



SCIEx GIS COP Enhancement

Software Requirements Specification

Version 1.0

Prepared by Shaun Good

J2 Software Solutions, Inc

Table of Contents

Table of Contents	ii
Revision History	v
Review Signatures.....	vi
1 Introduction.....	1
1.1 Purpose.....	1
1.2 Document Conventions.....	1
1.2.1 Requirements Syntax	1
1.2.2 Requirement Priorities	1
1.3 Intended Audience and Reading Suggestions	1
1.4 Project Scope.....	2
1.5 References.....	2
2 Overall Description.....	2
2.1 Product Perspective	2
2.2 Product Features	4
2.3 User Classes and Characteristics	7
2.4 Operating Environment.....	7
2.4.1 Hardware.....	7
2.4.2 Software	7
2.5 Design and Implementation Constraints.....	7
2.6 Assumptions and Dependencies	7
2.6.1 Assumptions	7
2.6.2 Dependencies.....	8
3 Enhancement Features	8
3.1 C1: Flex Viewer Interface.....	8
3.1.1 C1-F1: Person Location GIS COP Message.....	8
3.1.2 C1-F2: Incident Location GIS COP Message.....	9
3.1.3 C1-F3: Incident Information GIS COP Message	10
3.1.4 C1-F4: Token Authentication Value (User Identifier)	11
3.2 C2: LEOS Database.....	11
3.2.1 C2-F1: Map Plot List Helper Table	12

- 3.2.2 C2-F2: Save Selected Person Locations Stored Procedure 13
- 3.2.3 C2-F3: Save Selected Incident Locations Stored Procedure 14
- 3.2.4 C2-F4: Retrieve Selected Person Locations Stored Procedure 16
- 3.2.5 C2-F5: Retrieve Selected Incident Locations Stored Procedure 17
- 3.3 C3: LEADR Database 18
 - 3.3.1 C3-F1: Retrieve Selected Person Locations Stored Procedure 18
 - 3.3.2 C3-F2: Retrieve Selected Incident Locations Stored Procedure 21
- 3.4 C4: Activity Web Service 23
 - 3.4.1 C4-F1: Save Selected Person Location Message 23
 - 3.4.2 C4-F2: Save Selected Incident Location Message 24
 - 3.4.3 C4-F3: Load Selected Person Location Message 25
 - 3.4.4 C4-F4: Load Selected Incident Location Message 25
 - 3.4.5 C4-F5: Incident Information GIS Cop Message Ingestion 26
 - 3.4.6 C4-F6: Save Selected Person Locations Route 27
 - 3.4.7 C4-F7: Save Selected Incident Locations Route 28
 - 3.4.8 C4-F8: Load Selected Person Locations Route 30
 - 3.4.9 C4-F9: Load Selected Incident Locations Route 31
- 3.5 C5: LEADR User Interface 33
 - 3.5.1 C5-F1: Name Details Result Page Modifications 33
 - 3.5.2 C5-F2: Name Activity Result Page Modifications 34
 - 3.5.3 C5-F3: Vehicle Result Page Modifications 36
 - 3.5.4 C5-F4: Vehicle Activity Result Page Modifications 36
 - 3.5.5 C5-F5: Property Result Page Modifications 37
 - 3.5.6 C5-F6: Location Incident Activity Result Page Modifications 37
 - 3.5.7 C5-F7: Location Person Activity Result Page Modifications 37
 - 3.5.8 C5-F8: Narrative Result Page Modifications 38
 - 3.5.9 C5-F9: Case Number Result Page Modifications 38
 - 3.5.10 C5-F10: Map Plot List Screen 39
- 4 External Interface Requirements 41
 - 4.1 User Interfaces 41
 - 4.2 Hardware Interfaces 41
 - 4.3 Software Interfaces 41

- 4.4 Communications Interfaces..... 41
 - 4.4.1 Web Services 41
 - 4.4.2 Internal Web Page Operation..... 42
 - 4.4.3 External Web Page Operation 42
- 5 Nonfunctional Requirements 42
 - 5.1 Open-Source..... 42

Revision History

Name	Date	Reason For Changes	Version
Shaun Good		Original Draft	1.0

Review Signatures

Name	Date	Status	Signature
CPT Buddy Wilkes		Review/Approve	
Jay Good		Review/Approve	
David Lutfy		Review/Approve	
Teresa Woods		Review/Approve	
Shaun Good		Review	

1 Introduction

1.1 Purpose

SCIEx is an existing law enforcement information exchange hosted by SLED (South Carolina Law Enforcement Division). The SCIEx GIS COP Enhancement will provide law enforcement officers, investigators, and analysts the functionality to map information from their SCIEx searches. With the mapping functionality SCIIC analysts will be able to analyze collected data to discover patterns or abnormalities that could assist in linking terrorists or everyday crimes before they become a threat. This enhancement will provide law enforcement officers or investigators with the right information at the right time, which will improve public safety. It will be accomplished via the use of an interface customized to consume the Flex Viewer Application (mapping software application).

The purpose of this document is to define the detailed software requirements for the new SCIEX GIS COP Enhancement (interface) herein referred to as GIS COP.

1.2 Document Conventions

1.2.1 Requirements Syntax

The requirements in this document are broken down by component first, then feature, and then the requirement. The numbering system used will be as follows:

Cx-Fy-Rz

“C” stands for component and is followed by x, which is the sequential number of the component; if there are 10 components, then x will range from 1 to 10.

“F” stands for feature and is followed by y, which is the sequential number of the feature; if there are 6 features for a component, then y will range from 1 to 6.

“R” stands for requirement and is followed by z, which is the sequential number of the requirement; if there are 12 requirements for a feature, then z will range from 1 to 12.

1.2.2 Requirement Priorities

Each requirement will receive one of the following priorities:

- Critical – This feature is critical to the operation of the software.
- High – This feature is not critical but is of high priority.
- Medium – This feature is of medium priority.
- Low – This feature is of low priority.

1.3 Intended Audience and Reading Suggestions

The intended audiences for this document are software developers and project managers. It is suggested that the Business Requirements Document be reviewed prior to this document.

1.4 Project Scope

This project will provide improved law enforcement intelligence analysis by adding the ability to perform the following:

- ◆ Multiple queries can be run and selection of up to 500 results in which the user intends to have plotted on a map (Flex Viewer Application).
- ◆ Upon visiting the Map Plot List screen, the user can then adjust their final location selection and either plot them on a map or clear the list of locations and start over. If the user chooses to plot the results (locations), they will display as points on a GIS map maintained by SLED (Flex Viewer Application). This is accomplished by passing the location and other pertinent information via an XML message to a SLED supplied url. An interface, in the Flex Viewer Application, is currently being developed by SLED to accept these XML messages.
- ◆ If a map is currently open (Flex Viewer Application) and the user plotted the locations from the SCIEx system, they should be able to retrieve the SCIEx incident data by clicking on a plotted incident report location in the Flex Viewer Application. This is accomplished by passing the SCIEx incident record's unique identifier and user token, which was originally passed to the Flex Viewer Application when the user chose to plot their results, via an XML message. This XML message is sent from the Flex Viewer Application to a SCIEx url.

1.5 References

This document contains references to the SCIEX South Carolina Information Exchange System Business Requirements and the SCIEX GIS COP Enhancement Software Requirements Specification. To obtain a copy of the SCIEX South Carolina Information Exchange System Business Requirements or SCIEX GIS COP Enhancement Software Requirements Specification, please contact Teresa Woods at twoods@J2ss.com.

2 Overall Description

2.1 Product Perspective

This enhancement will be applied to the existing open-source LEADR system and will also be open-source and contained within the LEADR source code set. The user will have the ability to map results from the SCIEx incident and person location records that they have selected. The user will also have the ability to click on an incident location on the map (Flex Viewer Application) and view the SCIEx incident data. Below are some diagrams to illustrate this concept.

