



Safety Analysis Management System (SAMS) Overview

Version 1.0

June 2014



Table of Contents

- 1. Introduction.....1**
 - 5.1 Document Organization 1
 - 5.2 Document Purpose 1
 - 5.3 Version Information..... 1
- 2. Project Summary2**
- 3. Application URL.....3**
- 4. Technical – Application Details.....4**
 - 4.1 Current Release Version: 4
 - 4.2 Hardware/Software Platform: 4
 - 4.3 Technical Summary: 4
 - 4.4 Software Components: 4
 - 4.5 Main Application Code Classes/Modules: 5
 - 4.6 Other SAMS Projects 6
 - 4.7 Things to know about SAMS for development purposes:..... 6
- 5. Technical – Database Details8**
 - 5.1. Schema Details..... 8
 - 5.2. Database Connection Details 8
 - 5.3. Database Utilization..... 9
 - 5.4. ETL Processes..... 9
 - 5.3.1 Nightly Staging 9
 - 5.3.2 Weekly QA Processing..... 9
 - 5.3.3 Notes/Special Concerns 10
 - 5.3.4 FARS Data..... 10
- 6. Training Plan.....11**



1. Introduction

5.1 Document Organization

This document is separated into six sections that are defined below.

1. Introduction
- 2.



4. Project Summary
5. Application URL
- 6.



- 7. Technical – Application Details
- 8.



9. Technical – Database Details

10.



11. Training Plan

5.2 Document Purpose

The purpose of this document is to provide a high level overview of the components that make up the SAMS application, including hardware, software, and each of the Visual Studio projects that are part of the overall SAMS solution. This document also includes a training plan to help bring MDOT developers up to speed on maintaining the SAMS application.

5.3 Version Information

Version Num.	Edit Date	Edited By	Comments
1.0	June 3, 2014	Matt Hiland	Accept all changes and modify version.



2. Project Summary

SAMS is a safety analysis management system built for Mississippi DOT. SAMS is an intranet portal that includes services and supporting extract, transform, and load (ETL) functionality for nightly and weekly crash data loading, crash query, results display in map and data grid, crash data analysis, reporting, and collision diagramming.



3. Application URL

- Test Environment (**MDOT TEST**): <http://APTESTSRV02/SAMS/> (latest version released is accessible here)
- Production Environment (**MDOT PROD**): <http://APPRODSRV04/SAMS/> (latest version released is accessible here)



4. Technical – Application Details

4.1 Current Release Version:

- **MDOT TEST:** SAMS 1.0.11
- **MDOT PROD:** SAMS 1.0.11

4.2 Hardware/Software Platform:

In the server environment, the SAMS web application is hosted by Windows 2003 Server/IIS 6.0 using the Microsoft .Net 2.0 framework. The Oracle databases are hosted in Oracle 10g server running on Solaris. The standard Oracle 10.2 client is used to communicate with the Oracle database. Currently the source code resides in the MDOT code repository.

4.3 Technical Summary:

The SAMS application is comprised of the following:

- Web application
- Data staging service
- Data loader host service and associated data loaders
- QA process (for comparisons between MDPS crash database and MDOT SAMS data)
- Five Oracle schemas:
 - SAMS: The primary operational database
 - SAMS_DD: Application settings
 - SAMS_STAGE: Data staging functions
 - SAMS_QA: Quality assurance tables and processes
 - SAMS_CET: Operational database for the Crash Editing Tool (CET).

4.4 Software Components:

The following is a list of the software platforms and third party tools used to create SAMS and its associated projects.

Web Application

- The web application is built on ASP.Net Web Forms using Microsoft VB.Net 2.0.
- The solution requires Visual Studio 2010.

Ajax Framework

- Asynchronous JavaScript calls are handled by the Ajax.Net Professional 9.2.17.1 library.
- <http://www.ajaxpro.info/>

Data Grids

- SAMS data grids are implemented using Infragistics data grid components (Infragistics NetAdvantage ASP.NET 2009.2).
- <http://www.infragistics.com/>

Data Staging

- The SAMS data staging service is implemented using Microsoft VB.Net 2.0.

Data Loading



- The SAMS data loader host service and components are implemented using Microsoft VB.Net 2.0.

Map Client

- Maps are displayed in Internet Explorer using the Adobe SVG Viewer 3.0.
- <http://www.adobe.com/svg/viewer/install/>

Mapping

- Maps are generated using Geomedia WebMap Pro 6.00.34.94. On-site imagery mapping is performed by GeoMedia TerraShare version 2.3.3.

Reporting/Map Printing

- Reporting, MUCR Diagram output, and map printing functions are accomplished using ActiveReports 5.2.3.685.
- <http://www.datadynamics.com/>

Collision Diagramming

- The collision diagramming subsystem was supplied by Visual Statement.
- <http://www.visualstatement.com/en/Home.aspx>

4.5 Main Application Code Classes/Modules:

The SAMS application code resides in 6 related .Net projects. Two additional projects exist as part of the solution but have not been fully implemented into the SAMS application (SAMS.Core and SAMS.Reporting).

