A Physical Design
for the
State of Iowa
Department of Transportation’s
Linear Referencing System
PHYSICAL DESIGN TECHNICAL DOCUMENT

July 2000

Prepared by

GeoAnalytics™

1716 Fordem Avenue
Madison, WI 53704
Phone: 608-241-7100
Facsimile: 608-241-7116
E-mail: office@geoanalytics.com
URL: **http://www.geoanalytics.com**

TRANSDECISIONS

11835 W. Olympic Blvd.
Suite 700E
Los Angeles CA 90064
Phone: (865) 988-8345
Facsimile: (865) 988-8744
E-mail: **JulianRay@TransDecisions.com**

With

Oracle Corporation

222 West Las Colinas Blvd.
Consulting
Suite 1000
Irving, TX 75039
Phone: (972) 409-3907
Facsimile: (972) 409-3041
E-Mail: jlwinter@us.oracle.com

Iowa Department of Transportation

800 Lincoln Way
Ames, IA 50010
Phone: 515-233-7770
Facsimile: (515) 239-1828
E-mail: wschuma@max.state.ia.us

Table of Contents

List of Figures

List of Tables

Physical Design for the Iowa DOT Linear Referencing System Technical Document

1 OVERVIEW

The purpose of this document is to provide the detailed specifications of the Physical Design for the Iowa DOT Linear Referencing System (LRS). Specifications are how the LRS requirements will be met using Iowa DOT's technology and organizational environments. This technical document describes the technology, process, and organizational specifications of the LRS. The data specifications can be found in the Physical Database Model Technical Reference Document.

A physical design summary document also exists. Although still technical, the summary document attempts to explain the design for less technical business staff and managers. It contains an overview of the LRS technology architecture and system specifications by LRS subsystem. For insight into physical design results, please see the Decisions Behind the Physical Design Technical Reference Document.

Section 2 of this document describes the LRS technology architecture required to support the LRS. The technology architecture addresses both LRS maintenance and LRS decision support (GeoData Warehouse). This section includes an overview of the three-tier application framework, interoperability approach, the operational architecture, an inventory of software requirements, and an estimated level of effort to develop the system.

Section 3 provides GeoData Warehouse (GDW) data and process specifications. This section includes the interface specifications for the three key GDW user tools: LRS reference calculator, Linear Transform, and Linear Overlay. These specifications are followed by description of GDW metadata handling. Data staging specifications for both LRS and business data is provided next, focusing on LRS data formats in the GDW.

Business Logic is described in Section 4. Business logic is the nucleus of the system and is used by both the LRS maintenance and GDW architectures of the LRS. This section provides an overview of the algorithms for the transformation and staging processes. The actual algorithms are in the Appendices. The section also includes an overview of business logic management.

The next section, Section 5, details the LRS maintenance process specifications. This section provides an overview of user interface specifications, how transaction management can be implemented, and an overview of maintenance event propagation. Actual event specifications are provided in the appendices. Finally, this section covers the workflow specifications for specific LRS components.

The last section describes how the LRS Subsystems and their related components are assigned to DOT organizational units. This section assigns both subsystem management and specific workflows within these subsystems. The section also includes descriptions on why specific organizational units were chosen, how the assignment will change what the units do, and what technology and skills are needed in each unit.

It is important to note that this technical document is a dynamic document, with potential changes, added details, and comments added throughout the remaining project phases. Therefore, it is critical for the reader to possess the most current document version.

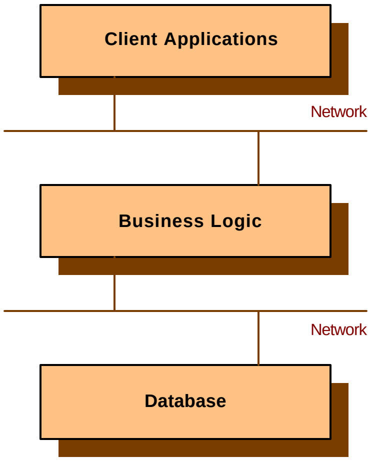
2 TECHNOLOGY SPECIFICATIONS

This section contains a description of the technology specifications for the LRS.

2.1 Application Framework

The LRS is designed as a multi-tier distributed application. Figure 1 illustrates the overall application framework for the LRS.

Figure 1 - 3-Tier Application Framework



The system is designed as a distributed three-tier system composed of clients, business logic, and data stores, all of which are logically connected via a network. The business logic, also called the middle tier, contains services and facilities that implement the linear transformation, linear overlay, and staging operations that are part of the LRS application.

2.1.1 Client Tier

The client tier is composed of GeoMedia clients and other applications that can communicate with the middle and database tiers. Examples of non-GeoMedia clients that can access the other layers are Oracle Forms, SQL-Plus and other command line interfaces, and interoperable systems that are specifically designed to access the LRS functionality across the network.

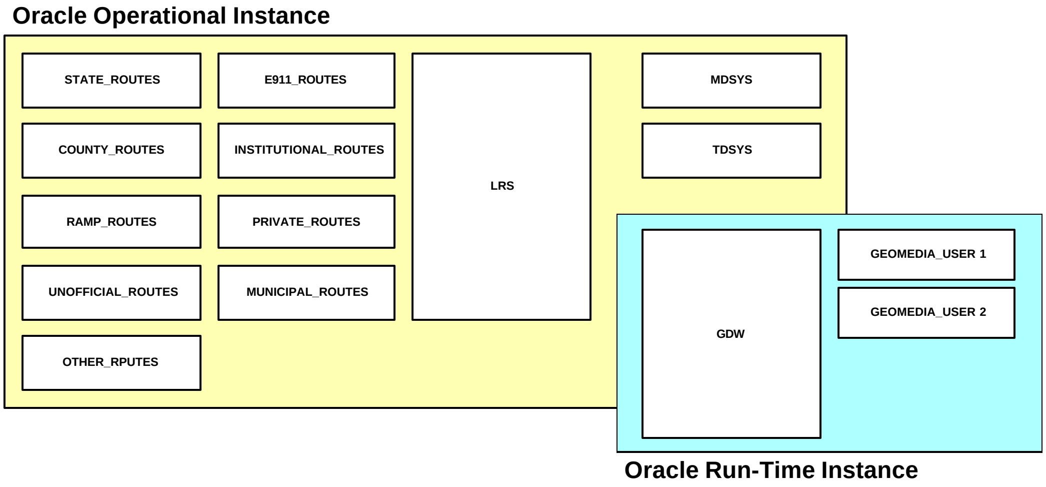
2.1.2 Business Logic Tier

The Business Logic tier is composed of a set of Java classes and stored procedures that implement the major functions of the LRS. These functions include the code that performs linear transformations, linear overlay, and staging functions.

2.1.3 Database Tier

The Database tier consists of two Oracle schemas. One schema contains the GDW while the other schema contains the LRS. The GDW schema and the LRS schema are logically separated and can be easily physically separated into different database instances if required. This organization is illustrated in Figure 2.

Figure 2 - Conceptual LRS and GDW Schema Organization

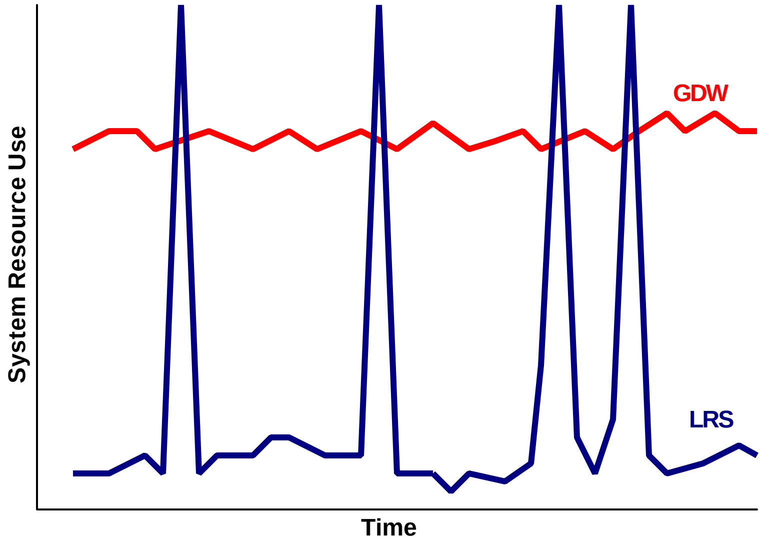


Maintaining the LRS and GDW schema in separate database instances has the advantage of allowing the run-time and maintenance environments to be separated, and therefore avoid conflicts for system resources. Using this approach, LRS maintenance operations will not affect the performance of run-time users of the system and visa-versa. During normal operation it is expected that the system resources required by the GDW server will be many times greater than the system resources required for the maintenance environment.

