Software Design Document

for

Referral / ERRA Trending Analysis Tool (R/E TAT)

Version 1.0 approved

Prepared by Wilson Thomas
Emanoel
Henry Liwag
Daniel Limas
Xiaoye Li
Philip Tran

QTC Management / Grace Pang-Truong

Preliminary version : December 8, 2017
Table of Contents........................................................................................................<pg #>
Revision History........................................................................................................<pg #>

Introduction..............................................................................................................<pg #>
  1.1. Purpose.............................................................................................................<pg #>
  1.2. Document Conventions.....................................................................................<pg #>
  1.3. Intended Audience and Reading Suggestions..................................................<pg #>
  1.4. System Overview..............................................................................................<pg #>

Design Considerations...............................................................................................<pg #>
  2.1. Assumptions and dependencies...........................................................................<pg #>
  2.2. General Constraints............................................................................................<pg #>
  2.3. Goals and Guidelines..........................................................................................<pg #>
  2.4. Development Methods.......................................................................................<pg #>

3. Architectural Strategies..........................................................................................<pg #>

4. System Architecture...............................................................................................<pg #>
  4.1. ................................................................................................................................<pg #>
  4.2. ................................................................................................................................<pg #>

5. Policies and Tactics.................................................................................................<pg #>
  5.1. Specific Products Used.........................................................................................<pg #>
  5.2. Requirements traceability....................................................................................<pg #>
  5.3. Testing the software...........................................................................................<pg #>
  5.4. Engineering trade-offs.........................................................................................<pg #>
  5.5. Guidelines and conventions................................................................................<pg #>
  5.6. Protocols................................................................................................................<pg #>
  5.7. Maintaining the software.....................................................................................<pg #>
  5.8. Interfaces...............................................................................................................<pg #>
  5.9. System’s deliverables...........................................................................................<pg #>
  5.10. Abstraction...........................................................................................................<pg #>

6. Detailed System Design...........................................................................................<pg #>
  6.x Name of Module..................................................................................................<pg #>
  6.x.1 Responsibilities.................................................................................................<pg #>
  6.x.2 Constraints........................................................................................................<pg #>
  6.x.3 Composition........................................................................................................<pg #>
  6.x.4 Uses/Interactions...............................................................................................<pg #>
  6.x.5 Resources...........................................................................................................<pg #>
6.x.6 Interface/Exports.................................................................<pg #>

7. Detailed Lower level Component Design
7.x Name of Class or File.................................................................<pg #>
    7.x.1 Classification.......................................................................<pg #>
    7.x.2 Processing Narrative(PSPEC)..............................................<pg #>
    7.x.3 Interface Description............................................................<pg #>
    7.x.4 Processing Detail....................................................................<pg #>
        7.x.4.1 Design Class Hierarchy...............................................<pg #>
        7.x.4.2 Restrictions/Limitations...............................................<pg #>
        7.x.4.3 Performance Issues....................................................<pg #>
        7.x.4.4 Design Constraints....................................................<pg #>
        7.x.4.5 Processing Detail For Each Operation.....................<pg #>

8. User Interface
8.1. Overview of User Interface......................................................<pg #>
8.2. Screen Frameworks or Images..............................................<pg #>
8.3. User Interface Flow Model......................................................<pg #>

9. Database Design

10. Requirements Validation and Verification....................................<pg #>

11. Glossary......................................................................................<pg #>

12. References....................................................................................<pg #>
## Revision History

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Reason For Changes</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wilson Thomas</td>
<td>12/8/17</td>
<td>Preliminary version</td>
<td>1.0</td>
</tr>
</tbody>
</table>
1. Introduction

1.1 Purpose
This document will outline in detail the Referral / ERRA Trending Analysis Tool (R/E TAT)’s software architecture and design. This document will display the system’s design from several viewpoint to provide a guide into how the system works and to communicate what the system does. It intends to get an insight into the architectural and design decisions that were made for the R/E TAT.

1.2 Document Conventions
This document follows APA-like format, with bold-faced text used to emphasize the section headings and 12-sized Times New Roman font. Bullet points have been used to quickly identify sub-sections and organize the document to be read easily. This document attempts to outline the requirements of the application clearly and accurately. It also outlines the expected behavior of the application.

1.3 Intended Audience and Reading Suggestions
This document is written on a technical level to address the QTC Management team and Cal State LA computer science department.

