Prevent, Mitigate, and Recover (PMR) Insight Collective Knowledge System (PICK)

Software Design Document

Version 0.5

10 March 2020

**Document Control**

**Approval**

The Guidance Team and the customer shall approve this document.

**Document Change Control**

|  |  |
| --- | --- |
| Initial Release: | 3/3/2020 |
| Current Release: | 3/10/2020 |
| Indicator of Last Page in Document: | $ |
| Date of Last Review: | 3/9/2020 |
| Date of Next Review: | 3/11/2020 |
| Target Date for Next Update: | 3/11/2020 |

**Distribution List**

This following list of people shall receive a copy of this document every time a new version of this document becomes available:

Guidance Team Members:

Dr. Gates

Dr. Salamah

Dr. Roach

ElsaTai Ramirez

Jake Lasley

Customer: Dr. Oscar Perez

Vincent Fonseca

Herandy Denisse Vazquez

Baltazar Santaella

Florencia Larsen

Erick De Nava

Software Team Members:

Eduardo Herrera – Systems Architect

Micheal Sansone – Lead Programmer

Jazmin Paz – Designer

Leslie Gomez – V&V

Jorge Flores – System Analyst

**Change Summary**

The following table details changes made between versions of this document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Modifier | Description |
| 0.1 | 02/28/2020 | Leslie Gomez | Creation of Document |
| 0.2 | 03/10/2020 | Eduardo Herrera has no soul | Creation of Section 3; the Intake Component. |
| 0.3 | 03/10/2020 | Micheal John Sansone | added to2.2 |
| 0.4 | 03/10/2020 | Jorge Flores | Made contributions to 2.2 |
| 0.5 | 03/10/2020 | Leslie Gomez | Added sections 1.1, 1.4, 1.5, 2.1 |
|  |  |  |  |

Table of Contents

[**DOCUMENT CONTROL II**](#_heading=h.30j0zll)

[Approval ii](#_heading=h.3as4poj)

[Document Change Control ii](#_heading=h.1pxezwc)

[Distribution List ii](#_heading=h.49x2ik5)

[Change Summary ii](#_heading=h.2p2csry)

[**1.**](#_heading=h.147n2zr) **INTRODUCTION 1**

[1.1.](#_heading=h.3o7alnk) Purpose and Intended Audience 1

[1.2.](#_heading=h.23ckvvd) Scope of Product 1

[*1.2.1.*](#_heading=h.ihv636) *Database 1*

[*1.2.2.*](#_heading=h.32hioqz) *Notification Manager 1*

[*1.2.3.*](#_heading=h.1hmsyys) *Unit Conversion 1*

[1.3.](#_heading=h.41mghml) References 1

[1.4.](#_heading=h.17dp8vu) Definitions, Acronyms, and Abbreviations 2

[*1.4.1.*](#_heading=h.2grqrue) *Definitions 2*

[*1.4.2.*](#_heading=h.vx1227) *Acronyms 2*

[*1.4.3.*](#_heading=h.3fwokq0) *Abbreviations 2*

[1.5.](#_heading=h.1v1yuxt) Overview 2

[**2.**](#_heading=h.4f1mdlm) **DECOMPOSITION DESCRIPTION 3**

[2.1.](#_heading=h.2u6wntf) Scope 3

[2.2.](#_heading=h.19c6y18) Use 3

[2.3.](#_heading=h.3tbugp1) Subsystem Description 3

[2.4.](#_heading=h.28h4qwu) Hierarchy Graphs 4

[*2.4.1.*](#_heading=h.nmf14n) *Database Manager Subsystem 4*

[*2.4.2.*](#_heading=h.37m2jsg) *Database Manager (API) 4*

[*2.4.3.*](#_heading=h.1mrcu09) *Unit Conversion 5*

[*2.4.4.*](#_heading=h.46r0co2) *Notification Manager 5*

[**3.**](#_heading=h.2lwamvv) **DEPENDENCY DESCRIPTION 6**

[3.1.](#_heading=h.111kx3o) Scope 6

[3.2.](#_heading=h.3l18frh) Use 6

[3.3.](#_heading=h.206ipza) Collaboration Description 6

[**4.**](#_heading=h.4k668n3) **DETAILED DESIGN 8**

[4.1.](#_heading=h.2zbgiuw) Scope 8

[4.2.](#_heading=h.1egqt2p) Use 8

[4.3.](#_heading=h.3ygebqi) Components 8

[*4.3.1.*](#_heading=h.2dlolyb) *Database Manger 8*

[*Scenario 1: Perform insert, update, select, or delete query. 8*](#_heading=h.sqyw64)