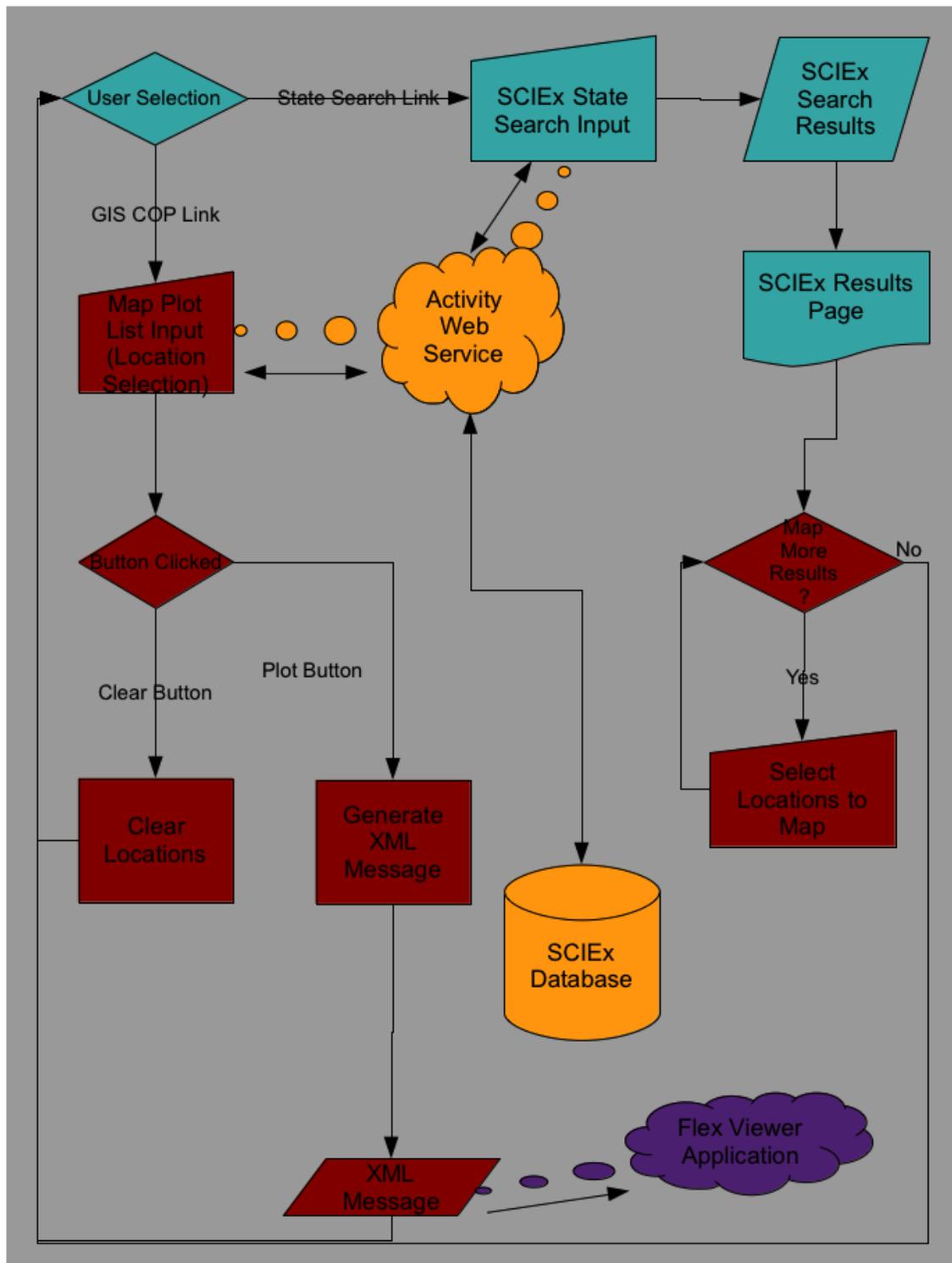


Figure 1: Illustrates how the LEADR system should interface with the Flex Viewer Application.

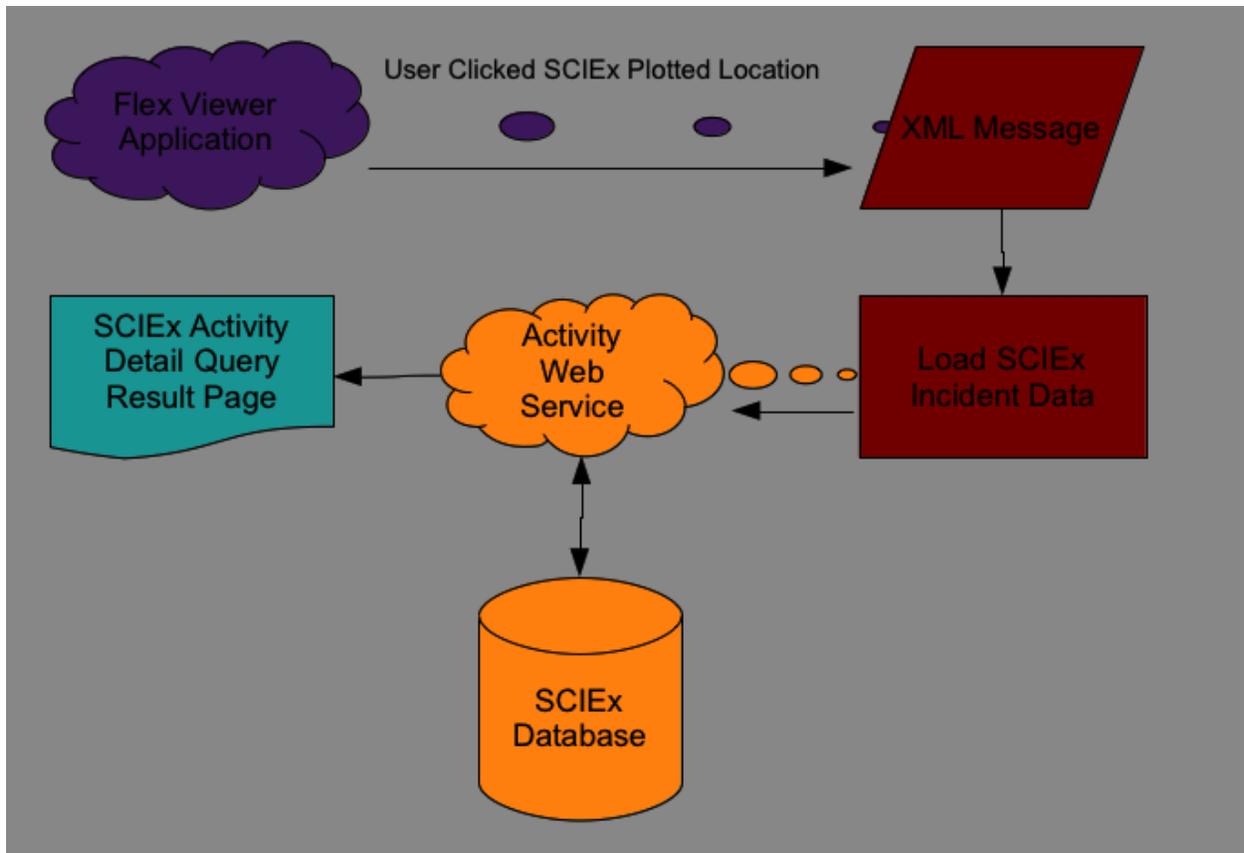


Figure 2: Illustrates how the Flex Viewer Application should interface with LEADR.

2.2 Product Features

This system will have the following features (including the effected components).

- LEOS Database
 - A new table will be added as a means of storing saved activity (incident/arrest) and person location information that a user wished to later, plot on a map.
 - A stored procedure will be added which is used for saving activity (incident/arrest) location map plotting information.
 - A stored procedure will be added which is used for saving person location map plotting information.
 - A stored procedure will be added which is used for loading activity (incident/arrest) location map plotting information.
 - A stored procedure will be added which is used for loading person location map plotting information.
- LEADR Database
 - A stored procedure will be added which is used for loading activity (incident/arrest) location records.
 - A stored procedure will be added which is used for loading person location records.
- Activity Web Service

- The following XML messages will be designed to save and retrieve selected locations that the user intends to have plotted on a map:
 - A Save Selected Person Location XML message will be developed that contains the following and is intended to save one or more person location records selected by the current user:
 - The current user.
 - The address.
 - A deletion flag allowing for the deletion of a person location already saved to a user’s list of saved person locations.
 - A Save Selected Incident Location XML message will be developed that contains the following and is intended to save one or more activity location records selected by the current user:
 - The current user.
 - The unique identifier of an activity (incident/arrest) record.
 - Vin and Tag number (vehicle only).
 - A deletion flag allowing for the deletion of an activity (incident/arrest) location already saved to a user’s list of saved activity locations.
 - A Load Selected Person XML message will be developed that contains the following and is intended to load all person locations that the current user has previously selected:
 - Current user
 - A Load Selected Incident XML message will be developed that contains the following and is intended to load all activity (incident/arrest) location records that the current user has previously selected:
 - Current user
- The following Routes will be added to accommodate the previously listed XML messages:
 - Save Selected Person Location route
 - Save Selected Incident Location route
 - Load Selected Person Location route
 - Load Selected Incident Location route
- Flex Viewer Interface
 - The following XML messages will be designed to send and receive data from the Flex Viewer Application:
 - An Incident Location XML message will be developed that contains the following information for each activity (incident/arrest) record location selected by the user for plotting on a map:
 - A token value used in identifying a LEADR user.
 - Address information.
 - The activity (incident/arrest) record’s unique identifier.

- Summary activity (incident/arrest) information that will display to the user, in the Flex Viewer Application, when a location is clicked on.
 - A Person Location XML message will be developed that contains:
 - A token value used in identifying a LEADR user.
 - Address information.
 - Summary person information that will display to the user, in the Flex Viewer Application, when a location is clicked on.
 - An Incident Information XML message will be developed that contains:
 - A token value used in identifying a LEADR user.
 - The activity (incident/arrest) record's unique identifier.
 - A token value will be added that is used in identifying LEADR users when XML messages are sent to and from the Flex Viewer Application. The token value is used to properly identify a user (essentially authenticating a user when data is sent between both applications).
- LEADR User Interface
 - Implement the existing GIS COP link to perform the following actions:
 - If the user has not selected any locations to plot on a map via the LEADR state search result screens and the user clicks the GIS COP navigation link, the Flex Viewer Application will open. It will then allow the user to search and plot locations on a map.
 - If the user has selected locations to plot on a map via the LEADR state search result screens, the Map Plot List screen is displayed. This screen provides the following functionality:
 - The user can adjust their location selections.
 - The user can clear all of their selected locations and start over.
 - The user can click the Plot button that sends an XML message to the Flex Viewer Application. When this occurs, the Flex Viewer Application will open displaying the mapped locations and associated information.
 - Listed below are existing screen design modifications that will be made:
 - A new column titled Map will be added to the following screens. This new column will contain a checkbox. The user will check this checkbox if they choose to map the activity (incident/arrest) record locations:
 - name_activity_query.jsp
 - vehicle_query_results.jsp
 - vehicle_activity_results.jsp
 - property_query_results.jsp
 - location_incident_activity_query.jsp
 - location_person_activity_query.jsp
 - narrative_query_results.jsp
 - case_number_query_results.jsp

- A select all checkbox will be added to the same screens listed above. It will exist as the first row of each result page. When this checkbox is checked, all remaining results on that page are selected and saved. When this checkbox is unchecked, all remaining results on that page are deselected and removed from the user's list of saved incident locations.
- On the name_details.jsp page, a checkbox will be added next to each row of the Last Known Address section of the screen. When each checkbox is checked, the person's location is saved.
- A new screen will be added containing all selected (saved) person and activity (incident/arrest) locations. This screen will contain:
 - A button to clear the user's selected locations.
 - A button to plot the user's selected locations on a map. When this button is clicked, all locations checked on this screen are plotted to a map.
 - A checkbox, in the left most column, allows the user to select/deselect their saved locations.
 - A select all checkbox is located as the first row of the saved locations. When this checkbox is checked, all of the saved locations listed are selected. When this checkbox is unchecked, all of the saved locations listed are deselected.

2.3 User Classes and Characteristics

The majority of the software development tasks associated with this enhancement pertain to sending XML messages to and from the Flex Viewer Application via a SLED supplied url, internal Save and Load selected location operations, and hidden user interface functionality (invisible to the user). The only changes noticeable to the user are noted in the LEADR User Interface bullet in section 2.2 of this document. The users of the enhanced system will remain the same as the existing LEADR system.

