Software Design Document for Drawdown Interface (DDI)

Version 1.0 approved

Prepared by
Pranil Dahal
Iovanni Enriquez
Wesley Hwang
Cesar Rojas
Vay Tang

LA County Department of Regional Planning

October 5th, 2018
# Table of Contents

- Revision History..............................................................................
- Introduction....................................................................................
  - Purpose.....................................................................................
  - Document Conventions............................................................
  - Intended Audience and Reading Suggestions...........................
  - System Overview........................................................................
- Design Considerations....................................................................
  - Assumptions and dependencies................................................
  - General Constraints..................................................................
  - Goals and Guidelines................................................................
  - Development Methods..............................................................
- Architectural Strategies................................................................
  - Data Extraction Strategy..........................................................
  - Angular Components...................................................................
- System Architecture.......................................................................!
  - Higher Level DFD.......................................................................!
  - Module Functional Descriptions...............................................!
- Policies and Tactics.......................................................................!
  - Specific Products Used..............................................................
  - Requirements traceability..........................................................
  - Testing the software......................................................................
  - Guidelines and conventions....................................................!
  - Hierarchical Organization..........................................................
  - System's deliverables.................................................................
- Detailed System Design................................................................
  - Backend Module........................................................................
    - Responsibilities......................................................................
    - Constraints............................................................................
    - Composition..........................................................................!
    - Uses/Interactions..................................................................
    - Resources................................................................................
    - Interface/Exports..................................................................
  - Frontend Module........................................................................
    - Responsibilities......................................................................
    - Constraints............................................................................
    - Composition..........................................................................!
    - Uses/Interactions..................................................................
    - Resources................................................................................
    - Interface/Exports..................................................................
- Detailed Lower level Component Design........................................
- Database Design............................................................................
- User Interface...............................................................................!
  - Overview of User Interface.......................................................!
  - Screen Frameworks or Images....................................................!
  - User Interface Flow Model.........................................................!
# Revision History

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Reason For Changes</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Introduction

1.1 Purpose

The purpose of this document is to document the design of the software Drawdown Interface, referred to as DDI.

This is the Version 1 of this document. It doesn’t include the final design.

This document contains designs related to the two main modules within DDI, Frontend and Backend.

1.2 Document Conventions

This document was written based on the SDD template provided on CS 4961.

No additional applicable document has been used in the production of this document.

There is no document controlling this document.

1.3 Intended Audience and Reading Suggestions

This document is intended for project managers, developers, testers, system administrators and documentation writers. Users of DDI will have no need to read this document.

Backend developers of this software are recommended to skip to Section 6 and 7 for design information.

UI and Frontend developers of this software are recommended to skip to Section 8 for UI design information.

System administrators are recommended to look at Section 5 for high level information regarding maintenance, interfaces, deliverables and guidelines of DDI.

Documentation writers will need to look at the table of contents (Page 2-3) to determine which section contains the information they seek.
1.4 System Overview

The Los Angeles County Department of Regional Planning does not have an application that can centralize all data that is related to Deposit Accounts. Because of this, ad hoc reports are constantly being created to track these accounts. DDI will allow users to run custom reports on deposit data, as well as provide a mechanism to drill into more specific data about the respective plans, case, and account.
2. Design Considerations

2.1 Assumptions and Dependencies

DDI design is created based on the assumption that the clients will have Internet Explorer browser version GREATER THAN 8.

DDI user authentication has been designed to fit La County employee LDAP. Any changes in the authentication process will affect the requirements in the future.

DDI uses mssql-jdbc 7.0.0.jre8 version, provided by com.microsoft.sqlserver. Any changes in the version of the Deposit Account database will affect the requirements.

DDI is designed based on the assumption that the targeted users are employees at LA County Department of Regional Planning, and have a clear understanding of what a plan, a deposit account and a case is, and how they relate to one another in Zoning Enforcement.

2.2 General Constraints

I. The Sql Server Express instance used by DDI must be configured to listen on TCP port 1433, along with a separate user credentials created for the local DDI server to authenticate with.