The graph in Figure 3 illustrates an expected system resource use model for the two environments. The GDW requirements are characterized by a relatively high constant demand for system resources. This demand model is typical for systems with large numbers of users performing typical database functions. As more GDW users are added to the system, the curve will tend to reduce its amplitude and flatten providing that system resources are matched to the demand. The LRS curve, on the other hand, is typical of a system with a small number of users who periodically perform intensive processing operations. This is the expected characteristic of the maintenance environment where a few users will be performing maintenance operations and staging data in bulk to the GDW. Unlike the GDW curve, adding more maintenance users to the LRS will not tend to normalize the system resource demand as the demand is characterized by low level use interspersed with short-duration high-resource demand.

Physically separating the two database instances will allow the GDW environment to scale at a different rate from the maintenance environment and protect the GDW users from the processing delays associated with the peak loads of the LRS users.

Figure 3 - Illustrative Resource Usage for GDW and LRS Schemas During a 15 Minute Period



2.2 Interoperability

There are two notions of interoperability that pertain to the use and maintenance of the LRS. Within the LRS itself, the subsystems are interoperable and within the Iowa DOT enterprise, applications can interoperate with the LRS.

2.2.1 LRS Subsystem Interoperability

The individual LRS management subsystems interoperate to form the LRS. Each management subsystem can be independently managed and maintained as a set of asynchronous processes. This idea is illustrated in Figure 4.

Changes generated in one of the subsystems can affect other subsystems. These effects are managed as events that can be trapped and used by other subsystems. As these subsystems all exist within the LRS schema, database triggers and referential integrity constraints can be used to provide the firing mechanisms for the triggers.

2.2.2 LRS Application Interoperability

Eventually the LRS will become the focus of several client systems that require the transformation, overlay, and staging services of the LRS. These systems can be developed to interoperate with the LRS business logic tier through the network. This concept is illustrated in Figure 5.

External applications interoperate by passing LRM data over the network. The GeoMedia custom commands detailed in this document are the first interoperable components that will be developed to run against the business logic.

Figure 4 - LRS Subsystem Interoperability

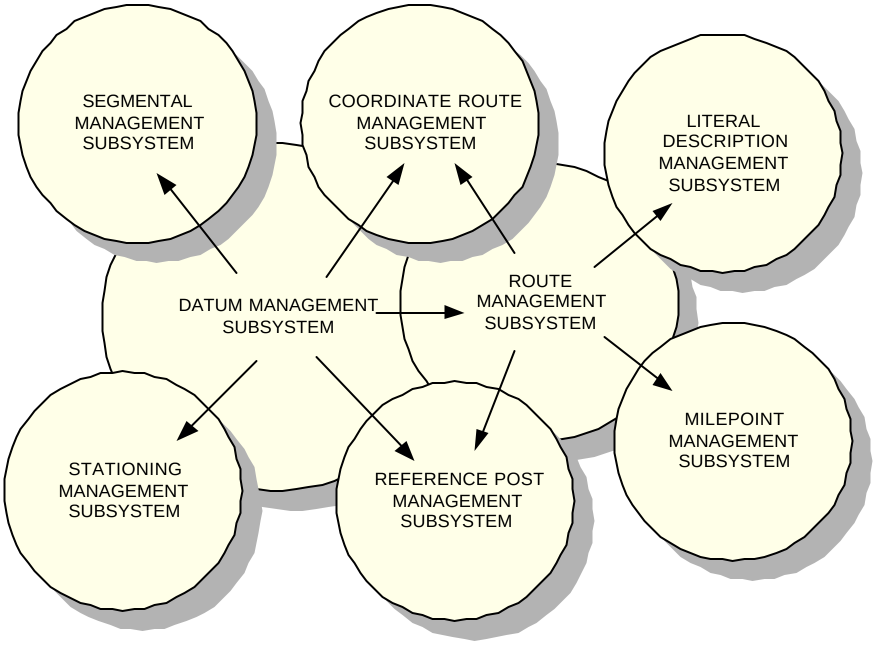
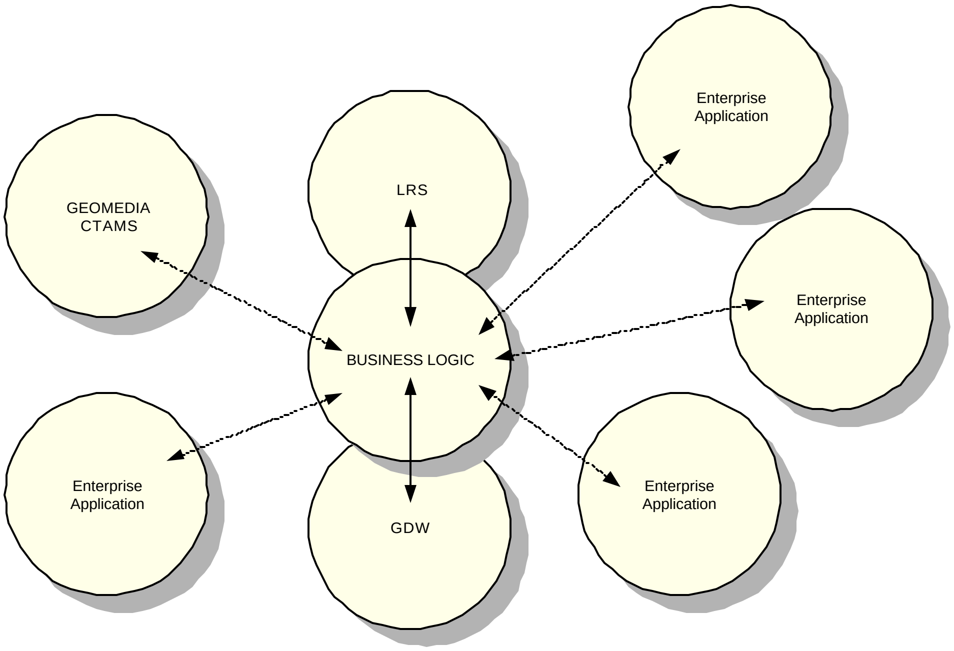

SEGMENTAL MANAGEMENT SUBSYSTEM
COORDINATE ROUTE MANAGEMENT SUBSYSTEM
LITERAL DESCRIPTION MANAGEMENT SUBSYSTEM
DATUM MANAGEMENT SUBSYSTEM
ROUTE MANAGEMENT SUBSYSTEM
STATIONING MANAGEMENT SUBSYSTEM
REFERENCE POST MANAGEMENT SUBSYSTEM
MILEPOINT MANAGEMENT SUBSYSTEM

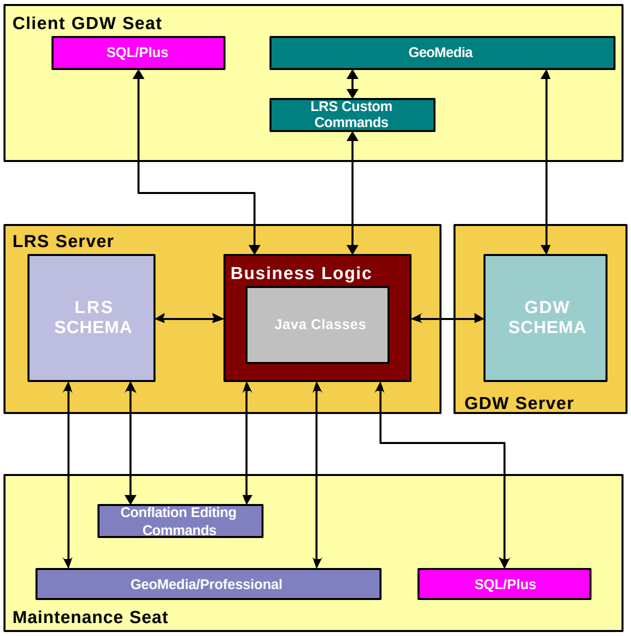
Figure 5 - LRS Application Interoperability



2.3 Operational Architecture

The operational architecture of the system is illustrated in Figure 6.

Figure 6 - Design for the LRS Application Architecture



2.3.1 Clients

The clients have access, through GeoMedia, to the data that has been staged in the GDW in read-only mode. Via the custom commands, they also have the ability to execute stored procedures that perform linear transform and overlay operations. Additionally, access to both the staged data and stored procedures is provided by other non-GIS database interfaces such as SQLPlus.

The GeoMedia clients will still be able to integrate data from other parts of the GDW such as Arc/Info, MGE, and GeoMedia staged data into views and analyses in the normal manner.

