<This is where all uncertainties, commentary, discussion etc. can go. I recommend timestamping and identifying oneself when leaving comments here.>* |

|  |  |
| --- | --- |
| **URL** | *data/writeToFile* |
| **Method** | *GET* |
| **Query Params** | n/a |
| **Mime Type** | text/html |
| **Returns** | A success message. |
| **Success Response** | **Example:** **Code:** 200  **Content:** {"sucess": true} |
| **Sample Call** | *<Visiting the url in a web browser is sufficient.>* |
| **Notes** | *<This is where all uncertainties, commentary, discussion etc. can go. I recommend timestamping and identifying oneself when leaving comments here.>* |

|  |  |
| --- | --- |
| **URL** | *data/* *parsingJsonDataValue* |
| **Method** | *GET* |
| **Query Params** | n/a |
| **Mime Type** | text/html |
| **Returns** | A success message. |
| **Success Response** | **Example:** **Code:** 200  **Content:** {"sucess": true} |
| **Sample Call** | *<Visiting the url in a web browser is sufficient.>* |
| **Notes** | *<This is where all uncertainties, commentary, discussion etc. can go. I recommend timestamping and identifying oneself when leaving comments here.>* |

|  |  |
| --- | --- |
| **URL** | *data/update/coordinates* |
| **Method** | *GET* |
| **Query Params** | n/a |
| **Mime Type** | text/html |
| **Returns** | A success message. |
| **Success Response** | **Example:** **Code:** 200  **Content:** {"sucess": true} |
| **Sample Call** | *<Visiting the url in a web browser is sufficient.>* |
| **Notes** | *Obtains the locations available in the Sesame store and looks up their coordinates in Clavin. Once coordinates have been obtained, they are added to Sesame.* |

|  |  |
| --- | --- |
| **URL** | *data/update/foodgroups* |
| **Method** | *GET* |
| **Query Params** | n/a |
| **Mime Type** | text/html |
| **Returns** | A success message. |
| **Success Response** | **Example:** **Code:** 200  **Content:** {"sucess": true} |
| **Sample Call** | *<Visiting the url in a web browser is sufficient.>* |
| **Notes** | *Loads the product description of each report loaded into Sesame and add a relationship with appropriate Food Group based on string matching with the food groups list.* |

1. <https://en.wikipedia.org/wiki/Representational_state_transfer> [↑](#footnote-ref-1)
2. https://github.com/Berico-Technologies/CLAVIN [↑](#footnote-ref-2)
3. <http://www.w3.org/RDF/> [↑](#footnote-ref-3)
4. <http://www.w3.org/2004/OWL/> [↑](#footnote-ref-4)