1.4 System Overview
This document provides the architecture and design of the RE TAT. Given several sets of data, RE TAT will create a trending analysis tool that will identify areas that are being underused and overused, validate appointment assigned to QTC, and determine if an area has enough providers within a given distance. This software will allow the QTC management team to effectively balance how the workload will be allocated across the provider areas.
2. Design Considerations

2.1 Assumptions and Dependencies

- Possible changes in functionality after delivery date
- Possible changes in the way users input data
- Software depends on external files by user
- Software depends on user input
- Software depends on a external database
- Software depends on external files formatting

2.2 General Constraints

- User's affiliated with QTC will have access to RETAT tool
- Admins have access to create users to use the RETAT tool
- 2GB of input from external files
- User name and password required to access tool for regular users.
- Software can only be view through an internet connection

2.3 Goals and Guidelines

- Let users know that claimants assign to QTC matches with the referral data.
- Let users know that claimants not assign to QTC are within the referral data
- Let users know that claimants that are supposed to be assign to QTC are not within the referral data
- The software should be able to list the specialties that are and are not covered around several miles around a given location
- This software should be designed with future teams in mind.
  - Future teams should easily understanding the source code and data flow of the software
  - Allow future teams to easily add new features
2.4 Development Methods

This project initially used the Waterfall Development method in order to measure the progress of the deliverable. Due to limited amount of data, constant changes to the requirement, and lack of interaction with the liaison during the beginning of the project, we switched to agile development method.
3. Architectural Strategies

- In the development of RETAT, development tools will include the following:
  - Java Oracle
  - Eclipse IDE
  - Tomcat Apache 7
  - MySQL
  - phpMyAdmin
  - Bootstrap

- In RETAT, the combination of MySQL and phpMyAdmin will be used to hold the Referrals database, the ERRA database, and the VA Provider database. It would also be able to read and update the records inside the databases.

- RETAT will be using the MVC (Model-View-Controller) architecture in order to simplify system implementation. The controller module of MVC will be the central dispatcher for different modules throughout the system.

- CSS libraries, such as Bootstrap, etc. will create the user interface, as well as to modularize each section of the JSP files.

- Users have the ability to create user or admin accounts and login to access the website and the databases.

- Input verification technologies implemented using form validation techniques will be used in order to verify the integrity data being input by the user. This is important especially during the cases when the user must input data from a form. Users could inadvertently execute commands through forms without some way of cleaning (or deleting) the input.

- Technicians or members of the team with appropriate access privileges will be able to query certain parts of a centralized read-only database, namely on the phpMyAdmin database. Regular users who will log in to the server are to be authenticated on the log-in system and be registered as a normal user. They will not have any privilege of handling any data except when submitting a ticket to be approved.
- For concurrency and synchronization, multiple users are allowed to access the website and the databases simultaneously. RETAT would be able support as much sessions as the server allows it.

- The web application will use the HTTPS protocol to ensure that sensitive data remains secure when the output is being sent to the user.
4. System Architecture

In Figure 4.0, displays the data flow diagram of level 0. Before using the tool the user must interact with the tool. The arrow indicates input to the system of modules. Each element of the list is a module. When user enters an input, the modules interact with the input data. The last item of the list is the database that the tool will be connecting to.

Figure 4.1
4.1.1 User Interface Module

The User Interface Module serves as a web page. This module will display everything to the user. As well as getting the input from the user. The module changes the way the webpage looks from the interaction of the user. The User Interface Module gets the input from the user and depending on the user’s interaction it will be sent to the Main Controller Module. And receive the output from the Main Controller Module.

4.1.2 Main Controller Module

The Main Controller Module receives input from the UI and depending on the user the main controller will give back a result. The Controller module goes to the Data Parser input if the Main controller Module has received excel files. The main controller serves as a servlet/controller of the view.

4.1.3 Data Input Parser Module.

The Data Input Parser Module receives input files and parses the data from the file. And creates models for the parsed data. Which then the module will give back the data to the Main Controller module. The input files are received from the Main controller Module. The higher-level components work together as whole and follow the MVC design pattern. M stands for models. Which when the data is parsed it creates model’s so the main controller module can manipulate the data. V stands for View and the User Interface Module is that part of the system’s design pattern. View is will in charge of display and having a way to send input to the Main Controller Module. C stands for controller and the Main Controller Module is the controller for the view. The controller is what controls the system. It sends and receives data.