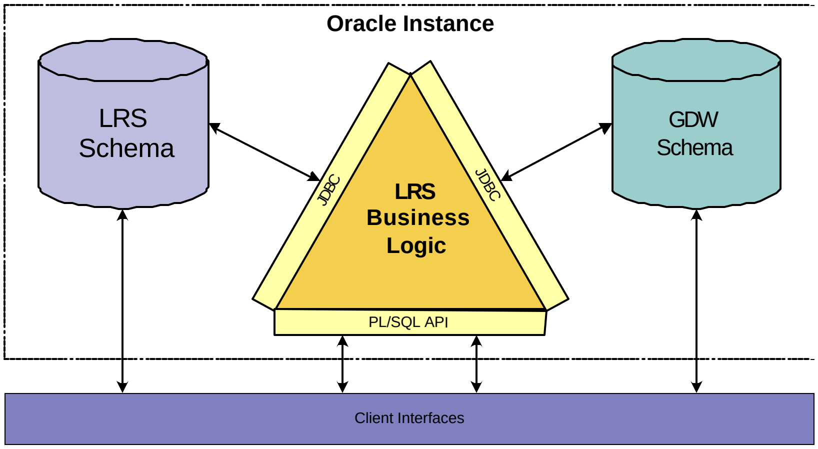
2.3.2 Maintenance

For maintenance operations, GeoMedia Pro provides the ability to manage and maintain the LRS schema and initiate stored procedures as necessary to perform workflows. GeoMedia Pro uses a read-write version of the Oracle GDO server to update the LRS components via its map and table interfaces. Staging of the data from the maintenance schema to the staged schema will be performed by stored procedures. These staging procedures can be initiated via custom commands in GeoMedia Pro or from a non-GIS environment such as SQL/Plus.

2.3.3 Business Logic

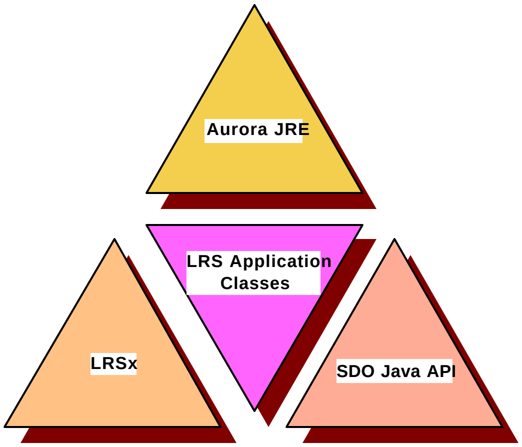
For the pilot project, the LRS business logic component is designed and written as a three-tier system but is implemented in two physical tiers. The LRS services and facilities are encapsulated as a set of Java classes that are wrapped with PL/SQL interfaces and interact with the data stores via a thin Java JDBC (Java Database Connectivity) client. Within the Oracle JServer environment, the JDBC client is implemented efficiently to allow the Java classes to be embedded within the database instance using the same code and logic as if they existed on an application server on the network. Figure 7 illustrates this approach.

Figure 7 - Organization of the Oracle Instance



The primary interface to the business logic is through a PL/SQL interface which wraps the Java coded business logic as a set of stored procedures. The stored procedures can be executed from any compliant environment including, but not limited to, SQL/Plus, Custom Applications, and Web-enabled environments (WebDB, OAS).

Figure 8 - Organization of the LRS Business Logic Components



The LRS services and facilities that comprise the core of the business logic are composed of four sets of Java classes illustrated in Figure 8. The LRS Application classes are those that are developed specifically for the Iowa DOT project and consist of the linear transform and data staging operations along with other wrapper classes and classes designed to manage components of the system. The LRS specific classes depend on the LRSx class library from TransDecisions, Oracle's Spatial Data Objects Java classes, and the Java run-time classes provided by the JServer environment in the Oracle instance. Table 1 defines run-time environments and vendors that are necessary to implement this system.

Table 1 - Business Logic Components By Vendor

Technology	**Capabilities**	**Vendor**	**Description**
LRS Stored Procedure Wrappers	Transformation interfaces, linear overlay interfaces.	Developed as part of LRS Project.	Stored procedure wrappers including polymorphic transformation and overlay interfaces.
LRS Specific Java Classes	Transformation algorithms, staging procedures	Developed as part of LRS Project.	LRS application specific classes.
LRSx	Conflation tools, dynamic segmentation tools, linear overlay tools.	TransDecisions Inc.	Conflation management tools. Dynamic Segmentation tools. Metadata management.
SDO Java API	Geometry storage and manipulation.	Oracle Corp.	OGIS compliant presentation and management of geometric features.
Aurora Java Run-Time Classes		Oracle Corp.	Java run-time environment.

2.4 Software Inventory

Each client machine should have, at a minimum, the following software components to be able to connect to the GDW and perform linear transformations using the LRS.

2.4.1 Client Workstation

This software inventory is specific to the client seats. Client seats are not necessarily GIS clients, but may run other database connectivity software such as SQL / Plus.

Note that GeoMedia is only required for clients that need to create maps or use the GeoMedia Custom Commands developed by Intergraph's project team. SQL-Plus is installed as part of the Oracle client installation and can be used as an alternate interface for running stored procedures and generating reports.

Table 2 - Client System Software Inventory

Software	Required	Vendor	Notes
Microsoft Win98 R2, NT 40 SP5 or Windows 2000	Yes	Microsoft	Must be compatible with GeoMedia 4.0. It is recommended that Windows NT or 2000 be used in preference to Win98.
Net8 Connectivity	Yes	Oracle	Compatible with LRS server. Version 8.1.6 is the targeted platform.
GeoMedia 4.0	No	Intergraph	Connections to GDW via the GDO Oracle Objects Read-Only dataserver.
LRS GeoMedia Custom Command	No	Intergraph Project Team	Might require additional connectivity of Oracles OLEDB provider. This to be decided.
SQL-Plus	No	Oracle	Used as command interface to run stored procedures etc.

2.4.2 Maintenance Workstation

This software inventory is specific to the maintenance seats. Maintenance seats are assumed to be primarily high-end GIS workstations although some processing such as staging data into the GDW can be performed from other database connectivity software such as SQL / Plus.

Table 3 - Maintenance Workstation Software Inventory

Software	**Required**	**Vendor**	**Notes**
Microsoft NT 4.0 SP5 or Windows 2000	Yes	Microsoft	Windowing system must be compatible with GeoMedia Professional.
Net8 Connectivity	Yes	Oracle	Compatible with LRS server. Version 8.1.6 is the targeted platform.
GeoMedia Professional 4.0	Yes	Intergraph	Connections to GDW via the GDO Oracle Objects Read-Write dataserver.
GeoMedia Professional Network Manger	Yes	Intergraph	This is a new product that is being developed by Intergraph to support applications which use linear referencing schemes.
SQL-Plus	No	Oracle	Used as command interface to run stored procedures etc.

2.4.3 Oracle Database Server

This software inventory is specific to the Database Server.

Additional software to manage the Oracle instance and perform routine backup and restores should also be considered. Examples are dbArtisan from Embarcadero Technologies for managing, monitoring and tuning the database and ARCServe from CAI for backing up the server(s) to offline storage media.

Table 4 - Oracle Database Server Software Inventory

Software	Required	Vendor	Notes
Windows NT Server 4 SP5 or Windows 2000 Server	Yes	Microsoft	
Oracle 8i Enterprise Release 2 (8.1.6)	Yes	Oracle	Must be installed with the JServer and associated components.
Oracle 8i Spatial Option	Yes	Oracle	
Oracle SDO and XML Java APIs	Yes	Oracle	These are distributed as free components from Oracle's Technology Network.
LRSx V1.0 Linear Referencing System Extensions for Oracle 8i	Yes	TransDecisions Inc.	This will be installed as part of the LRS installation. A perpetual run-time license will be granted to Iowa DOT.
SQL / Plus	No	Oracle	Used as command interface to run stored procedures etc.

2.5 Skills and Level of Effort for Component Development

The following table details the skills necessary to construct the individual components and approximates the level of effort in terms of person-hours necessary to construct, integrate, debug, and document the components.

Table 5 - Component Development Estimated Level Of Effort

Tier	**Component**	**Development Environments**	**Skills**	**Hours**
Client	GeoMedia Custom Commands	GeoMedia Visual Basic 6.0 Oracle Objects for OLE	Experience developing GeoMedia Custom Commands in Visual Basic. Experience developing GeoMedia custom pipes. Understanding of the 20-27(2) model and Iowa's implementation. Basic knowledge of Oracle 8i and stored procedures.	120 hours to program the components. Additional 8 hours to set up the database to develop against. Additional 20 hours to debug and document the system. Additional 40 hours to develop a user manual.
Maintenance[1]	Conflation Editing Commands	GeoMedia Pro Visual Basic 6.0	System design, advanced programming skills and experience developing GeoMedia Pro Custom Commands in Visual Basic. Understanding of the 20-27(2) model and Iowa's implementation. Basic knowledge of Oracle 8i and stored procedures.	150 hours to design and provide specifications. 1000 hours to program the editing interface and provide documentation.
Server	Transformation Algorithms	Java Oracle JServer	Advanced programming skills in Java and the Oracle JServer environment. Experience with math programming applications, system design, object oriented	280 Hours

[1] Intergraph has suggested that a GeoMedia Pro-based Network Manager product is being developed which will perform much of the functionality for these workflows.