2.4 Operating Environment

2.4.1 Hardware

There are not any additional hardware requirements for this enhancement to be successfully deployed.

2.4.2 Software

The Flex Viewer Application needs to be network accessible from LEADR via a SLED supplied url and allow XML messages to be sent to it. It must also be able to send network traffic to the existing LEADR system via a J2 Software Solutions supplied url.

2.5 Design and Implementation Constraints

There are not any design or implementation constraints associated with this enhancement.

2.6 Assumptions and Dependencies

2.6.1 Assumptions

The following assumptions are being made with this enhancement:

- The Flex Viewer Application is being enhanced to be able to accept Incident Location XML messages from the LEADR system as outlined in section 2.2 of this document.
- The Flex Viewer Application is being enhanced to be able to accept Person Location XML messages from the LEADR system as outlined in section 2.2 of this document.
- The Flex Viewer Application is being enhanced to be able to send Incident Information XML messages to the LEADR system as outlines in section 2.2 of this document.
- SLED will be configuring their network to allow the resulting aforementioned XML traffic before the implementation of this enhancement.
- The SCIEX (LEADR) system will be upgraded to version 5.8.2 or later prior to the deployment of this enhancement.

2.6.2 Dependencies

The map plotting functionality is dependent upon the ability of the Flex Viewer Application to accept and send the XML messages, listed in the previous subsection of this document, from and to the existing LEADR system via SLED and J2 Software Solutions supplied urls. It is also dependent upon the new functionality of the Activity Web Service and LEADR user interface as defined in the SCIEX GIS COP Enhancement Software Requirements Specification.

3 Enhancement Features

3.1 C1: Flex Viewer Interface

This component contains the XML messages that will be sent to and from the Flex Viewer Application as well as development of the token value used to identify a LEADR user.

3.1.1 C1-F1: Person Location GIS COP Message

3.1.1.1 Description

This message is used to interface with the Flex Viewer Application. The information, contained in this message, is used to plot person locations on a map in the Flex Viewer Application and should contain the following information. The information stored in this message will come from a query or queries loading the current user's selected person location records (originally selected from the name_details.jsp State Search result web page). The population of this message is handled in other feature definitions of this document.

- User Identifier – This field will contain a token identifier developed as part of feature C1-F4 of this document. This token identifier uniquely identifies the current user and is encrypted.
- Last Name – The last name of the person associated with the location record.
- First Name – The first name of the person associated with the location record.
- Middle Name – The middle name of the person associated with the location record.
- Suffix – The name suffix of the person associated with the location record.
- Full Address – The entire street address (excluding city, state, and zip) of the location record.
- Street Number – The street number field of the location record.
- Street Name – The street name field of the location record.

- City – The city field of the location record.
- State – The state field of the location record.
- Zip Code – The zip code field of the location record.
- Latitude– The latitude field of the location record.
- Longitude – The longitude field of the location record.

3.1.1.2 Priority

This is a critical feature.

3.1.1.3 Functional Requirements

3.1.1.3.1 C1-F1-R1: XML Format

The message shall be designed in XML format and be able to support multiple person locations in a single message. The User Identifier field should only be contained once per message regardless of how many locations are contained within. Although an XML message without an XML declaration above the root element is considered mal-formed, it is important not to include the XML declaration in this XML message. This is a requirement of the LEADR system.

3.1.2 C1-F2: Incident Location GIS COP Message

3.1.2.1 Description

This message is used to interface with the Flex Viewer Application. The information, contained in this message, is used to plot activity (incident/arrest) locations on a map in the Flex Viewer Application and should contain the following information. The information stored in this message will come from a query or queries loading the current user's selected activity (incident/arrest) location records (originally selected from the name_activity_query.jsp, vehicle_query_results.jsp, vehicle_activity_query.jsp, property_query_results.jsp, location_incident_activity_query.jsp, location_person_activity_query.jsp, narrative_query_results.jsp, or case_number_query_results.jsp State Search result web pages). The population of this message is handled in other feature definitions of this document.

- User Identifier – This field will contain a token identifier developed as part of feature C1-F4 of this document. This token identifier uniquely identifies the current user and is encrypted.
- Entity ID – The unique identifier of the activity record.
- Activity Type – The Activity Type field of the activity record. (i.e. INCIDENT, ARREST or CUSTODY for Incident, Arrest or Juvenile Arrest records).
- Activity Number – The Activity Number (incident number) field of the activity record.
- Activity Date Month – The month value of the Activity Date field of the activity record.
- Activity Date Day – The day value of the Activity Date field of the activity record.
- Activity Date Year – The year value of the Activity Date field of the activity record.
- Charge Description – The description of a charge associated with the activity record.
- Ucr Code – The ucr code field of a charge associated with the activity record.
- Full Address – The entire address of the location record.
- Street Number – The street number field of the location record.

- Street Name – The street name field of the location record.
- City – The city field of the location record.
- State – The state field of the location record.
- Zip Code – The zip code field of the location record.
- Latitude – The latitude field of the location record.
- Longitude – The longitude field of the location record.

3.1.2.2 Priority

This is a critical feature.

3.1.2.3 Functional Requirements

3.1.2.3.1 C1-F2-R1: XML Format

The message shall be designed in XML format and be able to support multiple activity (incident/arrest) locations in a single message. Additionally, each activity (incident/arrest) location should be able to support multiple offenses as activity records can have multiple offenses. The User Identifier field should only be contained once per message regardless of how many activity locations are contained within. Although an XML message without an XML declaration above the root element is considered malformed, it is important not to include the XML declaration in this XML message. This is a requirement of the LEADR system.

3.1.3 C1-F3: Incident Information GIS COP Message

3.1.3.1 Description

This message is used to interface with the Flex Viewer Application. The information, contained in this message, is used to open the activity_detail_query.jsp page containing the requested activity (incident/arrest) record's information in the LEADR system and should contain the following information. The information stored in this message will come from the Flex Viewer Application as the result of a user requesting to open the LEADR activity (incident/arrest) record associated with a location plotted on a map.

- User Identifier – This field will contain a token identifier developed as part of feature C1-F4 of this document. This token identifier uniquely identifies the current user logged into the LEADR system and is encrypted.
- EntityID – The unique identifier of the activity record the user is requesting to load.

3.1.3.2 Priority

This is a critical feature.

3.1.3.3 Functional Requirements

3.1.3.3.1 C1-F3-R1: XML Format

This message should be designed in XML format and only support a single activity record. Although an XML message without an XML declaration above the root element is considered malformed, it is

important not to include the XML declaration in this XML message. This is a requirement of the LEADR system.

3.1.4 C1-F4: Token Authentication Value (User Identifier)

3.1.4.1 Description

Every message, used in communicating with the Flex Viewer Application, should use a system of uniquely identifying the current user of the LEADR system. This is to ensure that:

- There is not any unauthorized access to the LEADR system.
- There is not any unauthorized access to the Flex Viewer Application.
- The LEADR system can properly identify the current user of the system for any other reasons necessary.

This value will be stored in the UserIdentifier value defined in the above features of this component.

3.1.4.2 Priority

This is a critical feature.

3.1.4.3 Functional Requirements

3.1.4.3.1 C1-F4-R1: Value Contents

This value should contain the LDAP information normally sent as the RequestedUserID XML field of the other State Search routes (Activity Web Service) (i.e. CaseNumber route).

3.1.4.3.2 C1-F4-R2: Formula

The value, in the above requirement, should be altered so that, although the information is still stored in this value, it will have an extremely low probability of being the same twice (a unique value).

3.1.4.3.3 C1-F4-R3: Encryption

The value should be encrypted using a minimum of 128-bit encryption. This ensures that the LEADR system remains secure between the 2 systems.

3.1.4.3.4 C1-F4-R4: Implementation

The source code should be written in such a way that when the Map Plot List screen generates these messages (back-end source code), it could easily be incorporated (written as a separate class perhaps). This source code will also be used when a POST is made to the activity_detail_query.jsp from the Flex Viewer Application (resolving the User Identifier). (Please see C4-F5 for more details).

3.2 C2: LEOS Database

This component contains all database modifications that shall be made to the LEOS MySQL database. The LEOS database is an existing database in the current LEADR system. It is used as a working, logging, and messaging database. Its current roles are:

- The agency_data table is used for queuing data that has been submitted via data replication (XML messages). The data is later processed, and then deleted from the queue (inserting the data into the LEADR database (Microsoft SQL Server)).
- The agencysequence, and incremter tables are used for data replication sequence numbering.
- The transaction, transactiondetail, and query_log tables are used for logging all LEADR transactions.
- The notifications table is used to store:
 - All errors that occur during any LEADR transactions.
 - Which error messages were sent to the system administrator(s) via the LEADR Emailer windows service.
 - All secret flag email messages to be sent to the appropriate agencies.
- The announcements table is used to store announcements in which system administrators have entered for all users to view upon login.
- The group_permissions, permissions, and sysgroups tables are used as internal lookup tables for user interface permissions (uses LDAP to compare to tables for proper user permissions).
- The saved_searches table is used to store all recent and named searches for each user.

3.2.1 C2-F1: Map Plot List Helper Table

3.2.1.1 Description

This database table will be used to store the activity and person location records that each user selects. Please note that a users location selections are only stored until the user clicks the Clear button on the Map Plot List screen.

3.2.1.2 Priority

This is a critical feature.

3.2.1.3 Functional Requirements

3.2.1.3.1 C2-F1-R1: Table Columns

The columns for this table are as follows:

- **User ID** – The users LDAP information normally sent as the RequestedUserID XML field of the other State Search routes (Activity Web Service) (i.e. CaseNumber route).
- **Activity** – Stores the EntityID (record identifier) of a selected (by the user) activity (incident/arrest) record.
- **Vehicle** – Stores the VIN (vehicle identification number) and license plate number of a vehicle associated with a selected (by the user) activity (incident/arrest) record in the format: key=value;key=value...
- **Person Location** – Stores a complete selected (by the user) person location record in the format: key=value;key=value;...