SAMS

- This is the main web project containing the web forms and the majority of the business logic for the application.

SAMS.Analysis

- This project contains code used to generate classes for Homogeneous Locations Analysis.

SAMS.DAO

- This project contains classes for queries and stored procedure calls.

SAMS.DataAccess

- This project contains the basic data access classes for calling Oracle stored procedures and performing Oracle commands and queries.

SAMS.Map

- This project contains classes for performing SAMS mapping functions. Mapping functions are shared between these classes and the mapping classes in the main SAMS web project.

SAMS.Test

- This project contains unit test and supporting classes for all of the above projects.

SAMS.Core (incomplete)



- This project contains interface definitions, enumerations, and shared classes. Several of the interfaces and classes are duplicated in the SAMS project. The original plan was to reference the Core project and remove the classes from the SAMS project, but this will not be implemented.

SAMS.Reporting (incomplete)

- This project contains classes supporting the map printing functions. The plan was to integrate the remaining report functions in ReportBuilder to the SAMS.Reporting project. Currently SAMS reporting is based off of the ReportBuilder Project and does not rely on this project.

SAMS.DB

- This folder contains the DDL for the indexes, packages, tables, sequences, types, and views for the SAMS, SAMS_DD, SAMS_STAGE, SAMS_QA, and SAMS_MIGRATE Oracle databases.

SAMS.Help

- This folder contains the help project created using MadCap Flare.

4.6 Other SAMS Projects

ReportBuilder

- This project contains the report generation classes.

SAMS_Data_Staging

- This project contains the code for the SAMS data staging service.

DataLoader

- This project contains the code for the SAMS data loader hosting service and data loader processes.

DataStats

- This project contains the code required to run the SAMS QA process.

4.7 Things to know about SAMS for development purposes:

- Each Infragistics webgrid has its own javascript file. When making changes to one webgrid, make sure that you change the code in all webgrid javascript files if the change needs to apply to the other webgrids.
- The web application follows the normal methodology of asp.net coding. There is a .aspx page, a .aspx.vb page and an aspx.designer.vb page.
- Almost all database access is performed through the SAMS.DAO project.
- The SAMS.DataAccess project encapsulates the database communication in order to make communication with the database less complicated throughout the application code.
- The solution requires Visual Studio 2010.



- Oracle 10g client must be installed on any machine being used for development.
- The web application uses framesets to create different areas in the browser. One frameset contains the top navigation bar, another contains the left navigation bar, etc. Each of these framesets can be accessed from the other framesets in order to cause them to refresh or retain values or state.
- The basic and advanced queries store the queries into an xml file, where they are saved into the database into a clob.
- The Release version displayed on the Default.aspx page gets its value from the web.config. Anytime the application gets updated, that value needs to be updated as well.
- The web application relies heavily on internet browser popups, so popup blockers need to be disabled.
- SAMS was designed for Internet Explorer only, and will not work properly on other browsers. SAMS will currently work properly with IE 8 and below. Specifically IE7 and IE8. It has not been fully tested with IE9.
- The help documentation was created using MadCap Flare.
- Launching Homogeneous analysis or sliding scale analysis will call a stored procedure in the database that adds that request to a queue, which will execute in the order they are placed in the queue. The app simply loads that queue with requests and does not actually perform the analysis itself.
- The Admin/Column Selector Admin page allows the user to select columns to be displayed in the data grid. This list is populated by retrieving the column names from the database.
- FARS (Fatality Analysis Reporting System) DataLoader
 - The Data Loading/Load Data page/Other Data Types page allows the user to load FARS data into the database.
 - The web component communicates with a windows service that runs continuously on the web server.
 - The web component will manipulate the xml files and upload the files necessary in order to load a FARS data file. This file structure is difficult for users to manage by hand, so the process of managing the xml files was automated by the web server.



5. Technical – Database Details

5.1. Schema Details

Schema Name	Description	DB Connection Name
SAMS_STAGE	ETL Schema for bringing in nightly crash file from Department of Public Safety (DPS). This schema includes packages to assist with aliasing and snapping. Refer to SAMS Data Staging Admin Guide 2014 for further details on the staging process.	MDOT TEST: SAMSTEST MDOT PROD: SAMSPROD
SAMS_DD	This schema contains domain tables and lookups used by the application. This is similar to an ADMIN schema.	MDOT TEST: SAMSTEST MDOT PROD: SAMSPROD
SAMS	Main application schema used by SAMS. It contains crash data, road and intersection information, analysis results and saved queries.	MDOT TEST: SAMSTEST MDOT PROD: SAMSPROD
SAMS_QA	Isolated ETL schema for processing a weekly QA data load, which verifies data loading from the last 2 calendar years.	MDOT TEST: SAMSTEST MDOT PROD: SAMSPROD
SAMS_CET	As Crashes are cleansed through MDOT cleansing process they are loaded here and used as part of the nightly staging process to preserve the cleansed data. Refer to SAMS Database Admin Guide for further details on the cleansing process.	MDOT TEST: SAMSTEST MDOT PROD: SAMSPROD