Tier	Component	Development Environments	Skills	Hours
			design and database architectures.	
	Linear Overlay	Java Oracle JServer	Advanced programming skills in Java and the Oracle JServer environment. Experience with math programming applications, system design, object oriented design and database architectures.	40 Hours
	Staging Algorithms	Java, Oracle JServer PL / SQL	Sound knowledge of Oracle stored procedures, packages and Oracle DBA skills.	120 Hours
	API and documentation	Java, Java DOC.	Understanding of Java, PL / SQL and technical documentation	160 Hours
	Installation and Testing	JServer and Java tools. Oracle instance management tools.	DBA and Java skills.	80 Hours

3 GEODATA WAREHOUSE

The GeoData Warehouse is an Oracle schema that contains the staged business and LRS data. The major access to the GDW will be through Intergraph's GeoMedia GIS client. GeoMedia will have the capability to view and map any of the data that has been staged into the GDW. GeoMedia will also, through the addition of the custom commands detailed below, be able to perform linear transformations and linear overlays against the LRS by utilizing the Business Logic tier of the system.

3.1 User Interface Components

This section contains a description and processing overview of the GeoMedia components that are required to provide interfaces for interaction with the GDW components on the Oracle server.

There are three new components that need to be developed and a general requirement for GeoMedia to view and analyze the data that has been staged into the GDW. These three components are GeoMedia custom commands[2]. The first GeoMedia custom command is a Reference Calculator that allows the GeoMedia user to enter a location reference using a selected LRM and return the equivalent location reference using an alternate LRM. The second custom command is a transformation command that allows the user to convert a recordset containing location reference information in one LRM to another LRM. This command can also add cartographic and datum references to the recordset. The third custom command is a linear overlay wizard, which allows the user to overlay one recordset containing location references, on a table containing linear features that have been staged into the GDW, and return the result as a third recordset.

3.1.1 Viewing and Analysis

3.1.1.1 Overview

GeoMedia will be able to connect to the GDW warehouse using a read-only GDO connection. Once connected to the GDW, a GeoMedia client will be able to list the available feature tables and operate on them using normal GeoMedia commands and functions.

Business data is stored in tables in legacy systems that will be staged into the GDW using specialized procedures. As this data is staged, at a minimum, a datum reference and a geometry column will be added to each table and populated for every row. References to the tables will also be added to the GDW, SDO and LRSx metadata systems.

[2] A GeoMedia custom command is user interface component that is packaged as an OCX and can be added into the GeoMedia menu structure using tools provided by Intergraph Corp. The custom command OCXs are typically written in Microsoft's Visual Basic language.

Other, non-business data from the LRS schema will also be staged. This data includes, but is not limited to Centerline, Anchor Sections, Anchor Points, Transportation Links, Transportation Nodes, and Routes. Each of these logical components will be processed through a staging algorithm and have geometry and datum references (with the exception of the centerline, and datum components) added to them. Other reformatting and simplification processes will be performed to aid in the viewing of the respective components from the GeoMedia client. In some cases, staged data will contain multiple geometry columns; Anchor Sections for example.

3.1.1.2 Requirements

GeoMedia will be able to connect to the GDW and perform the following functions:

- List all the feature tables that are available in the GDW.
- Allow the user to select any feature table and display the table in a GeoMedia data view window.
- Allow the user to select any feature table containing one or more geometry columns and display the feature table using a specific geometry column if more than one exists, in a map view. The user will be able to create thematic displays based on column values associated with and table.
- Allow the user to create a client-side query against any feature table using the standard GeoMedia query building tools.
- Allow the user to process any recordset generated against the GDW connection using standard GeoMedia analysis workflows.

3.1.2 Reference Calculator

3.1.2.1 Overview

The Reference Calculator is a GeoMedia custom command to be developed by Intergraph Corp. The Reference Calculator allows the user to enter a location reference using a selected LRM or geographical location, and return the location reference as an alternate LRM or geographical location. The transformation from the input LRM value to the output LRM should be performed using a server-side stored procedure.

3.1.2.2 Requirements

The Reference Calculator will utilize the following workflow.

1. The user will be prompted to select an input LRM from a pre-defined list of available location reference methods. The LRM methods will include the following list
    - LRS Milepoint

- Reference Post
- Literal Description
- Project Stationing
- GIMS Section
- HPMS Section
- Anchor Section and Offset
- Coordinate Route
- Geographical Location (X, Y)

2. The user will be prompted to elect to enter a point or linear location reference. If the input LRM is a GIMS or HPMS section, a section ID will be prompted for. If the input LRM is geographic, then the user is restricted to point events only.

3. If the user elected to enter a point, a single instance of the input location reference will be required. If a linear extent was selected, a beginning and ending location reference will be required. In the case of a linear extent, the beginning and ending location reference will use the same LRM.

4. The user will select an output LRM from the list of available LRMs specified in step 1.

5. The GeoMedia command will format a request to be sent to a stored procedure on the LRS server. The stored procedure will evaluate the transformation from the input LRM to the output LRM and return the result to the GeoMedia client.

6. If the transformation was successful the Reference Calculator will interpret the result and display it in a form consistent with the LRM format definitions.

7. The command will prompt the user to enter another location reference for conversion or allow them to exit the command.

3.1.3 Linear Transform Wizard

3.1.3.1 Overview

The Linear Transform Wizard is a GeoMedia custom command developed by Intergraph Corp. which allows the user to process an input feature class containing features referenced using one LRM method and associate a location reference from another distinct LRM with each feature in the feature class.

The Linear Transform Command utilizes a GeoMedia AddFieldsPipe processing framework to take an input recordset and add new columns that contain the output location reference for each row. The output location reference values for each row in the input recordset are calculated by delegating the transformation to the LRS by executing a stored procedure on the server. The results of the stored procedure are formatted into the columns which have been added by the GeoMedia AddFieldsPipe. There will be a single row in the output recordset for each row in the input recordset.

3.1.3.2 Requirements

The Linear Transform Wizard uses a "Wizard" approach to guide the user through a series of ordered steps which allows the user to specify the transformation to be performed and what to do with the results of the transformation. The Wizard will perform the following workflow.

1. The user is prompted to select an input recordset. The recordset can be either a feature class or a recordset from an existing query object.

2. The user is prompted to select a LRM for the input recordset. Valid input LRM types are:

    - LRS Milepoint
    - Reference Post
    - Literal Description
    - Project Station Post
    - GIMS Section
    - HPMS Section
    - Anchor Section and Offset
    - Coordinate Route
    - Geographical Location

3. The user is prompted to specify whether the input recordset contains point data, linear data. This step is skipped for HPMS and GIMS sections.

4. The user is prompted to enter the fields that are used to contain the LRM information. The number and specification of the fields will be dependent on the LRM selected. Note, if the input LRM is geographic then the user is restricted to point events only.

5. The user is prompted to select an output LRM. The same list of LRMs will be presented as in 2.

6. A default set of output field names is generated to hold the output LRM location reference information. The actual specification of the fields is dependent on the selected output LRM.

7. The user is prompted to enter the name for the query to hold the output recordset.

8. The user can elect to display the output query in a data view or, if geometry was generated during the transformation, display it in a map view.

9. The user is prompted to start the geo-processing operation. When the processing has completed the command is terminated and removed from the screen.

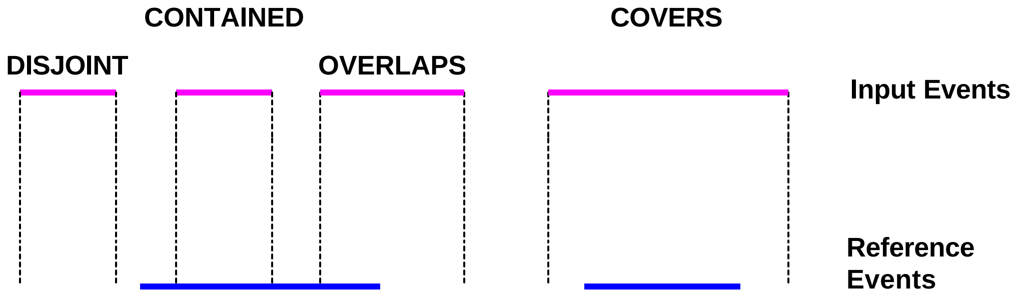
3.1.4 Linear Overlay Wizard

3.1.4.1 Overview

The Linear Overlay Wizard is a GeoMedia custom command developed by Intergraph Corp. which allows the user to perform a linear overlay operation between an input recordset containing linear or point events and a feature class containing business data which has been previously staged and is stored in the GDW. The actual overlay operation is performed on the LRS server using a stored procedure.

The Linear Overlay Command utilizes a GeoMedia BaseTablePipe processing framework to take an input event recordset and construct an output recordset derived from processing each row in the input recordset against a staged feature class in the GDW. The overlay process will calculate either the union or the difference between every row in the input recordset and the helper recordset. Each row in the input recordset will create zero or more rows in the output recordset depending on the results of the overlay operation.