3.2.1.3.2 C2-F1-R2: Overall Table Design

The table should be designed containing the fields previously mentioned and should be created as part of the existing LEOS database. A soft delete column should not be created (all deletes will be permanent (rows are actually erased)). This is due to the fact that this table is not supposed to store any permanent data.

3.2.1.3.3 C2-F1-R3: Update Database Script

An update script must be created to add this table to an existing database.

3.2.1.3.4 C2-F1-R4: Create Database Script

This table must be added to this database's create script. This allows all new installations to have the latest database version without running any new update scripts.

3.2.2 C2-F2: Save Selected Person Locations Stored Procedure

3.2.2.1 Description

This stored procedure will be used to save selected (by the user) person location records to the Map Plot List Helper Table. *Please see C2-F1 for more details about the Map Plot List Helper Table.*

3.2.2.2 Priority

This is a critical feature.

3.2.2.3 Stimulus/Response

Stimulus

This stored procedure is called passing in the User ID and Person Locations parameters.

Response

The data passed into the parameters is saved to the Map Plot List Helper table. No output is expected back from this stored procedure as it is used to merely save data. *Please see C2-F1 for more details.*

3.2.2.4 Functional Requirements

3.2.2.4.1 C2-F2-R1: Parameters

The parameters for this stored procedure are as follows:

(in) User ID – The users LDAP information normally sent as the RequestedUserID XML field of the other State Search routes (Activity Web Service) (i.e. CaseNumber route).

(in) Person Locations – The following semicolon separated key=value pairs (keys listed below). Each person location record should be pipe separated. The format is
key=value;key=value;...|key=value;key=value;...|

- Last Name – The value should be the person's last name that is associated with a selected (by the user) person location record.
- First Name – The value should be the person's first name that is associated with a selected (by the user) person location record.

- Middle Name – The value should be the person’s middle name that is associated with a selected (by the user) person location record.
- Address Entity Id – The value should be the person’s first name that is associated with a selected (by the user) person location record.
- Street Address – The value should be the complete street address of a selected (by the user) person location record.
- City – The value should be the city field of a selected (by the user) person location record.
- State – The value should be the state field of a selected (by the user) person location record.
- Zip Code – The value should be the zip code field of a selected (by the user) person location record.

(in) Deleted – This is a bit field set to either 1 or 0 (true or false). When set to 1, the specified selected person location is removed from the list of selected person location records.

(out) Return Value – No output is expected as this stored procedure is used to merely save data.

3.2.2.4.2 C2-F2-R2: Parameter Sizes

The size of the stored procedure parameters should be as follows:

User ID – varchar(128)

Person Locations – varchar(4000)

3.2.2.4.3 C2-F2-R3: Naming Convention

This stored procedure should be named sp_saveGisCopPersonLocations.

3.2.2.4.4 C2-F2-R4: Update Database Script

An update script must be created to add this stored procedure to an existing database.

3.2.2.4.5 C2-F2-R5: Create Database Script

This stored procedure must be added to this database’s create script. This allows all new installations to have the latest database version without running any new update scripts.

3.2.3 C2-F3: Save Selected Incident Locations Stored Procedure

3.2.3.1 Description

This stored procedure will be used to save selected (by the user) activity (incident/arrest) location records to the Map Plot List Helper Table. *Please see C2-F1 for more details about the Map Plot List Helper Table.*

3.2.3.2 Priority

This is a critical feature.

3.2.3.3 Stimulus/Response

Stimulus

This stored procedure is called passing in the User ID, Activities and Vehicles parameters.

Response

The data passed into the parameters is saved to the Map Plot List Helper table. No output is expected back from this stored procedure as it is used to merely save data. *Please see C2-F1 for more details.*

3.2.3.4 Functional Requirements**3.2.3.4.1 C2-F3-R1: Parameters**

The parameters for this stored procedure are as follows:

(in) **User ID** – The users LDAP information normally sent as the RequestedUserID XML field of the other State Search routes (Activity Web Service) (i.e. CaseNumber route).

(in) **Activities** – A pipe separated list of activity (incident/arrest) record EntityID's (unique identifier).

(in) **Vehicles** – The following semicolon separated key=value pairs (keys listed below). Each vehicle should be pipe separated. The format is key=value;key=value;... |key=value;key=value;... |

- RequestedVIN – The value should be the vehicle identification number (VIN) of the vehicle that is associated with a selected (by the user) activity (incident/arrest) record.
- RequestedTag – The value should be the license plate number of the vehicle that is associated with a selected (by the user) activity (incident/arrest) record.

(in) **Deleted** – This is a bit field set to either 1 or 0 (true or false). When set to 1, the specified selected incident location is removed from the list of selected person location records.

(out) **Return Value** – No output is expected as this stored procedure is used to merely save data.

3.2.3.4.2 C2-F3-R3: Parameter Sizes

The size of the stored procedure parameters should be as follows:

User ID – varchar(128)

Activities – varchar(4000)

Vehicles – varchar(4000)

3.2.3.4.3 C2-F3-R3: Naming Conventions

This stored procedure should be named sp_saveGisCopActivityLocations.

3.2.3.4.4 C2-F3-R4: Update Database Script

An update script must be created to add this stored procedure to an existing database.

3.2.3.4.5 C2-F3-R5: Create Database Script

This stored procedure must be added to this database's create script. This allows all new installations to have the latest database version without running any new update scripts.

3.2.4 C2-F4: Retrieve Selected Person Locations Stored Procedure

3.2.4.1 Description

This stored procedure will be used to retrieve selected (by the user) person location records from the Map Plot List Helper Table. *Please see C2-F1 for more details about the Map Plot List Helper Table.*

3.2.4.2 Priority

This is a critical feature.

3.2.4.3 Stimulus/Response

Stimulus

This stored procedure is called passing in the User ID parameter.

Response

All person location record information is returned for the specified User ID.

3.2.4.4 Functional Requirements

3.2.4.4.1 C2-F4-R1: Parameters

The parameters for this stored procedure are as follows:

(in) **User ID** – The information of the user that the person location record information needs to be retrieved for. The user information is the LDAP information normally sent as the RequestedUserID XML field of the other State Search routes (Activity Web Service) (i.e. CaseNumber route).

(out) **Last Name** – The person's last name that is associated with a selected (by the user) person location record.

(out) **First Name** – The person's first name that is associated with a selected (by the user) person location record.

(out) **Middle Name** – The person's middle name that is associated with a selected (by the user) person location record.

(out) **Address Entity** – The Entity ID of the address record that is associated with a selected (by the user) person location record.

(out) **Street Address** – The complete street address (excluding city, state, and zip code) of a selected (by the user) person location record.

(out) **City** – The city field of a selected (by the user) person location record.

(out) **State** – The state field of a selected (by the user) person location record.

(out) **Zip Code** - The zip code field of a selected (by the user) person location record.

3.2.4.4.2 C2-F4-R2: Parameter Sizes

The size of the stored procedure parameters should be as follows:

User ID – varchar(128)

Last Name – varchar(64)

First Name – varchar(64)

Middle Name – varchar(64)

Address Entity – nvarchar(255)

Street Address – varchar(1024)

City – varchar(64)

State – varchar(64)

Zip Code – varchar(64)

3.2.4.4.3 C2-F4-R3: Naming Conventions

This stored procedure should be named sp_getGisCopPersonLocations.

3.2.4.4.4 C2-F4-R4: Update Database Script

An update script must be created to add this stored procedure to an existing database.

3.2.4.4.5 C2-F4-R5: Create Database Script

This stored procedure must be added to this database's create script. This allows all new installations to have the latest database version without running any new update scripts.

3.2.5 C2-F5: Retrieve Selected Incident Locations Stored Procedure

3.2.5.1 Description

This stored procedure will be used to retrieve selected (by the user) activity (incident/arrest) location records from the Map Plot List Helper Table. *Please see C2-F1 for more details about the Map Plot List Helper Table.*

3.2.5.2 Priority

This is a critical feature.

3.2.5.3 Stimulus/Response

Stimulus

This stored procedure is called passing in the User ID parameter.

Response

All activity (incident/arrest) location record information is returned for the specified User ID.

3.2.5.4 Functional Requirements

3.2.5.4.1 C2-F5-R1: Parameters

The parameters for this stored procedure are as follows:

(in) **User ID** – The information of the user that the activity (incident/arrest) location record information needs to be retrieved for. The user information is the LDAP information normally sent as the RequestedUserID XML field of the other State Search routes (Activity Web Service) (i.e. CaseNumber route).

(out) **Entity ID** – The selected (by the user) activity (incident/arrest) record's Entity ID (unique identifier).

(out) **Vin** – The vehicle identification number (VIN) of the vehicle that is associated with a selected (by the user) activity (incident/arrest) record.

(out) **Tag** – The license plate number of the vehicle that is associated with a selected (by the user) activity (incident/arrest) record.

3.2.5.4.2 C2-F5-R2: Parameter Sizes

The size of the stored procedure parameters should be as follows:

User ID – varchar(128)

Entity ID – nvarchar(255)

Vin – varchar(64)

Tag – varchar(64)

3.2.5.4.3 C2-F5-R3: Naming Conventions

This stored procedure should be named sp_getGisCopActivityLocations.

3.2.5.4.4 C2-F5-R4: Update Database Script

An update script must be created to add this stored procedure to an existing database.

3.2.5.4.5 C2-F5-R5: Create Database Script

This stored procedure must be added to this database's create script. This allows all new installations to have the latest database version without running any new update scripts.

3.3 C3: LEADR Database

This component contains all database modifications that shall be made to the LEADR MS SQL Server database. The LEADR database is an existing database in the current LEADR system. It is used to store all fusion center data.

3.3.1 C3-F1: Retrieve Selected Person Locations Stored Procedure

3.3.1.1 Description

This stored procedure will be used to retrieve selected (by the user) person location records from the LEADR database based on the parameter values passed in.

3.3.1.2 Priority

This is a critical feature.

3.3.1.3 Stimulus/Response

Stimulus

This stored procedure is called passing in the Last Name, First Name, Middle Name, Address Entity, Street Address, City, State, and Zip Code parameters.

Response

All person location records, filtered by the parameters passed in, are fetched from the LEADR database. This is accomplished by running other LEADR database stored procedures from within this stored procedure.