II. The user credentials must be configured with read-only permission to avoid any possible data corruptions.

III. DDI shall be developed on a private repository. Software developers shall commit new patches on the private github repository, which they shall be added to as contributors.

IV. This product is used only by the employees at LA County Department of Regional Planning. Any changes to the product must be authorized by the Principal Application Developer at LA County.

V. Development milestones shall be reported to the respective IT Manager associated with this project.
2.3 Goals and Guidelines
   I. The product should be intuitive
      A. the drill-down mechanism provided on the UI must be easy to browse with
      B. the UI must be responsive, and functions must have a set life-cycle
   II. A working version of the software must be completed by the end of Fall 2018
      A. level 1 of the drill-down functionality must be functional
      B. all 4 reports must be functional
      C. email notifications must be functions

2.4 Development Methods

DDI will be developed using “Agile Development Workflow” on Github.

The workflow goes as following:
   I. Create requirements for a specific feature on the “Issues” tab
   II. Assign the “Issue” to a developer
   III. Developer implements the feature
   IV. Developer makes a pull-request to the master branch
   V. Peer-review will be conducted by two other developers on the team
   VI. Upon passing the code review, the patch will be applied to the master branch.
   VII. The new feature will be assigned to the Test-Lead for developing test cases
3. Architectural Strategies

3.1 Data Extraction Strategy

The data extraction mechanism has been split into three major sections on the backend. Every row of SQL data that gets extracted and displayed to the user goes through the following steps.

I. Rowmapper Java Object
II. Extraction Class
III. Controller Class

First, the Extraction Class uses pre-built SQL Queries to extract rows of data from the database. After that, each row gets mapped to a Java Object that corresponds to the type of data that is being extracted from the database. For example, the NonMatchingAccount.java class is used to create instances of Java Objects that hold the information about all the accounts with non-matching account numbers.

Finally, the Controller Class provides the endpoint that will be used by the frontend, and the data gets supplied in JSON format.

3.2 Angular Components

[To Be Determined]
4. System Architecture

The DDI architecture is summarized in the Context Diagram (DFD Level 0) given below. The Context Diagram provides the overall structure of the software modules and all its inputs and outputs. The notation used corresponds to that defined for any Data Flow Diagram (DFD).

4.1 Higher Level Data Flow Diagram

The major tool used to design DDI is the Data Flow Diagram (DFD). The rationale behind the selection of DFDs as the preferred design tool, was their simplicity and versatility. In the future more sophisticated tools may be used particularly if a correlation from Design to Requirement to Implementation and Testing is found to be a necessary addition.

The DDI major functional design components are shown in the DFDs below.
DDI is a full stack web application that will allow LA County Regional Planning employees to view information related to different CUP accounts held by LA County Regional Planning Department. DDI uses its Backend (2.1) module to query the database and retrieve financial and personal data about different CUP accounts. DDI also provides various REST APIs that can be called to using HTTP requests to return specific data in JSON format.

DDI also provides a Frontend (2.2) module that has a UI which consumes the REST APIs provided by Backend (2.1) and displays the data in a web UI. Users are allowed to view 4 different types of reports for the CUP accounts in the database. DDI Frontend also allows users to click on different objects on their screen to view details about the respective object.

4.2 Module Functional Descriptions

Backend - Module 2.1

Module 2.1 is the Backend of the DDI system. Within the module are four submodules as well as the database that contains all the CUP financial and personal information. There is a Data Access Module (DAM 2.1.1) which queries the database which returns data from the LA County Regional Planning CUP database (LAD). The DAM supplies data to the HTTP Request Handler Module (HRHM 2.1.2). The Testing CUP (TCUP 2.1.3) module makes HTTP requests to the HRHM with various types of data in order to test the behavior of HRHM. The Trend Analysis Module (TAM 2.1.4) uses algorithms to determine trends and patterns in financial data and makes predictions for future.

Frontend - Module 2.2

DDI also provides a Frontend (2.2) module that has a UI which displays the user with four reports.
1. All Balance Reports.
2. Individual Balance by Contract.
5. Policies and Tactics

5.1 Choice of which specific products used

I. Eclipse
II. Java JDK 8
III. Sql Server Express 2007
IV. RestAssured
V. Angular
VI. Maven

5.2 Plans for ensuring requirements traceability

Each requirement for Backend 2.1 Module has been mapped to a “Issue” on github. Example, as Issue has been created to track the development of Non-Matching Accounts API.