Figure 9 - Relationships Between Linear Features

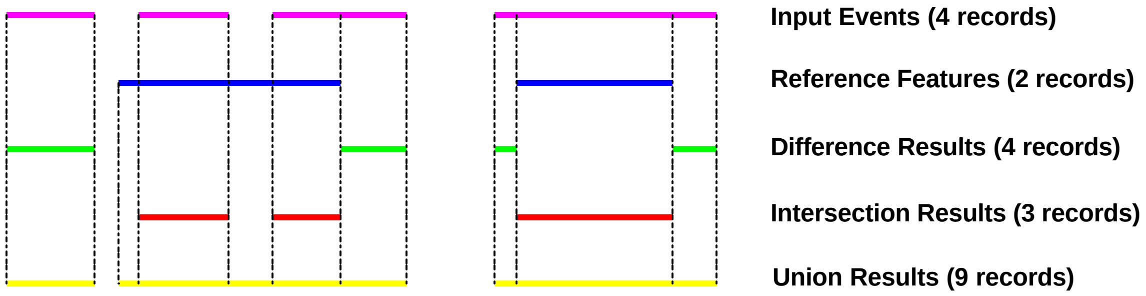


Given two linear events, there are four possible outcomes of an overlay operation. These relationships are illustrated in Figure 9. The two events might be *disjoint*, one event might be *contained* (*covered*) by the other event, or the events might overlap for some portion of their extent. Linear *equality* is an additional outcome wherein two linear features share the same linear space. This property is a generalization of contains as one linear feature exactly contains the other.

If either of the overlay events are point events note that the set of linear relationships is restricted to DISJOINT or CONTAINED.

The Overlay Wizard requires that the input recordset contains a datum reference which is the metric used to perform the overlay operation. If the input recordset does not contain a datum reference it must be processed through the LRM Transform Pipe (See Section 3.1.5.1) to create a datum reference prior to its use by this command.

Figure 10 - Results of Linear Overlay Operations


Input Events (4 records)
Reference Features (2 records)
Difference Results (4 records)
Intersection Results (3 records)
Union Results (9 records)

The Overlay Wizard sends the datum reference for each row in the input recordset to the server to be overlaid against a reference dataset. The wizard allows the user to specify whether the linear difference, linear union, or linear intersection of the overlay operation is performed. Figure 10 illustrates the results associated with the difference, intersection and union operators. The difference operator returns the portions of the input events that are not contained by any feature in the reference table. The Intersection operator returns those portions of the input events that are contained by the one or more features in the reference table. The union operator returns the portion of each input event that is either disjoint or contained by a reference feature.

In each case, each row in the input recordset will generate zero, one, or more output rows. In the case of the intersection and union operators, the user can specify which attributes of the reference recordset can be attached to the output recordset.

3.1.4.2 Requirements

The Linear Overlay Wizard uses a “Wizard” approach to guide the user through a series of ordered steps which allows the user to specify the overlay operation to be performed and what to do with the results of the transformation. The Wizard will perform the following workflow.

1. The user is prompted to select an input recordset. The recordset can be either a feature class or a recordset from an existing query object. The input recordset must contain datum references.

2. The user is asked to specify if the input recordset contains linear, point or section events.

3. The user selects a table containing business data from the list of business data that has been staged into the GDW. This is the reference table.

4. The user is asked to select a difference, intersection, or a union operation.

5. If the user selected a union operation, the user is asked to select one or more columns from the reference table to join to the data from the input recordset, which is then placed in the output recordset.

6. The user is asked if they would like a geometry field appended to the output recordset.

7. The user is prompted to enter the name for the query to hold the output recordset.

8. The user can elect to display the output query in a data view or, if geometry was generated during the transformation, display it in a map view.

9. The user is prompted to start the processing. When the processing is completed the command is terminated and removed from the screen.

The Linear Overlay Wizard performs only a pure linear overlay. Additional operations can be combined with the overlay wizard to produce more complex results. This will be necessary to meet the following DOT LRS analysis requirements:

- Overlay a linear input recordset with a point reference recordset. For example, input pavement sections overlaid with a reference accident set.

- Determine the frequency, average, or other statistic of an event based on one or more of its attributes that occur along another event based on one or more of its attributes. For example, the number of fatal accidents associated with each pavement section, the pavement section with the most accidents, and the average number of accidents by pavement condition.

These more complex results can be satisfied using GeoMedia. Once two recordsets are overlaid– for example - accident input records have pavement ids associated with them - GeoMedia can be used to physically join the two tables together using the Join Query command. This command will allow statistical and summary operations to be performed on the resultant recordset. These types of operations can be automated as necessary by creating new GeoMedia custom commands or by providing SQL statements to provide the results.

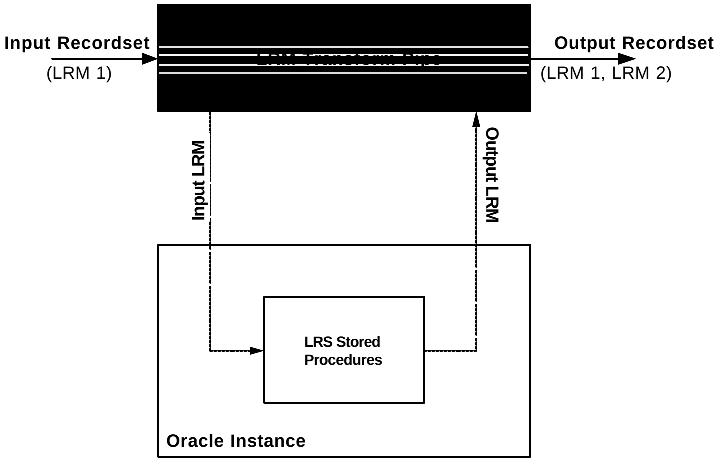
3.1.5 GeoMedia Components

There are two custom GeoMedia pipes that will be developed to create the user interface components detailed above. These pipes are

3.1.5.1 LRM Transform Pipe

The GeoMedia LRM Transform Pipe is a custom AddFieldsPipe that appends an output LRM to an input recordset containing an input LRM. The row-specific values for the output LRM are calculated by executing a stored procedure on the server containing the in the LRS Oracle instance. The pipe is a standard GeoMediaAddFields pipe and produces a single output record for every input record.

Figure 11 - LRM Transform Pipe Conceptual Design



The LRM Transform Pipe and the Oracle stored procedures use a formal protocol to pass structured data over the network. XML is a document format that will suit the requirements of the formal protocol and provide a generic system for passing data between the systems. The LRM Transform Pipe would perform the following steps to perform a single linear transformation on the server:

1. Encode a location reference as an XML document. The XML document can describe the LRM and provide the correct fields for the location reference. The XML document could also specify the output LRM format.

2. Pass the XML document as a text string to a stored procedure on the Oracle server.

3. The stored procedure extracts the data from the XML document and performs the transformation.

4. The results of the transformation are encoded into an XML document which is returned to the Pipe as a text string.

5. The LRM Transform Pipe decodes the XML document returned from Oracle and interprets the results.

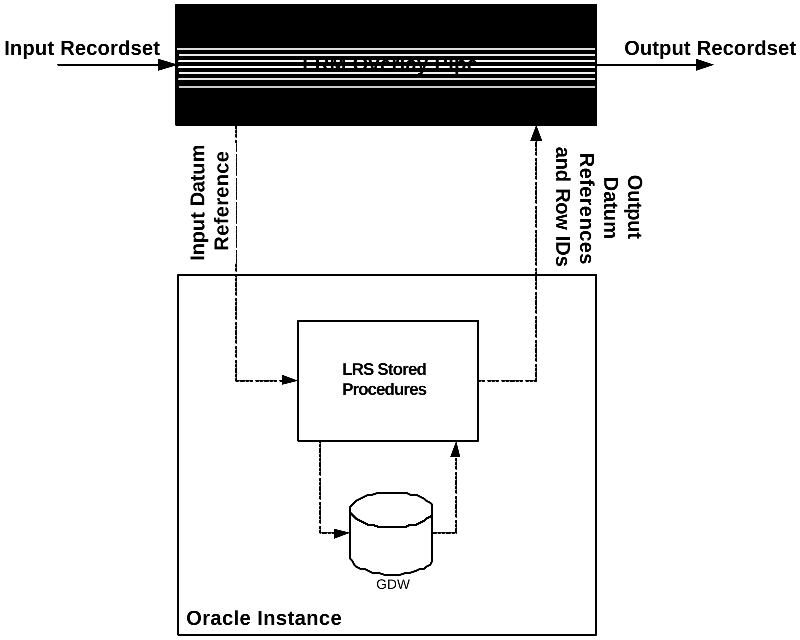
This LRM Transform Pipe forms the processing engine for the Reference Calculator and Linear Transform Wizard commands.

3.1.5.2 Linear Overlay Pipe

The GeoMedia LRM Overlay Pipe is a custom GeoMedia Pipe based on the GeoMedia BaseTablePipe that creates rows in an output recordset from rows in an input recordset. The row-specific values for the output recordset are calculated by sending a datum reference to, and executing a stored procedure on, the server containing the LRS Oracle instance. The stored procedure returns a structure that contains a list of all the record Ids for each reference feature which has a linear relationship with the input datum and the datum reference for the extent of the relationship.