3.3.1.4 Functional Requirements

3.3.1.4.1 C3-F1-R1: Parameters

The parameters for this stored procedure are as follows. Please note that for the input parameters, pipe characters must separate distinct records therefore; each input parameter should contain the same number of pipe characters.

(in) **Last Name** - The person's last name that is associated with a selected (by the user) person location record that a location record is to be retrieved for.

(in) **First Name** - The person's first name that is associated with a selected (by the user) person location record that a location record is to be retrieved for.

(in) **Middle Name** - The person's middle name that is associated with a selected (by the user) person location record that a location record is to be retrieved for.

(in) **Address Entity** - The Entity ID of the address record of a selected (by the user) person location record that a location record is to be retrieved for.

(in) **Street Address** - The complete street address (excluding city, state, and zip code) of a selected (by the user) person location record that a location record is to be retrieved for.

(in) **City** - The city field of a selected (by the user) person location record that a location record is to be retrieved for.

(in) **State** - The state field of a selected (by the user) person location record that a location record is to be retrieved for.

(in) **Zip Code** - The zip code field of a selected (by the user) person location record that a location record is to be retrieved for.

(out) **Last Name** - The person's last name that is associated with the retrieved location record.

(out) **First Name** - The person's first name that is associated with the retrieved location record.

(out) **Middle Name** - The person's middle name that is associated with the retrieved location record.

(out) **Full Address** – The Full Address field of the retrieved address.

(out) **Street Number** – The Street Number field of the retrieved address.

(out) **Street Name** – The Street Name field of the retrieved address.

(out) **City** – The City field of the retrieved address.

(out) **State** – The State field of the retrieved address.

(out) **Zip Code** – The Zip Code field of the retrieved address.

(out) **Latitude** – The X Coordinate field of the retrieved address.

(out) **Longitude** – The Y Coordinate field of the retrieved address.

3.3.1.4.2 C3-F1-R2: Parameter Sizes

The size of the stored procedure parameters should be as follows:

(in) **Last Name** - varchar(64)

(in) **First Name** - varchar(64)

(in) **Middle Name** - varchar(64)

Address Entity – nvarchar(255)

Street Address - varchar(1024)

(in) **City** - varchar(64)

(in) **State** - varchar(64)

(in) **Zip Code** - varchar(64)

(out) **Last Name** - varchar(64)

(out) **First Name** - varchar(64)

(out) **Middle Name** - varchar(64)

Full Address – varchar(2048)

Street Number – varchar(256)

Street Name – varchar(256)

(out) **City** – varchar(64)

(out) **State** – varchar(64)

(out) **Zip Code** – varchar(64)

Latitude – decimal

Longitude – decimal

3.3.1.4.3 C3-F1-R3: Naming Conventions

This stored procedure should be named sp_gisCopPersonLocation.

3.3.1.4.4 C3-F1-R4: Update Database Script

An update script must be created to add this stored procedure to an existing database.

3.3.1.4.5 C3-F1-R5: Create Database Script

This stored procedure must be added to this database's create script. This allows all new installations to have the latest database version without running any new update scripts.

3.3.2 C3-F2: Retrieve Selected Incident Locations Stored Procedure

3.3.2.1 Description

This stored procedure will be used to retrieve selected (by the user) activity (incident/arrest) location records from the LEADR database based on the parameter values passed in.

3.3.2.2 Priority

This is a critical feature.

3.3.2.3 Stimulus/Response

Stimulus

This stored procedure is called passing in the Entity, Vin and Tag parameters.

Response

All activity (incident/arrest) location records, filtered by the parameters passed in, are fetched from the LEADR database. This is accomplished by running other LEADR database stored procedures from within this stored procedure.

3.3.2.4 Functional Requirements

3.3.2.4.1 C3-F2-R1: Parameters

The parameters for this stored procedure are as follows. Please note that for the input parameters, pipe characters must separate distinct records therefore; each input parameter should contain the same number of pipe characters.

(in) **Entity** – The Entity ID (unique identifier) of a selected (by the user) activity (incident/arrest) record that a location record is to be retrieved for.

(in) **Vin** – The vehicle identification number (VIN) of a vehicle that is associated with a selected (by the user) activity (incident/arrest) record that a location record is to be retrieved for.

(in) **Tag** – The license plate number of a vehicle that is associated with a selected (by the user) activity (incident/arrest) record that a location record is to be retrieved for.

(out) **Entity ID** – The Entity ID (unique identifier) of the retrieved activity (incident/arrest) record.

(out) **Activity Type** – The Activity Type field of the retrieved activity (incident/arrest) record.

(out) **Activity Number** - The Activity Number field of the retrieved activity (incident/arrest) record.

(out) **Activity Date Month** – The month portion of the Activity Date field of the retrieved activity (incident/arrest) record.

(out) **Activity Date Day** – The day portion of the Activity Date field of the retrieved activity (incident/arrest) record.

(out) **Activity Date Year** – The year portion of the Activity Date field of the retrieved activity (incident/arrest) record.

(out) **Charge Description** – The Charge Description field of the retrieved activity (incident/arrest) record.

(out) **Ucr Code** – The Code Ucr field of the retrieved activity (incident/arrest) record.

(out) **Full Address** – The Full Address field of the retrieved address.

(out) **Street Number** – The Street Number field of the retrieved address.

(out) **Street Name** – The Street Name field of the retrieved address.

(out) **City** – The City field of the retrieved address.

(out) **State** – The State field of the retrieved address.

(out) **Zip Code** – The Zip Code field of the retrieved address.

(out) **Latitude** - The X Coordinate field of the retrieved address.

(out) **Longitude** - The Y Coordinate field of the retrieved address.

3.3.2.4.2 C3-F2-R2: Parameter Sizes

The size of the stored procedure parameters should be as follows:

Entity – nvarchar(255)

Vin – varchar(64)

Tag – varchar(64)

Entity ID – nvarchar(255)

Activity Type – varchar(64)

Activity Number – varchar(512)

Activity Date Month – varchar(2)

Activity Date Day – varchar(2)

Activity Date Year – varchar(4)

Charge Description – varchar(512)

Ucr Code – varchar(64)

Full Address – varchar(2048)

Street Number – varchar(256)

Street Name – varchar(256)

City – varchar(64)

State – varchar(64)

Zip Code – varchar(64)

Latitude - decimal

Longitude – decimal

3.3.2.4.3 C3-F2-R3: Naming Conventions

This stored procedure should be named sp_gisCopActivityLocation.

3.3.2.4.4 C3-F2-R4: Update Database Script

An update script must be created to add this stored procedure to an existing database.

3.3.2.4.5 C3-F2-R5: Create Database Script

This stored procedure must be added to this database's create script. This allows all new installations to have the latest database version without running any new update scripts.

3.4 C4: Activity Web Service

This component contains all modifications that are to be made to the Activity Web Service. This is a web service that is part of the current LEADR system.

3.4.1 C4-F1: Save Selected Person Location Message

3.4.1.1 Description

This message is used to save a person record's location, selected by the user, and should contain the following information. The information stored in this message will come from the appropriate fields of the State Search result page (from the user selected person location). This functionality only exists on the name_detail.jsp State Search result web page. The population of this message is handled in other feature definitions of this document.

- Last Name – The last name of the person associated with a selected location record.
- First Name – The first name of the person associated with a selected location record.
- Middle Name – The middle name of the person associated with a selected location record.
- Street Address – The entire street address (excluding city, state, and zip code) of a selected location record.
- City – The city of a selected location record.
- State – The state of a selected location record.
- Zip Code – The zip code of a selected location record.
- IP Address – The current user’s IP address.
- User ID – The current user’s user information (from LDAP).
- Deleted – This is a bit field set to either 1 or 0 (true or false). When set to 1, the specified selected person location is removed from the list of selected person location records.

3.4.1.2 Priority

This is a critical feature.

3.4.1.3 Functional Requirements

3.4.1.3.1 C4-F1-R1: XML Format

The message shall be designed in XML format and be able to support multiple person locations in a single message. The IP Address and User ID fields should only be contained once per message regardless of how many locations are contained within. Although an XML message without an XML declaration above the root element is considered mal-formed, it is important not to include the XML declaration in this XML message. This is a requirement of the LEADR system.

3.4.2 C4-F2: Save Selected Incident Location Message

3.4.2.1 Description

This message is used to save an activity (incident/arrest) record’s location, selected by the user, and should contain the following information. The information stored in this message will come from the appropriate fields of the State Search result page (from the user selected activity location). This functionality exists on the name_activity_query.jsp, vehicle_query_results.jsp, vehicle_activity_query.jsp, property_query_results.jsp, location_incident_activity_query.jsp, location_person_activity_query.jsp, narrative_query_results.jsp, and case_number_query_results.jsp State Search result web pages. The population of this message is handled in other feature definitions of this document.

- Entity ID – The unique identifier of a selected activity record.
- VIN – The vehicle identification number (VIN) of a vehicle associated with a selected activity record. The purpose of this field is that the vehicle_query_results.jsp web page does not contain the activity record’s unique identifier. This allows for later querying to find this identifier (during the Map Plot List web page loading process).

- TAG – The vehicle license plate number of a vehicle associated with a selected activity record. The purpose of this field is that the vehicle_query_results.jsp web page does not contain the activity record's unique identifier. This allows for later querying to find this identifier (during the Map Plot List web page loading process).
- IP Address – The current user's IP address.
- User ID – The current user's user information (from LDAP).
- Deleted – This is a bit field set to either 1 or 0 (true or false). When set to 1, the specified selected person location is removed from the list of selected person location records.

3.4.2.2 Priority

This is a critical feature.

3.4.2.3 Functional Requirements

3.4.2.3.1 C4-F2-R1: XML Format

The message should be designed in XML format and be able to support multiple activity Entity ID values as well as multiple VIN and TAG value combinations. The IP Address and User ID fields should only be contained once per message regardless of how many locations are contained within. Although an XML message without an XML declaration above the root element is considered mal-formed, it is important not to include the XML declaration in this XML message. This is a requirement of the LEADR system.

3.4.3 C4-F3: Load Selected Person Location Message

3.4.3.1 Description

This message is used to load all of the current user's selected person record's location records and should contain the following information:

- IP Address – The current user's IP address.
- User ID – The current user's user information (from LDAP).