These github Issues all go through development, code-review and test phase. This ensures that the requirements are being fully implemented, and the documentation of the Issues and test-cases ensure the traceability of it.

5.3 Plans for testing the software

The TCUP Module will test the validity of potential HTTP requests to endpoints from the HRHM and verify whether or not the response returned from these endpoints are valid. The response returned consists of data such as response code, response message, and response body. The TCUP will then determine whether or not this response is a valid response and return validation/verification to the HRHM.

5.4 Coding guidelines and conventions

Every developer in the team is expected to follow the follow guidelines while coding:

I. meaningful variable names (allExtractedAccounts vs tempList)
II. class names must be self-explanatory (NonMatchingAccountsExtractor.java)
5.5 Hierarchical organization

There are three main java packages used in DDI on the backend module:

I. Objects package - This package contains all the Java Classes used internally to hold the data extracted from the database.

II. DAM package - This package contains all the Java Classes with SQL Queries that perform the data extraction, and map the SQL rows to Java Objects.

III. API package - This package contains all the Java Classes that define the HTTP endpoints that will be used by the frontend.

5.6 System's deliverables

The deliverables of DDI is automatically generated by the Maven.

After running “mvn clean install”, a jar snapshot is generated inside the target folder. That will be delivered to the customer.
6. Detailed System Design

6.1 Backend Module

6.1.1 Responsibilities

Within the backend module are four submodules as well as the database that contains all the CUP financial and personal information. There is a Data Access Module (DAM 2.1.1) which queries the database which returns data from the LA County Regional Planning CUP database (LAD). The DAM supplies data to the HTTP Request Handler Module (HRHM 2.1.2). The Testing CUP (TCUP 2.1.3) module makes HTTP requests to the HRHM with various types of data in order to test the behavior of HRHM. The Trend Analysis Module (TAM 2.1.4) uses algorithms to determine trends and patterns in financial data and makes predictions for future.
6.1.2 Constraints

The database access credentials are expected to be specified in the application.properties file inside src/main/resources.

The database credentials are expected to have read-only access to the database.

6.1.3 Composition

Sub-module 2.1.1 Data Access Module – Module 2.1.1

The DAM will be the medium between the database, the TAM(implementation TBD), and the HRHM. It will create queries based on the data received from the HRHM, which will then return a result set from the database. Upon further development, the DAM will also send information about the pulled objects to the TAM.

Sub-module 2.1.2 HTTP Request Handler Module – Module 2.1.2

The HRHM will be the first module to see data from the Frontend and will connect the backend with the Frontend. It will receive HTTP requests from the front end and write out queries based on the data that is requested. The HRHM will also supply the TCUP module with either dummy data or real data to see if everything is correct. After receiving JSON data from the DAM, the data will be forwards back to the Frontend.

Sub-module 2.1.3 Testing CUP – Module 2.1.3

The TCUP will test the validity of potential HTTP requests to endpoints from the HRHM and verify whether or not the response returned from these endpoints are valid. The response returned consists of data such as response code, response message, and response body. The TCUP will then determine whether or not this response is a valid response and return validation/verification to the HRHM.

Sub-module 2.1.4 Trend Analysis - Module 2.1.4

Module 2.2 is the Frontend of the DDI System. Within the module there are four submodules to display CUP financial and personal information. There is the Authentication Module (AM 2.2.1) which authenticates users using the LDAP authenticator service provided by Los Angeles County. The (2.2.2) pushes HTTP requests to the Data Access Module (DAM) of the Backend. In turn, the DAM returns data to the Frontend. The (2.2.3) receives JSON scripts of data from the Backend. The Master Controller Module (2.2.4) controls the flow of data. This module determines which data to push to the Backend and where to send data received from the Backend.
6.1.4 Uses/Interactions
The Backend module’s only interaction with other modules is provided in the REST APIs inside HRHM 2.1.2 module.