If the operation was a disjoint overlay, the input record will be copied to the output recordset for every query which returns an empty list of related reference features. If the pipe is performing a union overlay, a row will be generated in the output recordset for every element in the list that is returned from the server-side query. The row ID of the related records will be used to append joined attributes to the output recordset.

Figure 12 - LRM Overlay Pipe Conceptual Design



The LRM Overlay Pipe and Oracle communicate using the same formal protocol as the LRM Transform Pipe. This LRM Overlay Pipe forms the processing engine for the Linear Overlay Wizard command.

3.1.6 Example GeoMedia Workflow.

This example workflow illustrates the use of the GeoMedia custom commands for performing an ad-hoc analysis. The example workflow is to take a set of accident locations stored in an Excel Spreadsheet, draw them in a map and determine the pavement quality for the location of each accident. The pavement sections are part of the GDW. The accidents are located using a literal description LRM.

Creating a Recordset

1. The user creates a connection using the new GeoMedia ODBC GDO server to the Excel Spreadsheet containing the accident data.

2. The user creates a GeoMedia query against the accident table in the spreadsheet and loads the results of the query into a data view.

Adding DATUM references to the recordset.

3. The user selects the Linear Transform Wizard and, using the wizard, specifies the accident query as the input recordset.

4. The user then specifies the input LRM to be LITERAL DESCRIPTION and that the output LRM will be DATUM. The user also elects to add a geometry object derived from the DATUM reference for each accident to the output recordset.

5. The wizard leads the user through the process of selecting the columns that contain the literal description information. The user specifies to create an output query recordset called "accidents_with_datum" which is stored in the query folders.

6. The user starts the transform process and GeoMedia creates a new query object containing a recordset with the accidents and a datum reference for each accident.

Performing a Linear Overlay

7. The user starts the Linear Overlay Wizard and selects the new accident query (accidents_with_datum) recordset that was generated by the Linear Transform Wizard as the input recordset.

8. The user selects PAVEMENT_SECTION from the list of linear features that are available in the GDW.

9. The user elects to perform a UNION operation and to join the PAVEMENT_SECTION_ID and PAVEMENT_CONDITION attributes from the PAVEMENT_SECTION table with the input recordset.

10. The user starts the overlay operation.

When completed GeoMedia creates a new recordset that is loaded into the current map view. Each record in the new recordset has the original accident information, a DATUM reference, a GeoMedia Geometry blob, and the pavement ID and pavement condition value associated with the pavement section on which the crash is located.

Once the accident records have pavement ids associated with them, GeoMedia can be used to physically join the two tables together using the Join Query command, which will allow statistical and summary operations to be performed on the resultant recordset. For example, once joined it would be possible to determine the number of fatal accidents associated with each pavement section, the pavement section with the most accidents, and the average number of accidents by pavement condition. These types of operations can be automated as necessary by creating new GeoMedia custom commands or by providing SQL statements to provide the results.

3.2 Metadata

Data that is staged into the GDW will have a row inserted into a GDW Metadata table. This table is designed to provide a guide for GeoMedia users about the contents of the Oracle component of the GDW. The Metadata table will provide the following information about each table that is staged:

- **Table Name**. This will also include the names of views in the GDW.
- **Table Description**: Provides a textual description of the information contained in the table or view.
- **Date Source Date**: This provides a date value for the table as a whole and its definition is dependent on the table. For example, year that the data was collected.
- **Date Staging Date**: This provides the time and date that the data was staged into the GDW.
- **Contact**: This provides the name and phone number of someone to contact about the data.

It is anticipated that a more detailed metadata system could be developed in the future which would replace this table. Any such metadata system that is developed in the future must be compatible with GeoMedia and the CTAMS systems.

3.3 Data Staging

The staging of LRS data is a managed process that is performed periodically by a GDW administrator. Staging takes data stored in a table from either a legacy system or the LRS schema and places a copy of the data in a new table in the Oracle schema associated with the GDW. Both business data and LRS data can be staged. The staging of these disparate types of data are discussed separately.

3.3.1 Staging Process Overview

During the processing stage the following operations will be performed.

3.3.1.1 Tables

For each table that is staged:

- A row referencing the staged table will be added into the SDO metadata structure.
- A row referencing the staged table will be added into the LRSx metadata structures.

- A row referencing the staged table will be added into the GDW metadata structures.

The actual processing steps for each table will depend on the type of data being staged. The two major categories of staged data are business data (crashes, pavement type, etc.) and LRS Component data (transport links and nodes, route links, etc.).

3.3.1.2 Table Rows

For each row in an input table that is staged

- A datum reference will be added to each row. If the row already contains a datum reference it will be verified and replaced if necessary.

- A SDO geometry object will be added to each row. If the row already contains a geometry object that has been generated against the datum it will be kept. If the existing geometry object has not been generated against the datum an additional geometry object will be added to the table.

- Other table-specific processing will take place. For example, mapping temporal values stored as dates into categories that enable thematic display in the GeoMedia environment.

3.3.2 Staging Interfaces

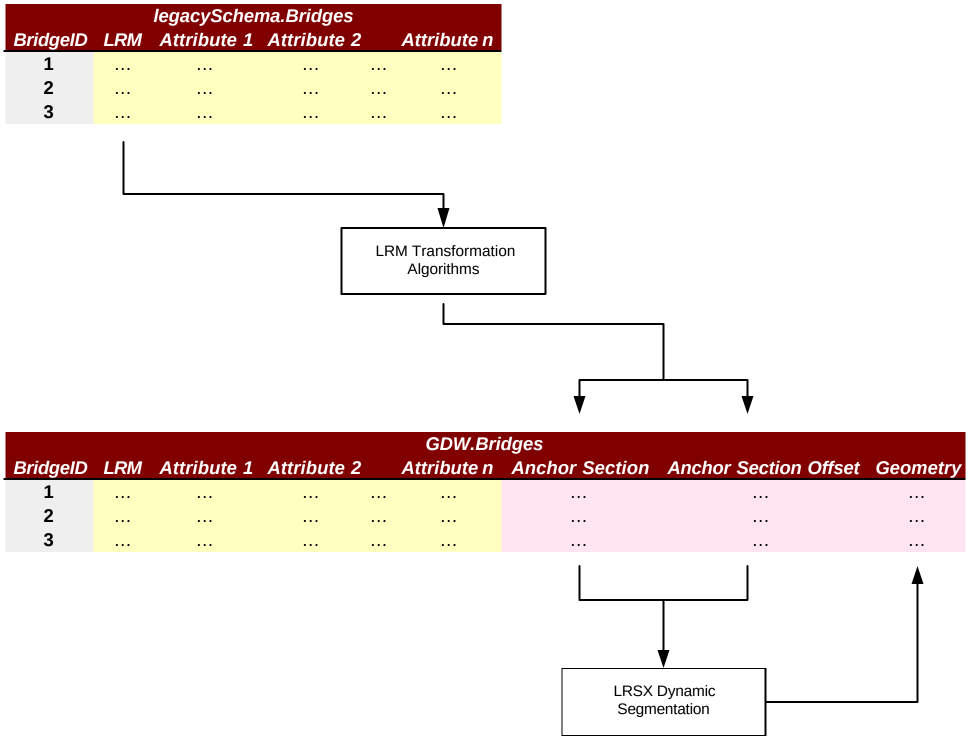
The staging of a particular table is accomplished via a stored procedure that is part of the Business Logic. There is one stored procedure for every table that is to be staged by an administrator. Additional tables can be staged by creating new stored procedures. Using stored procedures allows legacy data to be staged without transformation of its location reference format as the staging process will be specific to each table and can interpret LRM formats from the tables column values.

The staging procedures use the transformation algorithms to transform the LRM for a table into a staged Datum reference.

3.3.3 Business Data

The first category of stageable data is business data that exists in a table in some schema accessible by a GDW administrator. Examples of business data that can be staged are bridges, pavement sections, GIMS segments, HPMS Segments, crashes, etc. As the table is staged, each row in the business table is sent through a transformation algorithm that converts the business data’s location reference stored in one of the supported LRM formats and generates a datum reference. The datum reference is added to the table definition in the GDW. The staging process will then take the datum reference and perform a dynamic segmentation operation against the datum itself and generate a cartographic representation of the row. The cartographic representation will be stored in a SDO geometry object in the table in the GDW. This process is illustrated in Figure 13.

Figure 13 - Staging Steps for a Table Containing Typical Business Data



legacySchema.Bridges

BridgeID	LRM	Attribute 1	Attribute 2		Attribute n
1	...	...	...	...	...
2	...	...	...	...	...
3	...	...	...	...	...