4.1.3 User Authentication Module (mark)
6. Detailed System Design (DFD Level 2)

6.1 User Interface Module

6.1.1 Responsibilities

Requirement check feature allows QTC to check whether there are areas that are missing providers. The filter can be used to filter based on state and also find specific Zip Code. There is also an option to filter whether the result is Zip Code which are meeting the requirement or not. Crosscheck feature helps QTC to check between referral list and ERRA data to see what part of the data matches and what doesn't match.

6.1.3 Composition

Filters in our features is a meaningful subcomponents that helps QTC narrow down their results from their data.

6.1.4 Uses/Interactions

The user will use our interactive website by using the hierarchy options we implemented that manipulates the data we were given by QTC.

6.1.5 Resources

The main resource we have is QTC excel files. Due to security and privacy concerns, our data is limited but it gives us the necessary data needed to execute the desired goals of helping QTC identify areas where they are missing providers and help them know when and where are referrals were assigned.

6.2 Main Controller Module

6.2.1 Responsibilities

This component is one of the cores of the system. The module handles three sub modules depending on the user’s interaction. It sends useful information to the client. This module is the controller of the User Interface Module. This module will control the input given by the user and send output to the user.

6.2.2 Constraints
An assumption is that passing 2GB of data from a file will take longer for the module to send back the data. Constraint that exists is the format of the data that will be passing through the module. For example, in the input should be a string with no spaces.

6.2.3 Composition

The module will help the client have a list of network providers that are being underused and overused. It will also help the client to see if they receive any referrals when they were going to. The module will also help the client identify if they have a provider within a given mile radius.

6.2.4 Uses/Interactions

The Main Controller Module receives input from the User Interface Module and sends it back to the User Module. If the MCM receives an input file from the UIM then MCM will send the file to the Data Input Parser Module. Then, MCM receives output from DIPM. Inside MCM there are four submodules. Each module receives input depending on the UIM. Only three submodules will receive output from the DIPM.

6.2.5 Resources

- JDBC library

6.3 Data Input Parser Module

6.3.1 Responsibilities

The Data Input Parser Module reads in QTC’s excel files and outputs data models the RETAT can understand and operate on. This module serves as the main communication interpreter between the QTC’s excel data format and the RETAT’s data model format.

6.3.2 Constraints

- QTC’s excel file formats are a constraint on the Data Input Parser Module. Currently, there are three formats this system can read and parser. If QTC changes the formatting of their excel files, the module would not be able to interpret it.
- The Data Input Parser Module might take about 5-10 second to read and parse the excel files. Some of the files has more than 200,000 data entries. Issues about memory storage and wait times can arise when the number of data entries increases.

6.3.3 Composition
There are 3 subcomponents in the Data Input Parser Module: QTC’s ERRA excel file formatter, QTC’s referral list excel file formatter, and QTC’s VA provider network list excel file formatter.

- **ERRA excel file formatter**
  - This subcomponents translate the QTC’s ERRA excel file data into a dataset the RE TAT can interpret.

- **Referral list excel file formatter**
  - This subcomponents translate the QTC’s referral list excel file data into a dataset the RE TAT can interpret.

- **VA provider network list excel file formatter**
  - This subcomponents translate the QTC’s VA provider network list excel file data into a dataset the RE TAT can interpret.

### 6.3.4 Uses/Interactions

- The Data Input Parser Module only communicate with the Main Controller Module in the RE TAT. The DIPM received the excel files from the MCM and sent the data model back to the MCM.
- There is a wait time between when the user sends the excel file and when the RE-TAT’s UI displays the results. The wait time comes from the reading and parsing of the excel files in the DIPM.

### 6.3.5 Resources

Apache POI is an external java library resource used in this software to read the excel files. Depending on the size of the excel file, Apache POI might use as much or little of the machine system’s memory.

### 6.4 User Authentication Module *(mark)*

#### 6.4.1 Responsibilities

The primary responsibilities and/or behavior of this component. What does this component accomplish? What roles does it play? What kinds of services does it provide to its clients? For some components, this may need to refer back to the requirements specification.

#### 6.4.2 Constraints

Any relevant assumptions, limitations, or constraints for this component. This should include constraints on timing, storage, or component state, and might include rules for interacting with this component (encompassing preconditions, post conditions, invariants, other constraints on input or output values and local
or global values, data formats and data access, synchronization, exceptions, etc.)

6.4.3 Composition
A description of the use and meaning of the subcomponents that are a part of this component.

6.4.4 Uses/Interactions
A description of this components collaborations with other components. What other components is this entity used by? What other components does this entity use (this would include any side-effects this entity might have on other parts of the system)? This concerns the method of interaction as well as the interaction itself. Object-oriented designs should include a description of any known or anticipated subclasses, superclass’s, and metaclasses.