The Frontend 2.2 module makes HTTP requests to the endpoints, and the HRHM forwards those requests accordingly and returns the requested data to the Frontend.

6.1.5 Resources
Backend module requires RestAssured library for its TCUP module. In order to develop test cases, developers will be using the RestAssured library, integrated with Maven, to develop and run the tests during build.

6.1.6 Interface/Exports
Currently, DDI Backend Module provided four main REST APIs:

I. /nonmatching/all - This endpoint doesn’t require any parameters to be passed in by the user. It returns a list of all the accounts whose account number doesn’t match the case number.

II. /balances/all - This endpoint doesn’t require any parameters to be passed in by the user. It returns a list of all the accounts in the database.

III. /balances/plan - This endpoint doesn’t require any parameters to be passed in by the user. It returns a list of all the accounts, categorized by plan.

IV. /balances/contact - This endpoint doesn’t require any parameters to be passed in by the user. It returns a list of all the accounts, categorized by contact.
6.2 Frontend Module

6.2.1 Responsibilities

The Frontend module provides a UI with four main reports functionality. Each report provides the user with balance reports. The Balance Report allows the user to search for an account balance report by account, plan, and region. The Individual Balance by Contract Report searches balance reports by Contact name or number. The Individual Balance by Plan allows the user to search for accounts by planner. Each planner is linked to multiple accounts to which users will choose the account in order to view the balance report. The Non-Matching Accounts Report will display all non-matching linked reports and the user will have the ability to send an email of the non-matching report to the person in charge of the plan.

6.2.2 Constraints

The Backend Module is expected to be running the jar file.

The Host must have the server port open for HTTP connections.
6.2.3 Composition

Sub-module 2.1.1 User Interface – Module 2.2.1
Module 2.2.1 is the main display that the user sees and interacts with. It contains rendered components with tables and data filters that allow user to view data on different levels. UI will be rendered HTML code with CSS, controlled by the components received from PPM. UI shall send parameter for different UI events (clicks, drags, etc) to the PPM. UI also received updated components from the PPM based on the parameters passed to the PPM.

Sub-module 2.2.2 Parameter Processing Module – Module 2.2.2
Module 2.2.2 is the central processing module for the Frontend. PPM receives the parameters from the UI (example: account ID, plan ID, date ranges, etc) and constructs the API query. The API query gets sent to the HRHM module (2.1.2). PPM also receives the JSON data and renders the necessary UI components and passes them on to the UI.

6.2.4 Uses/Interactions
The Frontend module heavily interacts with the endpoints provided by the Backend Module.
More details will be added as the Frontend Module gets developed.

6.2.5 Resources
More details will be added as the Frontend Module gets developed.

6.2.6 Interface/Exports
More details will be added as the Frontend Module gets developed.
7. Detailed Lower level Component Design

Not Applicable at the moment.
Document will be revised when lower level components get developed.
8. Database Design

Not Applicable to DDI
DDI uses the database already provided by LA County Department of Regional Planning.
9. User Interface

9.1 Overview of User Interface

DDI also provides a Frontend module that has a UI which displays the user with four reports.
1. All Balance Reports.
2. Individual Balance by Contract.

9.2 Screen Frameworks or Images

As the UI gets developed, more images will be added and the document will be revised accordingly.
9.3 User Interface Flow Model
Document will be updated with this information as the Frontend module gets developed.
10. Requirements Validation and Verification

The methodologies and testing strategies identified at this point include four major approaches: TESTING, DEMONSTRATION, INSPECTION, and ANALYSIS with various variations to adapt to the iCUP characteristics:

I. Testing using additional ad-hoc created software including a correlation testing unit.
II. Demonstration of the specified capability
III. Inspection of the software code possibly using additional inspection techniques
IV. Analysis of the specific code operation/algorithm to prove functionality.

The DDI Functional and Performance Requirements Validation Matrix is given below.
- The “Requirement No.” column mentions which module implements the specific requirement.
- The “Requirement Description” column mentions the description of the requirement.
- The “V&V Methodology” mentions the testing methodology used in order to test the requirement.