3.4.3.2 Priority

This is a critical feature.

3.4.3.3 Functional Requirements

3.4.3.3.1 C4-F3-R1: XML Format

This message should be designed in XML format. Although an XML message without an XML declaration above the root element is considered mal-formed, it is important not to include the XML declaration in this XML message. This is a requirement of the LEADR system.

3.4.4 C4-F4: Load Selected Incident Location Message

3.4.4.1 Description

This message is used to load all of the current user's selected activity (incident/arrest) location records and should contain the following information:

- IP Address – The current user’s IP address.
- User ID – The current user’s user information (from LDAP).

3.4.4.2 Priority

This is a critical feature.

3.4.4.3 Functional Requirements

3.4.4.3.1 C4-F4-R1: XML Format

This message should be designed in XML format. Although an XML message without an XML declaration above the root element is considered mal-formed, it is important not to include the XML declaration in this XML message. This is a requirement of the LEADR system.

3.4.5 C4-F5: Incident Information GIS Cop Message Ingestion

3.4.5.1 Description

The purpose of this feature is to:

1. Accept an Incident Information GIS Cop Message (see C1-F3) submitted to the activity_detail.jsp web page via the POST method. The parameters for this submission should be populated as follows (populated in the Flex Viewer Application):
 - a. xml – This parameter should contain the XML message (see C1-F3) from the Flex Viewer Application.
 - b. action – This parameter should contain a developer specified number. This action parameter instructs the web page as to what type of action the user is requesting.
2. Decrypt and parse the User Identifier field from the XML message (see C1-F3) converting it to a valid User ID (the LDAP information usually used as the RequestedUserID field in most routes). This value is to be converted into the RequestedUserID XML message field (ActivityDetailRoute).
3. Parse the activity type value from the Entity ID XML message field (see C1-F3). This value is to be converted into the RequestedActivityType XML message (ActivityDetailRoute) field.
4. Consume the Activity Web Service’s runService web method passing in the following parameter values:
 - a. routeid – A static value of “ActivityDetailRoute”.
 - b. message - The converted XML message (ActivityDetailRoute). These conversions are mentioned in #2 and #3 above.
5. The results, from the web service call above, should then be handled the same way the existing activity_detail.jsp web page requests are currently handled (eventually redirecting the users web page to the activity_detail.jsp page).

3.4.5.2 Priority

This is a critical feature.

3.4.5.3 Stimulus/Response

Stimulus

A web page POST is made to the activity_detail.jsp page with the appropriate action parameter value (the action value for requesting activity record information from the Flex Viewer Application).

Response

The user's web browser is redirected to the activity_detail.jsp page displaying the activity (incident/arrest) record that the user had requested from the Flex Viewer Application.

3.4.5.4 Functional Requirements

3.4.5.4.1 C4-F5-R1: Performance/Security

For the purposes of performance and security, all source code changes for this feature, if possible, should be made in the ActivityHandler java class (code behind). It has been realized that there is a minimal amount of source code that must exist in the activity_detail.jsp web page itself to complete this feature.

3.4.5.4.2 C4-F5-R2: Decrypting/Parsing User Identifier Field

The User Identifier field of the Incident Information GIS Cop Message (XML Message, see C1-F3) must be decrypted and parsed using the source code created in section C1-F4.

3.4.5.4.3 C4-F5-R3 – XML Message Conversion

The Incident Information GIS Cop Message (see C1-F3) must be converted to the ActivityDetailRoute XML message format. Please note that it has been realized that since the ActivityType XML message field (ActivityDetailRoute) does not exist in the Incident Information GIS Cop Message (see C1-F3) as a separate field, some parsing is necessary. This allows for requirement C4-F5-R4.

3.4.5.4.4 C4-F5-R4 – Request Behavior

The "ActivityDetailRoute" route of the Activity Web Service must be consumed to fetch the results. The remainder of the steps taken, to eventually redirect the user to the activity_detail.jsp web page, should be the same as if the user was taken from a typical LEADR State Search result page to the activity_detail.jsp web page.

3.4.6 C4-F6: Save Selected Person Locations Route

3.4.6.1 Description

The goal of this route is to save a user's person location selection by:

1. Accepting a Save Selected Person Locations Message (see C4-F1) (XML).
2. Running the sp_saveGisCopPersonLocations stored procedure from the LEOS database (MySQL) (see C2-F2).

This route will be consumed from the name_detail.jsp web page when users check the added checkbox (selecting locations). These additional checkboxes and checked checkbox actions are handled in other features in this document.

3.4.6.2 Priority

This is a critical feature.

3.4.6.3 Stimulus/Response

Stimulus

The runService web method is run (Activity Web Service) passing in a routeid parameter value of "SaveSelectedPersonLocations" along with a Save Selected Person Location Message (see C4-F1) (XML message) as the message parameter value.

Response

The sp_saveGisCopPersonLocations stored procedure from the LEOS database (MySQL) (see C2-F2) is run effectively saving the user's selections.

3.4.6.4 Functional Requirements

3.4.6.4.1 C4-F6-R1: Web Service Usage

This route must be added to the existing runService web method of the Activity Web Service. The routeid parameter should be "SaveSelectedPersonLocations".

3.4.6.4.2 C4-F6-R2: Message Acceptance

This route should accept the Save Selected Person Location Message (see C4-F1). As used in the current Activity Web Service framework, a new xsl stylesheet should be developed to convert the XML message into the appropriate stored procedure call (see C4-F6-R3 for stored procedure information).

3.4.6.4.3 C4-F6-R3: Stored Procedure

The sp_saveGisCopPersonLocations stored procedure (LEOS Database (MySQL)) should be run upon message acceptance.

3.4.6.4.4 C4-F6-R4: Uniformity

This route should use the existing LEIE.Activity.ActivityRoute class if possible. It has been realized that this is not always possible. If it is not possible, any new class created should inherit from the LEIE.Activity.ActivityRoute class.

3.4.6.4.5 C4-F6-R5: Return Value

The response, from this route, should be

"<SaveSelectedPersonLocations>SUCCESS</SaveSelectedPersonLocation>" if an error did not occur.

Please note that the Activity Web Service has built-in error handling. If a run-time error does occur, the built-in functionality of the Activity Web Service should be used for error handling.

3.4.7 C4-F7: Save Selected Incident Locations Route

3.4.7.1 Description

The goal of this route is to save a user's activity location selection by:

1. Accepting a Save Selected Incident Locations Message (see C4-F2) (XML).
2. Running the sp_saveGisCopActivityLocations stored procedure from the LEOS database (MySQL) (see C2-F2).

This route will be consumed from the following State Search result web pages when the users check an added checkbox (selecting locations). These additional checkboxes and checked checkbox actions are handled in other features of this document:

- name_activity_query.jsp
- vehicle_query_results.jsp
- vehicle_activity_query.jsp
- property_query_results.jsp
- location_incident_activity_query.jsp
- location_person_activity_query.jsp
- narrative_query_results.jsp
- case_number_query_results.js

3.4.7.2 Priority

This is a critical feature.

3.4.7.3 Stimulus/Response

Stimulus

The runService web method is run (Activity Web Service) passing in a routeid parameter value of "SaveSelectedIncidentLocations" along with a Save Selected Incident Location Message (see C4-F2) (XML message) as the message parameter value.

Response

The sp_saveGisCoplIncidentLocations stored procedure from the LEOS database (MySQL) (see C2-F2) is run effectively saving the user's selections.

3.4.7.4 Functional Requirements

3.4.7.4.1 C4-F7-R1: Web Service Usage

This route must be added to the existing runService web method of the Activity Web Service. The routeid parameter should be "SaveSelectedIncidentLocations".

3.4.7.4.2 C4-F7-R2: Message Acceptance

This route should accept the Save Selected Incident Location Message (see C4-F2). As used in the current Activity Web Service framework, a new xsl stylesheet should be developed to convert the XML message into the appropriate stored procedure call (see C4-F7-R3 for stored procedure information).

3.4.7.4.3 C4-F7-R3: Stored Procedure

The sp_saveGisCoplIncidentLocations stored procedure (LEOS Database (MySQL)) should be run upon message acceptance.

3.4.7.4.4 C4-F7-R4: Uniformity

This route should use the existing LEIE.Activity.ActivityRoute class if possible. It has been realized that this is not always possible. If it is not possible, any new class created should inherit from the LEIE.Activity.ActivityRoute class.

3.4.7.4.5 C4-F7-R5: Return Value

The response, from this route, should be

“<SaveSelectedIncidentLocations>SUCCESS</SaveSelectedIncidentLocation>” if an error did not occur. Please note that the Activity Web Service has built-in error handling. If a run-time error does occur, the built-in functionality of the Activity Web Service should be used for error handling.

3.4.8 C4-F8: Load Selected Person Locations Route

3.4.8.1 Description

The goal of this route is to retrieve a list of a user’s person location selections by:

1. Accepting a Load Selected Person Locations Message (see C4-F3) (XML).
2. Running the sp_getGisCopPersonLocations stored procedure from the LEOS database (MySQL) (see C2-F4). This stored procedure retrieves the user’s person location selections.
3. Running the sp_gisCopPersonLocations stored procedure passing in the results from the sp_getGisCopPersonLocations stored procedure run in #2 above (see C3-F1). This stored procedure retrieves location records, based on the user’s person location selections, from the LEADR database.
4. The results from the stored procedure run, in #3 above, should be converted into an XML message then passed on as the response to the consumer of this web request.

This route will be consumed from the Map Plot List screen (LEADR system). Please note that this is the same message also used in submitting via the POST method to the Flex Viewer Application.

3.4.8.2 Priority

This is a critical feature.

3.4.8.3 Stimulus/Response

Stimulus

The runService web method is consumed (Activity Web Service) passing in a routeid parameter value of “LoadSelectedPersonLocations” along with a Load Selected Person Location Message (see C4-F3) (XML) as the message parameter value.

Response

The sp_getGisCopPersonLocations (LEOS database (MySQL)) (see C2-F4) and sp_gisCopPersonLocation (LEADR database (MS SQL Server)) (see C3-F1) stored procedures are called sequentially to retrieve the location records, from the LEADR database, that the current user has selected and intends to plot on a map (Flex Viewer Application). The response of the sp_gisCopPersonLocation stored procedure is then transformed into an XML message and sent as the response to the consumer of this web request.