GDW.Bridges

BridgeID	LRM	Attribute 1	Attribute 2		Attribute n	Anchor Section	Anchor Section Offset	Geometry
1	...	...	...	...	...	...	...	...
2	...	...	...	...	...	...	...	...
3	...	...	...	...	...	...	...	...

The actual staging process is run as a stored procedure available from the LRS schema. The following processing steps are performed by the stored procedure:

1. The business data stored in some schema as part of a legacy system is copied into the GDW schema. The user defines the name for the GDW version of the table as a parameter to the stored procedure.

2. The GDW version of the table is modified to add datum specific columns and a column to contain the cartographic representation of each feature.

3. One of the LRM transformation algorithms is invoked on every row of the table to derive Anchor Section and Anchor Section Offset values. These values are stored in the staged table in the new columns that have been added to the table.

4. The anchor section and anchor section offset values are used by the LRSx Dynamic Segmentation engine to derive a cartographic representation of the object against the staged version of the datum.

5. The new table information is entered into the SDO metadata table to make it available to GeoMedia clients.

6. The new table is entered into the LRSx Event Metadata table.

7. The new table is added to the GDW metadata table.

For business data containing linear events, the staging process creates the following datum references

- Beginning Anchor Section and Anchor Section Offset values from the beginning location reference LRM.
- Ending Anchor Section and Anchor Section Offset values from the ending location reference LRM.
- An Anchor Section Conflation object containing the conflation of the event against the datum.

The anchor section conflation object is added to the table to support the linear overlay operations that can be performed using the table as a reference feature. This information can be recovered from the beginning and ending location references, but this operation is costly and would have to be performed for every row for every client that requests a linear overlay operation using the table as a reference feature class. Storing a derived conflation object is more efficient and will optimize the processing of linear overlays.

3.3.4 LRS Components

The second category of data that needs to be staged consists of components of the LRS that exist in the LRS schema and have utility as part of the GDW. The list below contains those components.

- Centerlines
- Datum (Anchor Sections and Anchor Points)
- Network (Transportation Links and Transportation Nodes)
- Routes (routes along the network)
- Reference Posts
- Improvement Projects (Project Sections and Project Nodes)
- Station Posts

Each of these tables is processed via separate staging algorithms which are detailed below. During the staging process, the input LRS elements are transformed into GIS features as they are written to the GDW. In many cases, this is a semantic transition. However, the purpose is to create geographical features from the linear elements of LRS which can be mapped, reported and analyzed using the GIS tools available as part of the GeoMedia toolkit.

Note that the LRM transformation algorithms specified in Appendix A operate against the LRS schema and not the data that is staged into the GDW. This enables the LRS “intelligence” to be removed from the LRS components as they are being staged which reduces the complexity of the features and trims the tables making them smaller and faster for the GeoMedia client-side processes.

3.3.4.1 Centerlines

The centerline table is staged by copying the table containing the centerline rows into the GDW. The table already contains SDO geometry objects that do not need to be modified.

3.3.4.1.1 Metadata

The Centerline table will be entered into the GDW metadata and the SDO metadata tables.

3.3.4.2 Anchor Sections

The Anchor Section table is staged by copying the table into the GDW schema. During the processing stage, each anchor section record is given a new attribute that denotes its temporal classification. The following columns are removed from the table during the staging process.

Column	**Reason for Removal**
CONFLATION_GIMSCARTO	Not required as part of the GDW.

The following columns are added to the Anchor Section table in the GDW.

Column	**Definition**	**Description**
TEMPORAL_STATE	NUMBER(1,0)	This is a derived attribute that is created from a combination of the DATE_RETIRED attribute and the association of the anchor section with the improvement projects. The possible values are -1 (retired), 0 (current), 1(proposed), 2 (planned), 3 (under construction). This is necessary to allow GeoMedia clients to create thematic displays and queries against the current version of the datum.

3.3.4.2.1 Anchor Section Views

The anchor section table will be staged as a single table containing all retired, current, and proposed anchor sections. To accommodate the more fundamental requirements of the majority of the GDW users, a series of views will be made in the GDW which will allow the GeoMedia user to select an appropriate level of detail for their application. In addition, the complete anchor section table will be available for sophisticated users to work with. The following views are planned. Other views may be added as necessary.

View Name	Description
ANCHOR_SECTION_CURRENT	All anchor sections with a TEMPORAL_STATE value of –1 or 0. This is the default view for anchor sections.
ANCHOR_SECTION_COMPLETE	All anchor sections.
ANCHOR_SECTION_PROPOSED	All anchor sections with a TEMPORAL_STATE value > 0.
ANCHOR_SECTION_RETIRED	All anchor sections with a TEMPORAL_STATE value of –1.
ANCHOR_SECTION_CURRENT_ONLY	All anchor sections with a TEMPORAL_STATE value of 0.

3.3.4.2.2 Metadata

Only the views will be entered into the GDW and SDO metadata. The views will also be entered into the LRSx LRS metadata table.

3.3.4.3 Anchor Point

The Anchor Point table is staged by copying the table into the GDW schema. This table is staged in a similar manner to the anchor sections. No columns are removed from the table during the staging process. The following columns are added to the Anchor Point table in the GDW.

Column	Definition	Description
TEMPORAL_STATE	NUMBER(1,0)	This is a derived attribute that is created from a combination of the DATE_RETIRED attribute and the association of the anchor point with the improvement projects. The possible values are -1 (retired), 0 (current), 1(proposed), 2 (planned), 3 (under construction). This is necessary to allow GeoMedia clients to create thematic displays and queries against the current version of the datum.

3.3.4.3.1 Anchor Point Views

The anchor point table will be staged as a single table containing all retired, current, and proposed anchor points. To accommodate the more fundamental requirements of the majority of the GDW users, a series of views will be made in the GDW which will allow the GeoMedia user to select an appropriate level of detail for their application. In addition, the complete anchor point table will be available for sophisticated users to work with. The following views are planned. Other views may be added as necessary.

View Name	Description
ANCHOR_POINT_CURRENT	All anchor points with a TEMPORAL_STATE value of –1 or 0. This is the default view for anchor points.
ANCHOR_POINT_COMPLETE	All anchor points.
ANCHOR_POINT_PROPOSED	All anchor points with a TEMPORAL_STATE value > 0.
ANCHOR_POINT_RETIRED	All anchor points with a TEMPORAL_STATE value of –1.
ANCHOR_POINT_CURRENT_ONLY	All anchor points with a TEMPORAL_STATE value of 0.

3.3.4.3.2 Metadata

Only the anchor point views will be entered into the GDW and SDO metadata.

3.3.4.4 Transport Link

The Transport Link table is staged by copying the entire table into the GDW schema. No columns are removed from the table during the staging process. The following columns are added to the transport link table in the GDW.

Column	Definition	Description
TEMPORAL_STATE	NUMBER(1,0)	This is a derived attribute that is created from a combination of the DATE_RETIRED attribute and the association of the transport link with the improvement projects. The possible values are -1 (retired), 0 (current), 1(proposed), 2 (planned), 3 (under construction). This is necessary to allow GeoMedia clients to create thematic displays and queries against the current version of the transport network.
DIRECTION_OF_TRAVEL	NUMBER(1,0)	This is a derived attribute which denotes the permitted direction of travel on the transport link. The values are –1 (reverse direction only), 0 (bi-directional), 1 (forward direction only). The information to populate this field is taken from the TRANSPORT_LINK_STATUS table.

Column	Definition	Description
IS_CLOSED	NUMBER(1,0)	This is a derived attribute that is created from the linked row in the TRANSPORT_LINK_STATUS table. The possible values are: 0 (Not closed), 1 (Closed). The value of this column would override the DIRECTION_OF_TRAVEL column value in an application context.

3.3.4.4.1 Transport Link Views

The transport link table will be staged as a single table containing all retired, current, and proposed transport links for all transportation systems in the LRS. To generate a application specific network a series of join-views will be created with the TRANSPORT_LINK_XREF[3] table. Each view will render a transportation network. In addition, default views will be generated for the transport link table.

Each network-specific view will be named NETWORK_LINK_xxxx where xxxx is the name of the particular transportation network. The use of the NETWORK prefix denotes that this table is optimized for use by GeoMedia Network and contains a network application-level view of the transportation system. The network views can be tailored to meet the demands of various network applications.

The following views are planned. Other views may be added as necessary.

View Name	Description
TRANSPORT_LINK_CURRENT	All transport links with a TEMPORAL_STATE value of –1 or 0. This is the default view for transport links.
TRANSPORT_LINK_COMPLETE	All transport links.
TRANSPORT_LINK_PROPOSED	All transport links with a TEMPORAL_STATE value > 0.
NETWORK_LINK_AMES (example)	All transport links with a TEMPORAL_STATE of 0 which are part of the AMES transportation system.
TRANSPORT_LINK_CURRENT_ONLY	All links with a TEMPORAL_STATE value of 0.

[3] The TRANSPORT_SYSTEM_XREF table may or may not be staged into the GDW. It is possible to provide a join-view between tables in different schemas although this makes the GDW and LRS inter-related rather than inter-operable. If the GDW and LRS schema were in separate database instances it would be beneficial to duplicate the XREF table.