[*Scenario 1: Upload. 9*](#_heading=h.3cqmetx)

[*Scenario 2: Download. 9*](#_heading=h.1rvwp1q)

[*4.3.2.*](#_heading=h.4bvk7pj) *Unit Conversion 9*

[*Scenario 1: Request conversion of units. 9*](#_heading=h.2r0uhxc)

[*4.3.3.*](#_heading=h.1664s55) *Notification mails 10*

[*Scenario 1: Edit data. 10*](#_heading=h.3q5sasy)

[*Scenario 2: Add new weather station. 10*](#_heading=h.25b2l0r)

[4.4.](#_heading=h.qsh70q) Database Schema 11

# Introduction

## Purpose and Intended Audience

<< document purpose: system design. Describe the intent behind this document >>

The system to be developed is named the PMR Insight Collective Knowledge (PICK), and it is a tool used to perform analyses that has the capability of deriving a thorough story from log files having to do with a series of events that take place on a given network. The system is able to ingest raw log files of a multitude of formats, parse the information, and store within the system as log entries. Furthermore, the user can specify a vector to analyze, flagging significant events and introducing them as nodes of the vector. Once the vector is finalized, the system can export a visual representation of the vector, known as a graph, as well as the table representation of the vector. The purpose of the system is to facilitate and accelerate event analysis, which would otherwise take up to months to complete. The system will be dynamic in that it will allow for little user intervention, if desired, because most of the work will be performed by the system.

The purpose of creating the software design document is

## Scope of Product

The development of this tool is essential because the time and complexity needed in analyzing logs and generating reports from given data is currently a setback for the client. In creating this tool, the clients will be able to quickly analyze relevant logs and artifacts, and from the gathered data, generate detailed reports from the adversarial assessment. PMR Insight Collective Knowledge will do most of the work, retracting sizable workload from the analyst.

## References

[1] Dr. Roach et al, Prevent, Mitigate, and Recover (PMR) Insight Collective Knowledge System (PICK)

Software Requirements Specification, version 1.7, 2020.

## Definitions, Acronyms, and Abbreviations

### Definitions

|  |  |
| --- | --- |
| Contract | Set of cohesive responsibilities |
| Pre- condition | Capture the conditions that must be true in order of the method to execute correctly |
| Post- condition | Must clearly state what is true when the method completes execution. |
| Subsystem | Set of cohesive classes that collaborate among themselves to assist a set of contracts |

### Acronyms

|  |  |
| --- | --- |
| SDD | Software Design Document |
| UTEP | University of Texas at El Paso |
| PICK | Prevent Insight Collective Knowledge |
| GUI | Graphical User Interface |

### Abbreviations

|  |  |
| --- | --- |
| e.g. | For example |
| i.e. | That is |

## Overview

The Software Design Document comprises the following sections: Decomposition Description, Detailed Description of Component and Database.

The Decomposition Description provides a description of how the component descriptions can be used by designers and maintainers. It will identify major design entities, for purposes such as determining which entity is responsible for specific functions and tracing requirements to design entities.