3.4.8.4 Functional Requirements

3.4.8.4.1 C4-F8-R1: Web Service Usage

This route must be added to the existing runService web method of the Activity Web Service. The routeid parameter should be “LoadSelectedPersonLocations”.

3.4.8.4.2 C4-F8-R2: Message Acceptance

This route should accept the Load Selected Person Location Message (see C4-F3). As used in the current Activity Web Service framework, new XSL stylesheets should be developed to convert the XML message and stored procedure results into the appropriate stored procedure calls (see C4-F8-R3 and C4-F8-R4 for stored procedure information).

3.4.8.4.3 C4-F8-R3: Retrieving Saved Person Location Selections (Stored Procedure)

The `sp_loadGisCOPPersonLocations` stored procedure (LEOS Database (MySQL)) should be run upon message acceptance.

3.4.8.4.4 C4-F8-R4: Retrieving Location Records (Stored Procedure)

The `sp_gisCOPPersonLocation` stored procedure should be run by passing the results of C4-F8-R3 above into the input parameters of this stored procedure (see C3-F1)

3.4.8.4.5 C4-F8-R5: Uniformity

This route should use the existing `LEIE.Activity.ActivityRoute` class if possible. It has been realized that this is not always possible. If it is not possible, any new class created should inherit from the `LEIE.Activity.ActivityRoute` class.

3.4.8.4.6 C4-F8-R6: Return Value

The response, from this route, should be a Person Location GIS COP Message (see C1-F1). As used in the current Activity Web Service framework, a new XSL stylesheet should be developed to convert the `sp_gisCOPPersonLocation` stored procedure results into the proper response message format.

3.4.9 C4-F9: Load Selected Incident Locations Route

3.4.9.1 Description

The goal of this route is to retrieve a list of a user's activity location selections by:

1. Accepting a Load Selected Incident Locations Message (see C4-F4) (XML).
2. Running the `sp_getGisCOPIncidentLocations` stored procedure from the LEOS database (MySQL) (see C2-F5). This stored procedure retrieves the user's person location selections.
3. Running the `sp_gisCOPIncidentLocations` stored procedure passing in the results from the `sp_getGisCOPIncidentLocations` stored procedure run in #2 above (see C3-F2). This stored procedure retrieves location records, based on the user's person location selections, from the LEADR database.
4. The results from the stored procedure run, in #3 above, should be converted into an XML message then passed on as the response to the consumer of this web request.

This route will be consumed from the Map Plot List screen (LEADR system). Please note that this is the same message also used in submitting via the POST method to the Flex Viewer Application.

3.4.9.2 Priority

This is a critical feature.

3.4.9.3 Stimulus/Response

Stimulus

The runService web method is consumed (Activity Web Service) passing in a routeid parameter value of "LoadSelectedIncidentLocations" along with a Load Selected Incident Location Message (see C4-F4) (XML) as the message parameter value.

Response

The sp_getGisCoplIncidentLocations (LEOS database (MySQL)) (see C2-F5) and sp_gisCoplIncidentLocation (LEADR database (MS SQL Server)) (see C3-F1) stored procedures are called sequentially to retrieve the location records, from the LEADR database, that the current user has selected and intends to plot on a map (Flex Viewer Application). The response of the sp_gisCoplIncidentLocation stored procedure is then transformed into an XML message and sent as the response to the consumer of this web request.

3.4.9.4 Functional Requirements

3.4.9.4.1 C4-F9-R1: Web Service Usage

This route must be added to the existing runService web method of the Activity Web Service. The routeid parameter should be "LoadSelectedIncidentLocations".

3.4.9.4.2 C4-F9-R2: Message Acceptance

This route should accept the Load Selected Incident Location Message (see C4-F4). As used in the current Activity Web Service framework, new XSL stylesheets should be developed to convert the XML message and stored procedure results into the appropriate stored procedure calls (see C4-F9-R3 and C4-F9-R4 for stored procedure information).

3.4.9.4.3 C4-F9-R3: Retrieving Saved Incident Location Selections (Stored Procedure)

The sp_loadGisCoplIncidentLocations stored procedure (LEOS Database (MySQL)) should be run upon message acceptance.

3.4.9.4.4 C4-F9-R4: Retrieving Location Records (Stored Procedure)

The sp_gisCoplPersonLocation stored procedure should be run by passing the results of C4-F9-R3 above into the input parameters of this stored procedure (see C3-F2).

3.4.9.4.5 C4-F9-R5: Uniformity

This route should use the existing LEIE.Activity.ActivityRoute class if possible. It has been realized that this is not always possible. If it is not possible, any new class created should inherit from the LEIE.Activity.ActivityRoute class.

3.4.9.4.6 C4-F9-R6: Return Value

The response, from this route, should be an Incident Location GIS COP Message (see C1-F2). As used in the current Activity Web Service framework, a new XSL stylesheet should be developed to convert the sp_gisCoplIncidentLocation stored procedure results into the proper response message format.

3.5 C5: LEADR User Interface

This component consists of all modifications that shall be made to the user interface component of the LEADR system.

3.5.1 C5-F1: Name Details Result Page Modifications

3.5.1.1 Description

The name_details.jsp web page contains details about a person. The Last Known Address section of this page lists all known locations where a person resides. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by adding a new column next to the addresses listed containing a checkbox.

3.5.1.2 Priority

This is a high priority feature.

3.5.1.3 Stimulus/Response

Stimulus

The user checks a checkbox.

Response

An asynchronous call is made to a web page that saves the checked person location to the Map Plot List Helper Table (see C2-F1).

Stimulus

The user unchecks a checkbox.

Response

An asynchronous call is made to a web page that removes the unchecked person location from the Map Plot List Helper Table (see C2-F1).

Stimulus

The user navigates away from the web page.

Response

Any saving (adding/removing) involved with the Map Plot List Helper Table (see C2-F1) should continue processing until complete.

3.5.1.4 Functional Requirements

3.5.1.4.1 C5-F1-R1: Checkbox

A checkbox should be added as the content of a new column to the left of the list of addresses in the Last Known Address section of this web page. One checkbox shall exist per row.

3.5.1.4.2 C5-F1-R1: Functionality

See Stimulus/Response above.

3.5.1.4.3 C5-F1-R2: AJAX/JQuery

The asynchronous calls, specified in the Stimulus/Response section above, should use AJAX via JQuery to accomplish this. The GET method should be used in all asynchronous calls.

3.5.1.4.4 C5-F1-R3: Look and Feel

All modifications made to this web page must have a similar look and feel when compared to the current LEADR user interfaces.

3.5.1.4.5 C5-F1-R4: Saving

Items are added and removed to and from the Map Plot List Helper Table (see C2-F1) using a call to the Save Selected Person Locations Route (Activity Web Service) (see C4-F6). All errors encountered, during this save operation, are to be reported to the user, if possible.

3.5.1.4.6 C5-F1-R5: Uniformity

All source code used for the following should be contained within the existing ActivityHandler java class (LEADR user interface):

- Building XML messages,
- Calling the Activity Web Service

3.5.1.4.7 C5-F1-R6: Backwards Compatibility

To maintain backward compatibility, all checkboxes added should only be visible if the “giscop.enabled” user interface configuration setting is set to true. This configuration setting is located in the {Apache Tomcat Application Folder}/webapps/leadr/WEB-INF/classes/ui.properties file.

3.5.2 C5-F2: Name Activity Result Page Modifications

3.5.2.1 Description

The name_activity_query.jsp web page contains a listing of activity (incident/arrest) records related to a person. These activity records contain a location where the activity took place. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by inserting a new column entitled “Map” as the first column of the result listing, on this page, containing a checkbox.

3.5.2.2 Priority

This is a high priority feature.

3.5.2.3 Stimulus/Response

Stimulus

The user checks a checkbox.

Response

An asynchronous call is made to a web page that saves the checked activity location to the Map Plot List Helper Table (see C2-F1).

Stimulus

The user unchecks a checkbox.

Response

An asynchronous call is made to a web page that removes the unchecked activity location from the Map Plot List Helper Table (see C2-F1).

Stimulus

The user checks the check all checkbox.

Response

All remaining checkboxes, that are currently unchecked on the current page, are checked. An asynchronous call is made to a web page, for each newly checked checkbox, which then saves the checked activity locations to the Map Plot List Helper Table (see C2-F1).

Stimulus

The user unchecks the check all checkbox.

Response

All remaining checkboxes, which are currently checked on the current page, are unchecked. An asynchronous call is made to a web page, for each newly unchecked checkbox, which then removes the unchecked activity locations from the Map Plot List Helper Table (see C2-F1).

Stimulus

The user navigates away from the web page.

Response

Any saving (adding/removing) involved with the Map Plot List Helper Table (see C2-F1) should continue processing until complete.

3.5.2.4 Functional Requirements

3.5.2.4.1 C5-F2-R1: Checkbox

A checkbox should be added as the content of a new column to the left of the list of addresses in the Last Known Address section of this web page. One checkbox shall exist per row.

3.5.2.4.2 C5-F2-R2: Checkall Checkbox

A new row should be added to the web page results. The only contents of this row should be a checkbox in the first column. This is the check all checkbox and is used by the user to check or uncheck all items in the current result list.

3.5.2.4.3 C5-F2-R3: Functionality

See Stimulus/Response above.

3.5.2.4.4 C5-F2-R4: AJAX/JQuery

The asynchronous calls, specified in the Stimulus/Response section above, should use AJAX via JQuery to accomplish this. The GET method should be used in all asynchronous calls.

3.5.2.4.5 C5-F2-R5: Look and Feel

All modifications made to this web page must have a similar look and feel when compared to the current LEADR user interfaces.

3.5.2.4.6 C5-F2-R6: Saving

Items are added and removed to and from the Map Plot List Helper Table (see C2-F1) using a call to the Save Selected Incident Locations Route (Activity Web Service) (see C4-F7). All errors encountered, during this save operation, are to be reported to the user, if possible.