5.2. Database Connection Details

```
SAMSPROD =  
(DESCRIPTION =  
  (ADDRESS_LIST =  
    (ADDRESS = (PROTOCOL = TCP)(HOST = 10.1.1.41)(PORT = 1521))  
  )  
  (CONNECT_DATA =  
    (SERVICE_NAME = SAMSPROD)))
```

```
SAMSTEST =  
(DESCRIPTION =  
  (ADDRESS_LIST =  
    (ADDRESS = (PROTOCOL = TCP)(HOST = 10.1.1.48)(PORT = 1525))  
  )
```



```
(CONNECT_DATA =  
  (SERVICE_NAME = SAMSTEST)))
```

5.3. Database Utilization

Most of the SAMS application utilizes the SAMS database schema. In some cases from the Admin section of the SAMS site, portions of the SAMS_DD schema are utilized to manage user credentials and to define application wide domain lookups.

5.4. ETL Processes

SAMS Data Staging processes occur in two categories:

5.3.1 Nightly Staging

Crash data is loaded nightly at MDOT from a downloaded MS Access warehouse built by Mississippi Department of Public Safety (MDPS). Each web server (in DEV, TEST, and PROD) has a data staging service installed that picks up the file from the MDPS server each evening, with a file naming convention as follows:

Nightly file: ms_part_11_1_2009.rar

Once the file is downloaded, using VB.net code within the staging service it is extracted from the .rar format and loaded into Oracle. Once the raw data is in Oracle, the processing is handled by procedures in the SAMS_STAGE.SAMS_STAGE_LOAD package. Once the load has completed, the SAMS_DATA_OPS email group at MDOT is notified with a summary email sent also from the SAMS_STAGE schema in Oracle.

5.3.2 Weekly QA Processing

On a weekly basis, the SAMS QA process runs to download a full file of crashes from the current year and all of the last calendar year. This file is meant to be used as a catch up mechanism when staging outages occur for longer than a few days. It is also useful for verification that data processed in the past are loaded into the system correctly or for catching up disparate date ranges of missing records. The QA Process runs in the SAMS_QA schema in Oracle. The initial operation is similar to the nightly load, in that a .rar file is pulled down, and within it are multiple MS Access databases holding the accident data. This data is then loaded in raw format into the SAMS_QA tables. Once all access files have been processed, the staging procedure is very similar to that of the nightly staging, but the process doesn't go all the way through but stops at a point where the data is in the same format as the nightly staging records and missing records can be identified between SAMS_QA and SAMS, then a summary email is sent to the SAMS_DATA_OPS group. Once the Admins of the system review the email, they can run a stored procedure, SAMS_QA.SAMS_STAGE_LOAD.PROCESS_MISSING_RECORDS, to load the missing records into SAMS. This routine pushes only the missing records into SAMS_STAGE from SAMS_QA and then launches the staging components of the nightly staging process that are necessary for completion.



For more information on all SAMS staging processes and how to administer them, see the SAMS Data Staging Admin guide.

5.3.3 Notes/Special Concerns

5.3.4 FARS Data

FARS Data is loaded and some SAMS Injury Severity values are overridden using this data. The best way to see the history for a crash record is to look at the SCID_AUDIT table in the SAMS_DD schema and query by SAMS_CRASH_ID. This audit table lists all operations that have been performed on the data since it has been in the system or since the SCID Audit table came into existence.



6. Training Plan

- Establish an MDOT Development Environment on a Client Machine
 - Install appropriate software and verify the code will compile
- Verify that the database on the MDOT Oracle development Server is up to date with System Test and Production
- Determine client level of expertise on:
 - VB.net
 - JavaScript
 - HTML
 - GeoMedia/GeoMedia Web Map
 - Oracle stored procedures
- Based on level of expertise determined in previous step, schedule four to five two-hour training sessions to review the following:
 - Session 1
 - Database Design
 - Review of Schemas
 - Data Staging Process
 - Data Loading Service
 - Stored Procedures for staging crashes
 - CET Copy/Preserve
 - FARS Data Loader
 - Session 2
 - SAMS User Interface
 - Query Tool
 - Mapping
 - Session 3
 - SAMS Analysis
 - Homogeneous Section
 - Homogeneous Intersection
 - Sliding Scale
 - Countermeasures
 - Session 4



- System Administration
- Data Administration
- Session 5
 - Open Discussion on miscellaneous items
 - Reports
 - Collision Diagrams