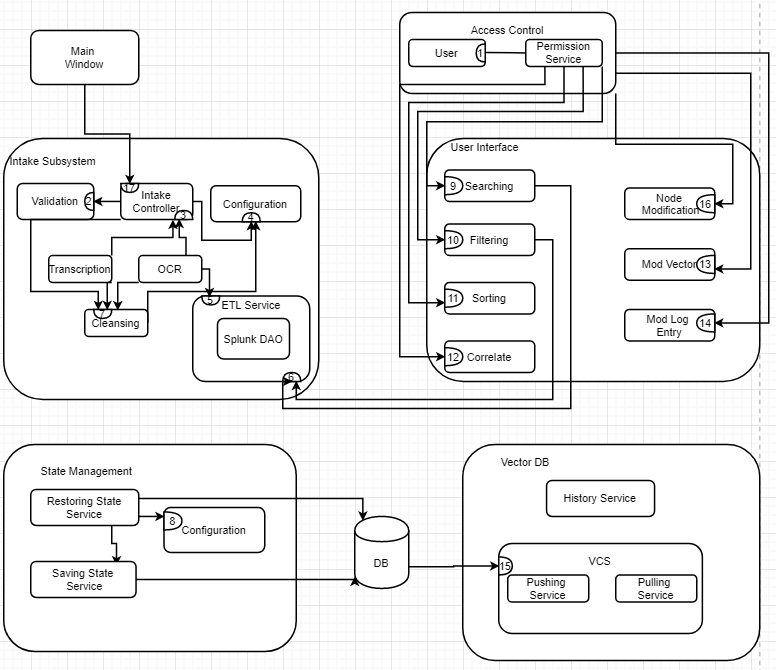
The Detailed Description of Component will provide a section with a detailed design description of each of the components listed in section 2.2, Subsystem and Component Description.

The Database section will describe the database schema and layout required by the system.

# Decomposition Description

Our system was designed using the following steps: first, we identified a set of classes, then we determined their responsibilities, and finally their collaborations. Each class description gives a description of the responsibilities that the class should have, as well as contains a list of contracts with their respective methods or functions that relate to those responsibilities. Each contract has a cohesive set of responsibilities it will implement and a list of the other contracts they will interact with to fulfill their task. Each contract is numbered and named so that it will be easier to identify and trace which contracts are interacting together. Once all the classes were properly identified, they were assembled into subsystems. These subsystems enclose groups of classes that collaborate with one another in order to support a set of contracts.