3.5.2.4.7 C5-F2-R7: Uniformity

All source code used for the following should be contained within the existing ActivityHandler java class (LEADR user interface):

- Building XML messages,
- Calling the Activity Web Service

3.5.2.4.8 C5-F2-R8: Backwards Compatibility

To maintain backward compatibility, all checkboxes added should only be visible if the “giscop.enabled” user interface configuration setting is set to true. This configuration setting is located in the {Apache Tomcat Application Folder}/webapps/leadr/WEB-INF/classes/ui.properties file.

3.5.3 C5-F3: Vehicle Result Page Modifications

3.5.3.1 Description

The vehicle_query_results.jsp web page contains a listing of vehicle records. These vehicle records are linked to activity (incident/arrest) records that contain a location where the activity took place. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by inserting a new column entitled “Map” as the first column of the result listing on this page containing a checkbox.

3.5.3.2 Priority

This is a high priority feature.

3.5.3.3 Stimulus/Response

See section 3.5.2.3 Stimulus/Response.

3.5.3.4 Functional Requirements

See section 3.5.2.4 Functional Requirements.

3.5.4 C5-F4: Vehicle Activity Result Page Modifications

3.5.4.1 Description

The vehicle_activity_query.jsp web page contains a listing of activity (incident/arrest) records related to a vehicle. These activity records contain a location where the activity took place. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by inserting a new column entitled “Map” as the first column of the result listing on this page containing a checkbox.

3.5.4.2 Priority

This is a high priority feature.

3.5.4.3 Stimulus/Response

See section 3.5.2.3 Stimulus/Response.

3.5.4.4 Functional Requirements

See section 3.5.2.4 Functional Requirements.

3.5.5 C5-F5: Property Result Page Modifications

3.5.5.1 Description

The property_query_results.jsp web page contains a listing of property records. These property records are linked to activity (incident/arrest) records that contain a location where the activity took place. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by inserting a new column entitled “Map” as the first column of the result listing on this page containing a checkbox.

3.5.5.2 Priority

This is a high priority feature.

3.5.5.3 Stimulus/Response

See section 3.5.2.3 Stimulus/Response.

3.5.5.4 Functional Requirements

See section 3.5.2.4 Functional Requirements.

3.5.6 C5-F6: Location Incident Activity Result Page Modifications

3.5.6.1 Description

The location_incident_activity_query.jsp web page contains a listing of activity (incident/arrest) records related to a location. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by inserting a new column entitled “Map” as the first column of the result listing on this page containing a checkbox.

3.5.6.2 Priority

This is a high priority feature.

3.5.6.3 Stimulus/Response

See section 3.5.2.3 Stimulus/Response.

3.5.6.4 Functional Requirements

See section 3.5.2.4 Functional Requirements.

3.5.7 C5-F7: Location Person Activity Result Page Modifications

3.5.7.1 Description

The location_person_activity_query.jsp web page contains a listing of activity (incident/arrest) records related to a location. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by inserting a new column entitled “Map” as the first column of the result listing on this page containing a checkbox.

3.5.7.2 Priority

This is a high priority feature.

3.5.7.3 Stimulus/Response

See section 3.5.2.3 Stimulus/Response.

3.5.7.4 Functional Requirements

See section 3.5.2.4 Functional Requirements.

3.5.8 C5-F8: Narrative Result Page Modifications

3.5.8.1 Description

The narrative_query_results.jsp web page contains a listing of activity (incident/arrest) records and their narratives. These activity records contain a location where the activity took place. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by inserting a new column entitled “Map” as the first column of the result listing on this page containing a checkbox.

3.5.8.2 Priority

This is a high priority feature.

3.5.8.3 Stimulus/Response

See section 3.5.2.3 Stimulus/Response.

3.5.8.4 Functional Requirements

See section 3.5.2.4 Functional Requirements.

3.5.9 C5-F9: Case Number Result Page Modifications

3.5.9.1 Description

The case_number_query_results.jsp web page contains a listing of activity (incident/arrest) records and their case numbers (incident numbers). These activity records contain a location where the activity took place. This modification will allow the user to select these locations for plotting them on a map (Flex Viewer Application) at a later time. This is achieved by inserting a new column entitled “Map” as the first column of the result listing on this page containing a checkbox.

3.5.9.2 Priority

This is a high priority feature.

3.5.9.3 Stimulus/Response

See section 3.5.2.3 Stimulus/Response.

3.5.9.4 Functional Requirements

See section 3.5.2.4 Functional Requirements.

3.5.10 C5-F10: Map Plot List Screen

3.5.10.1 Description

The purpose of this screen is to list all location records previously selected by the user for plotting on a map at a later time (stored in the Map Plot List Helper Table (see C2-F1)). The user can reset this list (clearing all selected locations) by clicking the Clear All button. In essence, the results listed are the current selected locations for the current user. This list contains a checkbox, in the leftmost column, that defaults to checked when the screen is first loaded. The first row of the result list should not contain any content except for a checkbox in the left most column. This is the check all checkbox and should also default to checked when the screen is first loaded. When the user clicks the Plot button, all selected location records on this screen are sent, via the POST method, to a SLED supplied url. This url should open the Flex Viewer Application.

3.5.10.2 Priority

This is a critical feature.

3.5.10.3 Stimulus/Response

Stimulus

The page is loaded.

Response

The Load Selected Incident Locations Route (see C4-F9) and Load Selected Person Locations Route (see C4-F8) Activity Web Service routes are run. This resulting XML message is then parsed and populated on the screen in the appropriate columns (see C5-F10-R3).

Stimulus

The user checks a checkbox.

Response

The checkbox becomes checked. This means that when the user clicks the Plot button, this location record is included in the POST message sent to the Flex Viewer Application.

Stimulus

The user unchecks a checkbox.

Response

The checkbox becomes unchecked. This means that when the user clicks the Plot button, this location record is not included in the POST message sent to the Flex Viewer Application.

Stimulus

The user checks the check all checkbox.

Response

All remaining checkboxes, that are currently unchecked on the current page, are checked.

Stimulus

The user unchecks the check all checkbox.

Response

All remaining checkboxes, which are currently checked on the current page, are unchecked.

Stimulus

The Clear All button is clicked.

Response

The Save Selected Incident Locations Route (see C4-F7) of the Activity Web Service should be run with all Save Selected Incident Location Message (see C4-F2) fields blank except for IP Address, User ID, and Deleted. This should delete all location records selected for the current user. This screen should then be navigated away from (perhaps to the announcements page or Name & Number Search web page).

Stimulus

The Plot button is clicked.

Response

The Load Selected Incident Locations Route (see C4-F9) and Load Selected Person Locations Route (see C4-F8) Activity Web Service routes are re-run (ran when the screen was loaded) filtering out the results that were left unchecked on this screen. This resulting XML message is then submitted to a SLED supplied url via the POST method. The action expected from the Flex Viewer Application is that the XML location data is then plotted on a map.

3.5.10.4 Functional Requirements

3.5.10.4.1 C5-F10-R1: Overall Screen Design

The screen layout should be similar to the name_activity_query.jsp web page containing the columns Map, Case, Person, Date, Type, Address, Latitude, and Longitude.

3.5.10.4.2 C5-F10-R2: Checkboxes

The first column should contain a checkbox. The first row should not contain any data except for a checkbox in the first column (Map column).

3.5.10.4.3 C5-F10-R3: Column Field Population

The columns should be populated as follows (the XML field values supplied are from the Incident Location GIS COP Message (see C1-F2) and the Person Location GIS COP Message (see C1-F1):

- Case – Activity Number XML field.
- Person – The Last Name, First Name, and Middle Name XML fields concatenated together in the format: {Last Name}, {First Name} {Middle Name}
- Date – The Activity Date Month, Activity Date Day and Activity Date Year XML fields concatenated together in the format: yyyy-MM-dd
- Type – The Activity Type XML field.
- Address – If the Full Address XML field is the only address related XML field populated, then it is used in this fields population; otherwise, the Street Number, Street Name, City, State, and Zip Code XML fields are concatenated in the format: {Street Number} {Street Name}, {City}, {State} {Zip Code}
- Latitude – The Latitude XML field.
- Longitude – The Longitude XML field.

3.5.10.4.4 C5-F10-R4: Buttons

There should be two buttons on the lower right bottom of the listed location records. They should be labeled "Plot" and "Clear All" respectively. (Plot button on the left)

3.5.10.4.5 C5-F10-R5: Functionality

See Stimulus/Response above.

3.5.10.4.6 C5-F10-R6: AJAX/JQuery

AJAX and JQuery should be used if it will enhance the performance of sending XML messages to the Flex Viewer Application via the POST method.

3.5.10.4.7 C5-F10-R6: Uniformity

All source code used for the following should be contained within the existing ActivityHandler java class (LEADR user interface):

- Building XML messages,
- Calling the Activity Web Service

4 External Interface Requirements

4.1 User Interfaces

The successful deployment of this enhancement is dependent upon the appropriate modifications made to the State Search result web pages and development of a new Map Plot List web page. While these are all LEADR system modifications, it is also dependent on the proper source code modifications made to the Flex Viewer Application allowing for the plotting (on a map) of activity and person location records as well as the user to load a LEADR activity (incident/arrest) record via an action (i.e. button) in the Flex Viewer Application.

The Flex Viewer Application is external to the current LEADR system. All modifications made to the Flex Viewer Application are being performed by SLED.

4.2 Hardware Interfaces

There are no additional hardware requirements for this enhancement.

4.3 Software Interfaces

With the addition of this software enhancement, the existing LEADR system will contain the ability to interface with the Flex Viewer Application accepting XML messages to and from it.

4.4 Communications Interfaces

4.4.1 Web Services

This enhancement will utilize the existing Activity web service to internally save and load user saved location data. This web service utilizes SOAP and is implemented by Microsoft .NET Framework 2.0.

4.4.2 Internal Web Page Operation

All new checkboxes added as part of this enhancement, will utilize AJAX via JQuery to add and remove user-selected locations.

4.4.3 External Web Page Operation

All XML messages sent to and from the Flex Viewer Application will send the data via the POST method.

5 Nonfunctional Requirements

5.1 Open-Source

This solution will be open-source and as such, all J2 library code that is used must be copied and pasted into a new library for use exclusively with this project.