<table>
<thead>
<tr>
<th>Requirement No.</th>
<th>Requirement Description</th>
<th>V&amp;V Methodology</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.1-1</td>
<td>The DAM shall query the LAD</td>
<td>Testing</td>
</tr>
<tr>
<td>3.1.1-2</td>
<td>The DAM shall receive data returned by the LAD</td>
<td>Testing</td>
</tr>
<tr>
<td>3.1.1-3</td>
<td>The DAM shall accept requests for data from HRHM and TAM</td>
<td>Testing</td>
</tr>
<tr>
<td>3.1.1-4</td>
<td>The DAM shall return data to the HRHM</td>
<td>Testing</td>
</tr>
<tr>
<td>3.1.2-1</td>
<td>The HRHM shall accept HTTP Requests from Frontend</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>3.1.2-2</td>
<td>The HRHM shall request the DAM for data</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>3.1.2-3</td>
<td>The HRHM shall return JSON data to the Frontend</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>3.1.2-4</td>
<td>The HRHM shall accept HTTP Requests from the TCUP</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>3.1.2-5</td>
<td>The HRHM shall return JSON data to TCUP</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>3.1.2-6</td>
<td>The HRHM shall accept JSON</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>Requirement No.</td>
<td>Requirement Description</td>
<td>V&amp;V Methodology</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>3.1.2-7</td>
<td>The HRHM shall call “/nonmatching/all” to query for all non-matching accounts</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>3.1.2-8</td>
<td>The HRHM shall call “/balances/plan/dates” to query for individual balances by plans with date range</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>3.1.2-9</td>
<td>The HRHM shall call “/balances/contact/dates” to query for individual balances by contact with date range</td>
<td>Testing, Demonstration</td>
</tr>
<tr>
<td>3.1.2-10</td>
<td>The HRHM shall call “/balances/all” to query for all available balances</td>
<td>Testing, Demonstration</td>
</tr>
</tbody>
</table>

V&V Related to Design Module 2.2 and Sub-modules 2.2.1, 2.2.2, etc.…

<table>
<thead>
<tr>
<th>Requirement No.</th>
<th>Requirement Description</th>
<th>V&amp;V Methodology</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2.1-1</td>
<td>The UI shall render and display “IndividualBalancebyPlan” component to the User.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.1-2</td>
<td>The UI shall render and display “IndividualBalancebyContact” component to the User.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.1-3</td>
<td>The UI shall render and display “NonMatchingAccounts” component to the User.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.1-4</td>
<td>The UI shall render and display “AllAccountBalance” component to the User.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.1-5</td>
<td>The UI shall call functions inside PPM for parsed JSON data.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.1-6</td>
<td>The UI will pass parameters to the PPM.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.1-7</td>
<td>The UI will display the appropriate data received from the PPM using components.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.2-1</td>
<td>The PPM shall call the endpoint mentioned in 3.2.1-7</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.2-2</td>
<td>The PPM shall call the endpoint mentioned in 3.2.1-8</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>3.2.2-3</td>
<td>The PPM shall call the endpoint mentioned in 3.2.1-9</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.2-4</td>
<td>The PPM shall call the endpoint mentioned in 3.2.1-10</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.2-5</td>
<td>The PPM shall parse JSON data received from endpoints.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.2-6</td>
<td>The PPM shall provide parses data to UI for rendering components.</td>
<td>Demonstration, Analysis</td>
</tr>
<tr>
<td>3.2.2-7</td>
<td>The PPM shall receive data parameters from UI.</td>
<td>Demonstration, Analysis</td>
</tr>
</tbody>
</table>
11. Glossary
Plan - a case that is created to track any development that is performed on a property

Deposit Account - is an account that a customer deposits a predetermined amount of money in so that planners can perform work on a property

CUP - Conditional Use Permit
REST - Representational State Transfer
API - Application Programming Interface
HTTP - Hypertext Transfer Protocol
HRHM - HTTP Request Handler Module
TCUP - Testing CUP
TAM - Trend Analysis Module
LAD - Los Angeles Database
JSON - JavaScript Object Notation
PPM - Parameter Processing Module
UI - User Interface
DDI - Drawdown Interface
SSO - Single Sign On

12. References

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