## System Collaboration Diagram



## Subsystem and Component Descriptions

**Subsystem Name:** validation

**Subsystem Purpose:** The purpose is to validate the files based on rules before ingestion.

**Subsystem Contracts:**

**Subsystem Name:** intake controller

**Subsystem Purpose:** The purpose is to decide what to do with a file based on its type.

**Subsystem Contracts: 247**

**Subsystem Name:** config

**Subsystem Purpose:** The purpose is to allow for the configuration of the event.

**Subsystem Contracts:**

**Subsystem Name:** transcribe

**Subsystem Purpose:** The purpose is to send the file for transcription.

**Subsystem Contracts: 367**

**Subsystem Name:** ocr

**Subsystem Purpose:** The purpose is to send the files to the ocr.

**Subsystem Contracts: 367**

**Subsystem Name:** cleansing

**Subsystem Purpose:** The purpose is to remove empty files or lines.

**Subsystem Contracts:**

**Subsystem Name:** etl

**Subsystem Purpose:** The purpose is to ingest files based on rules of ingestion.

**Subsystem Contracts:**

**Subsystem Name:** restoring state service

**Subsystem Purpose:** The purpose is to restore the program to its previous state.

**Subsystem Contracts: 8**

**Subsystem Name:** saving state service

**Subsystem Purpose:** The purpose is to save the current state of the program.

**Subsystem Contracts: 8**

**Subsystem Name:** configuration

**Subsystem Purpose:** The purpose is to configure ui elements.

**Subsystem Contracts:**

**Subsystem Name:** User

**Subsystem Purpose:** Provide a means for a user to utilize the system.

**Subsystem Contracts:** 13, 14, 9, 10, 11, 12, 16

**Subsystem Name:** Permission Service

**Subsystem Purpose:** To be able to provide certain privileges to a user.

**Subsystem Contracts:** 9, 13, 14, 10, 11, 12, 16

**Subsystem Name:** Searching

**Subsystem Purpose:** Be able to search by criteria

**Subsystem Contracts:** 6

**Subsystem Name:** Filtering

**Subsystem Purpose:** Be able to filter what is currently being shown to a user.

**Subsystem Contracts:** 6

**Subsystem Name:** Sorting

**Subsystem Purpose:** Be able to sort what is currently being showed to a user.

**Subsystem Contracts:** N/A

**Subsystem Name:** Correlate

**Subsystem Purpose:** Be able to provide additional features for searching to the user.

**Subsystem Contracts:** N/A

**Subsystem Name:** Modify Vector

**Subsystem Purpose:** Provide the user with capabilities to modify vector information.

**Subsystem Contracts:** N/A

**Subsystem Name:** Modify Log Entry

**Subsystem Purpose:** Provide the user with capabilities to modify a log entry.

**Subsystem Contracts:** N/A

**Subsystem Name:** Modify Node

**Subsystem Purpose:** Provide the user with capabilities to modify information on a node.

**Subsystem Contracts:** N/A

**Subsystem Name:** VCS

**Subsystem Purpose:** Provide user with pushing and pulling capabilities with vectors.

**Subsystem Contracts:** N/A

**Subsystem Name:** History Service

**Subsystem Purpose:** To provide a means of being able to see a log of what has been changed.

**Subsystem Contracts:** N/A

## Dependencies

<< describe how the component dependencies will impact development >>

# Detailed Description of Intake Component

## Component Description

Intake various log files and convert them into log entries; the log entries are then going to be validated. If there are Image files the OCR Service is going to convert the files into text, then using the ETL Service its going to convert the text into Log Entries. Similarly, If there is an audio Log File encountered, the Transcription is going to convert the audio file into text, then ETL Service is going to convert the text into Log Entries. Finally, the Intake Component is responsible for the validation of the Log Entries; for example, when a Log Entry is outside of the date range (set at the beginning of the workflow) it's going to be declared violated.

## Class Description - Intake Controller

The intake controller is going to be the medium between the different intake services and the presentation of the intake. From a path to a directory it's going to grab all the log files and convert them to log entries with the use of the ETL Service; similarly, the process is applied to a single file, if a path to a file is given. However, before converting the Log File into a Log Entry, the Intake Controller is going to cleanse the file using the Cleansing Service. Then it's going to continue with the conversion of Log Files into Log Entries. Furthermore, It's going to call the Transcription or the OCR Services if the log file is an unreadable format; which extracts the text and returns the log entry, again with the use of the ETL Service. Finally, it allows an external class to the component to subscribe an Observer to the Intake to receive the log entries that failed the Validation Service.

|  |  |
| --- | --- |
| Class Name: Intake Controller | |
| Superclass: N/A | |
| Subclasses: N/A | |
| Private Responsibilities· Gran all the log files and convert them to log entry | |
| Contract: Grab Configuration | |
| Responsibilities | Collaborations |
| · Grabbing the relative path of where the log files are located at. | Configuration |
| Contract: Validation | |
| Responsibilities | Collaborations |
| · Validating the Log Entries based on the time range that is given in the Configuration file. | Validation Service |
| Contract: Cleansing of Log Entries | |
| Responsibilities | Collaborations |
| · Using the Cleansing Service it's going to cleanse the Log Files that are being ingested | Cleansing Service |

## Class Description - Transcription Service

The Transcription Service is going to grab the audio files that are being ingested and extract the text from those files. Once the text is extracted from the audio files, the Transcription Service is going to make use of the ETL Service to convert that into Log Entries.

|  |  |
| --- | --- |
| Class Name: Transcription Service | |
| Superclass: N/A | |
| Subclasses: N/A | |
| Private Responsibilities· The transformation of audio files into Log Entries.· Through the use of the Cleansing Service, cleanse the text that is extracted from the Log Files. | |
| Contract: Extraction of Audio Files | |
| Responsibilities | Collaborations |
| · During the ingestion file, this service is going to grab the audio files that can be converted to Log Entries with this class. | Transcription Service |
| Contract: 2. Cleansing Text | |
| Responsibilities | Collaborations |
| · Cleansing the text that is being extracted from the Log Files, using the Cleansing Service. | Cleansing Service |
| Contract: 2. Log Entry Conversion | |
| Responsibilities | Collaborations |
| · Converting the text that was extracted and validated from the Log File into a Log Entry. | ETL Service |

### 

## Class Description - Configuration

Contains the general configuration of the project. This includes - the path of each of the team’s directory, the current working directory and any other information that affects the responsibilities within the Intake Component.

|  |
| --- |
| Class Name: Configuration |
| Superclass: N/A |
| Subclasses: N/A |
| Private Responsibilities· Holds the configuration for the paths that the Intake Controller is going to use to extract the log files.· Contains the TMUX file that is going to be used in the Cleansing Service |