6.4.5 Resources
A description of any and all resources that are managed, affected, or needed by this entity. Resources are entities external to the design such as memory, processors, printers, databases, or a software library. This should include a discussion of any possible race conditions and/or deadlock situations, and how they might be resolved.
7. Database Design

The RETAT system utilizes a total of three tables:

1. **Referral**: every referral has its own zip code, state, date, specialty group, and amount. Using its zip code and state as super key to track information from VA provider table.

2. **VARO**: VARO is short for Veteran affair regional office. It consists of state, date, signed provider and city feature. Its state and date feature match to the referral table. By have this table, we can track if our company get the referral.

3. **VA provider**: VA provider table consists of zip code, state, specialty group and provider ID feature. Using it zip code and state feature as super key, we can track what kind specialty group we have in certain area.
8. User Interface

8.1 Overview of User Interface

Upon visiting the site the user will be at the home with a list of the features. The User then will be able to navigate to each specific feature page from the home page. Each feature page will have a filter which will allow the user to broaden or narrow the result down based on the user's need. The user will also be able to search for specific results they are looking for. For each of the results the user will then have the option to visit the result specific page will show more information on this result.

8.2 Screen Frameworks or Images
These can be mockups or actual screenshots of the various UI screens and popups.
Requirements Check

92778 Does not Meet Requirement

Missing Specialties

<table>
<thead>
<tr>
<th>Specialty</th>
<th>Referrals (Requested)</th>
<th>Nearest Provider ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUD</td>
<td>14</td>
<td>14/5/5 (132 miles)</td>
</tr>
<tr>
<td>DER</td>
<td>3</td>
<td>92/7/4 (138 miles)</td>
</tr>
<tr>
<td>DEN*</td>
<td>8</td>
<td>2/17/22 (22 miles)</td>
</tr>
</tbody>
</table>

Available Specialties

<table>
<thead>
<tr>
<th>Specialty</th>
<th>Provider ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>GYN</td>
<td>213</td>
</tr>
<tr>
<td></td>
<td>5005</td>
</tr>
<tr>
<td></td>
<td>1218</td>
</tr>
<tr>
<td>CAR</td>
<td>1098</td>
</tr>
<tr>
<td></td>
<td>28107</td>
</tr>
<tr>
<td>QTO</td>
<td>1373</td>
</tr>
<tr>
<td></td>
<td>18065</td>
</tr>
</tbody>
</table>

Specialty Group (from referral list) | Minimum Requirement
AUD, CAR, DEN, DER, GAS, GYN, NIS, NDU, GPH, ORT, QTO, PSY, PUL, RAD, URO | 100
GEN | 50

Crosscheck

<table>
<thead>
<tr>
<th>Date</th>
<th>UpToDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/3/2017</td>
<td>9/2303</td>
</tr>
<tr>
<td>10/3/2017</td>
<td>9/2303</td>
</tr>
</tbody>
</table>

Range

<table>
<thead>
<tr>
<th>November 2017</th>
<th>December 2017</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6 7</td>
<td>1 2 3 4 5 6 7</td>
</tr>
<tr>
<td>8 9 10 11 12 13</td>
<td>8 9 10 11 12 13</td>
</tr>
<tr>
<td>14 15 16 17 18 19</td>
<td>14 15 16 17 18 19</td>
</tr>
<tr>
<td>20 21 22 23 24 25</td>
<td>20 21 22 23 24 25</td>
</tr>
<tr>
<td>26 27 28 29 30 31</td>
<td>26 27 28 29 30 31</td>
</tr>
</tbody>
</table>

Select one of the following:

- [ ] ERAS not in referral
- [ ] Referral not to ERAS
- [ ] Both not valid

Submit
8.3 User Interface Flow Model
A discussion of screen objects and actions associated with those objects. This should include a flow diagram of the navigation between different pages.
11. Glossary

An ordered list of defined terms and concepts used throughout the document. Provide definitions for any relevant terms, acronyms, and abbreviations that are necessary to understand the SDD document. This information may be listed here or in a completely separate document. If the information is not directly listed in this section provide a note that specifies where the information can be found.

12. References

<List any other documents or Web addresses to which this SDD refers. These may include other SDD or SRS documents, user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

Brad Appleton <brad@bradapp.net>  http://www.bradapp.net
https://www.cs.purdue.edu/homes/cs307/ExampleDocs/DesignTemplate_Fall08.doc