## Class Description - OCR Service

Just like the Transcription Service, the OCR Service is going to grab Log Files that are being processed during the ingestion process that are images. The OCR is going to take those images and extract the text from the file. Just before converting the text into Log Entries with the ETL Service, it's going to cleanse the text with the use of the Cleansing Service.

|  |  |
| --- | --- |
| Class Name: Transcription Service | |
| Superclass: N/A | |
| Subclasses: N/A | |
| Private Responsibilities· The transformation of image files into Log Entries.· Through the use of the Cleansing Service, cleanse the text that is extracted from the Log Files. | |
| Contract: 1. Extraction of Image files | |
| Responsibilities | Collaborations |
| 1. During the ingestion process, it's going to take in the image files from the Intake Controller. | Intake Controller |
| Contract: 2. Cleansing Text | |
| Responsibilities | Collaborations |
| 2. Cleansing the text that is being extracted from the Log Files, using the Cleansing Service. | Cleansing Service |
| Contract: 2. Log Entry Conversion | |
| Responsibilities | Collaborations |
| Converting the text that was extracted from the Log Files into Log Entries | ETL Service |

### 

## Class Description - Cleansing Service

The Cleansing Service is going to remove many unwanted series of characters from the Log Files. Furthermore, it's going to take in a TMUX script (which is defined in the configuration file) that applies that cleansing to each of the Log Files.

|  |  |
| --- | --- |
| Class Name: Cleansing Service | |
| Superclass: N/A | |
| Subclasses: N/A | |
| Private Responsibilities· Removing the series of unwanted characters in each of the Log Files.· Applying the TMUX script to each of the Log Files. | |
| Contract: 1. Grab Configuration | |
| Responsibilities | Collaborations |
| 1. Grabbing the location of the TMUX from the given path; this script is going to be applied to each of the Log Files. | Configuration |

## Class Description - ETL Service

The ETL Service is going to communicate with the SplunkDAO which converts Log Files into Log Entries, Filters the Log Entries with a given pattern, and returns the Log Entries that matched a given pattern.

|  |
| --- |
| Class Name: Transcription Service |
| Superclass: N/A |
| Subclasses: N/A |
| Private Responsibilities· Converting Log Files into Log Entries.· Filtering a set of Log Entries through the comparison of a given pattern, for example, the filtering of Time Stamp· Searching a set of Log Entries with a given pattern. |

### 

### 

# Detailed Description of VectorDB Component

<< For each component or subsystem listed in section 2.2, add a section with a detailed design description of the component.>>

## Component Description

## Class Description - IntakeController

This class is going to be responsible for handling

<< For each class in this component, create a new Heading 2 subsection. In each subsection, describe the internal details of each design entity. Each class should have a class name, a description, super classes, and private responsibilities. >>

### Contract <contract 1 name>

<< For each contract supported by this class, create a new Heading 3 subsection. Give the detailed contract descriptions here. A contract description must have a contract identifier (used in the diagrams and in cross references), a contract name, a description, and a set of protocols. A protocol includes a list of responsibilities, method signatures to support those responsibilities, pre- and post-conditions, algorithm descriptions and collaborations. A collaboration must specify the contract supported by the service provider. >>

### Contract <contract 2 name>

## Class Description <class 2 name>

### Contract <contract 1 name>

vcs

# Detailed Description of VectorGeneration Component

<< For each component or subsystem listed in section 2.2, add a section with a detailed design description of the component.>>

## Component Description

Intaking the log files defined in a path and convert them into multiple log entries. Furtheremore,

## Class Description - IntakeController

This class is going to be responsible for handling

<< For each class in this component, create a new Heading 2 subsection. In each subsection, describe the internal details of each design entity. Each class should have a class name, a description, super classes, and private responsibilities. >>

### Contract <contract 1 name>

<< For each contract supported by this class, create a new Heading 3 subsection. Give the detailed contract descriptions here. A contract description must have a contract identifier (used in the diagrams and in cross references), a contract name, a description, and a set of protocols. A protocol includes a list of responsibilities, method signatures to support those responsibilities, pre- and post-conditions, algorithm descriptions and collaborations. A collaboration must specify the contract supported by the service provider. >>

### Contract <contract 2 name>

## Class Description <class 2 name>

### Contract <contract 1 name>