3.3.4.4.2 Metadata

All the views will be entered into the GDW and SDO metadata.

3.3.4.5 Transport Node

The Transport Node table is staged by copying the entire table into the GDW schema. The following columns are removed from the table during the staging process.

Column	Reason for Removal
REFERENCE_FEATURE	Not required as part of the GDW.

The following columns are added to the Transport Node table in the GDW.

Column	Definition	Description
TEMPORAL_STATE	NUMBER(1,0)	This is a derived attribute that is created from the association of the transport node with the improvement projects. The possible values are -1 (retired), 0 (current), 1(proposed), 2 (planned), 3 (under construction). This is necessary to allow GeoMedia clients to create thematic displays and queries against the current version of the transport network.
IS_CLOSED	NUMBER(1,0)	This is a derived attribute that is created from the linked row in the TRANSPORT_NODE_STATUS table

3.3.4.5.1 Transport Node Views

The transport node table will be staged as a single table containing all retired, current, and proposed transport nodes for all transportation systems in the LRS. To generate a application specific network a series of join-views will be created with the requisite NETWORK_LINK_xxx table. Each view will render the nodes for a particular transportation network. In addition, default views will be generated for the transport node table.

Each network-specific view will be named NETWORK_NODE_xxxx where xxxx is the name of the particular transportation network. The use of the NETWORK prefix denotes that this table is optimized for use by GeoMedia Network and contains a network application-level view of the transportation system. The network views can be tailored to meet the demands of various network applications. The following views are planned. Other views may be added as necessary.

View Name	Description
TRANSPORT_NODE_CURRENT	All transport nodes with a TEMPORAL_STATE value of –1 or 0. This is the default view for the transport nodes.
TRANSPORT_NODE_COMPLETE	All transport nodes.
TRANSPORT_NODE_PROPOSED	All transport nodes with a TEMPORAL_STATE value > 1.
NETWORK_NODE_AMES (example)	All transport nodes with a TEMPORAL_STATE of 0 which are part of the AMES transportation system.
TRANSPORT_NODE_CURRENT_ONLY	All nodes with a TEMPORAL_STATE value of 0.

3.3.4.5.2 Metadata

All the transport and network node views will be entered into the GDW and SDO metadata.

3.3.4.6 Route

The Route table is staged by copying the table into the GDW schema. The following columns are removed from the table during the staging process.

Column	Reason for Removal
ROUTE_SYSTEM_ID	Replaced by the ROUTE_SYSTEM_TYPE column.

The following columns are added to the Route table in the GDW.

Column	Definition	Description
TEMPORAL_STATE	NUMBER(1,0)	This is a derived attribute that is created from the association of the route with the improvement projects. The possible values are -1 (retired), 0 (current), 1(proposed), 2 (planned), 3 (under construction). This is necessary to allow GeoMedia clients to create thematic displays and queries against the current version of the routes.

Column	Definition	Description
ROUTE_SYSTEM_TYPE	VARCHAR2(30)	This is a de-normalized category for the routes which have the domain values of "STATE", "COUNTY", "MUNICIPAL", "E911", "PRIVATE", "UNOFFICIAL", "INSTITUTIONAL", "RAMP". The values are derived from the ROUTE_SYSTEM_ID column. As additional route types are added to the LRS this domain list will be modified.
GEOMETRY	SDO Geometry Object	A cartographic representation of the extent of the route.
DATE_ESTABLISHED	DATE	Derived from the DATE_ESTABLISHED column of the route table containing the definition of the route.
DATE_RETIRED	DATE	Derived from the DATE_RETIRED column of the route table containing the definition of the route.
RETIRED	NUMBER(1,0)	This is a derived attribute that is created from the DATE_RETIRED attribute. The possible values are 1 (retired) or 0 (current). This is necessary to allow GeoMedia clients to create thematic displays and queries against the current version of the datum.
BEGIN_MPT	NUMBER(10,3)	Derived beginning milepoint for the route.
END_MPT	NUMBER(10,3)	Derived end milepoint for the route.

The route BEGIN_MPT and END_MPT are added to the route table to allow GeoMedia clients to generate milemarker values against the routes and display them in a map view. Note that this capability is a convenience and recommended for display only as the metric is not controlled. Clients wishing to perform analysis using a milepoint LRM are recommended to use the transformation algorithms against the LRS.

3.3.4.6.1 Route Views

The route table will be staged as a single table containing all retired, current, and proposed transport routes in the LRS. The routes may be separated out by route system type for convenience. The following views are planned. Other views may be added as necessary.

View Name	Description
ROUTES_CURRENT	All routes with a TEMPORAL_STATE value of –1 or 0. This is the default view for the transport nodes.
ROUTES_COMPLETE	All routes in all temporal states..
ROUTES _PROPOSED	All routes with a TEMPORAL_STATE value > 1.

View Name	Description
ROUTES_STATE (example)	All routes with a TEMPORAL_STATE of 0 and a ROUTE_SYSTEM_TYPE value of "STATE".
ROUTES_CURRENT_ONLY	All routes with a TEMPORAL_STATE value of 0.

3.3.4.6.2 Metadata

The route views will be entered into the GDW and SDO metadata.

3.3.4.7 Reference Post

The Reference Post table is staged by copying the table into the GDW schema. The following columns are removed from the table during the staging process.

Column	Reason for Removal
ROUTE_LINK_ID	Not required in the GDW.
TRANSPORT_LINK_ID	Not required in the GDW.
TRANSPORT_LINK_OFFSET	Not required in the GDW.

The following columns are added to the Reference Post table in the GDW.

Column	Definition	Description
GEOMETRY	SDO Geometry Object	A cartographic representation of the reference post.
TEMPORAL_STATE	NUMBER(1,0)	This is a derived attribute that is created from the association of the reference post with the improvement projects. The possible values are -1 (retired), 0 (current), 1(proposed), 2 (planned), 3 (under construction). This is necessary to allow GeoMedia clients to create thematic displays and queries against the current version of the transport network.

3.3.4.7.1 Reference Post Views

The reference post table will be staged as a single table containing all retired, current, and proposed reference posts in the LRS.

The following views are planned. Other views may be added as necessary.

View Name	Description
REFERENCE_POSTS_CURRENT	All reference posts with a TEMPORAL_STATE value of –1 or 0. This is the default view for the reference posts.
REFERENCE_POSTS_COMPLETE	All reference posts in all temporal states..
REFERENCE_POSTS_PROPOSED	All reference posts with a TEMPORAL_STATE value > 1.
REFERENCE_POSTS_CURRENT_ONLY	All reference posts with a TEMPORAL_STATE value of 0.

3.3.4.7.2 Metadata

The reference post views will be entered into the GDW and SDO metadata.

3.3.4.8 Project Section, Project Section Node, Station Post

These three table are staged by directly copying the tables without modification into the GDW.

3.3.4.8.1 Metadata

All three tables will be entered into the GDW and SDO metadata.

3.4 Network Staging

Staging of the network components is described above. The structure of the features that comprise the network are consistent with the requirements for GeoMedia Network 4.0 to be used as a network client.

3.4.1 GeoMedia Network 4.0 Requirements

The specific requirements for GeoMedia Network 4.0 are as follows:

3.4.2 Network Links

The set of links in the network are stored in a feature table. The feature table must have the following properties.

- Linear geometries consisting of either simple polylines or composite polylines. These features must be stored in a single feature table in the GDW.
- An ID column that uniquely identifies each row in the transport link table.
- A length measurement on each transport link.

- An optional column containing a direction flag. The flag is an integer consisting of the values –1, 0, +1 where –1 indicates that the permitted flow over the link is in the reverse direction to the direction of the geometry, 0 indicating that bi-direction flow is permitted over the link, and +1 which indicates that flow is only permitted in the direction of the geometry.

3.4.3 Network Nodes

Network nodes are optional. If a node table is present the GeoMedia Network 4.0 application will use an existing node table.

- Point geometries

- An ID column that uniquely identifies each row in the transport node table.

When the network is compiled in the GDW, a Network Metadata table will be added to the GDW schema. Note that a copy of GeoMedia Pro will be required to compile the transport network but once compiled, GeoMedia clients using the GeoMedia Network 4.0 software will be able to perform network operations such as routing tracing in the network.

3.4.4 Staged Transport Network

The requirements for GeoMedia Network 4.0 are met by the staging of the transport layer from the LRS to the GDW and building the network specific views. The TRANSPORT_LINK_ID and TRANSPORT_NODE_ID columns satisfy the unique ID requirements of the respective tables. The SDO_GEOMETRY columns in the tables meet the geometric requirements. The DISTANCE field meets the LENGTH requirement for the link table and the DIRECTION field satisfies the optional requirement for the direction flag on the link table.

Each of the transport system specific views of the transport network will provide an application specific network for use.

4 BUSINESS LOGIC

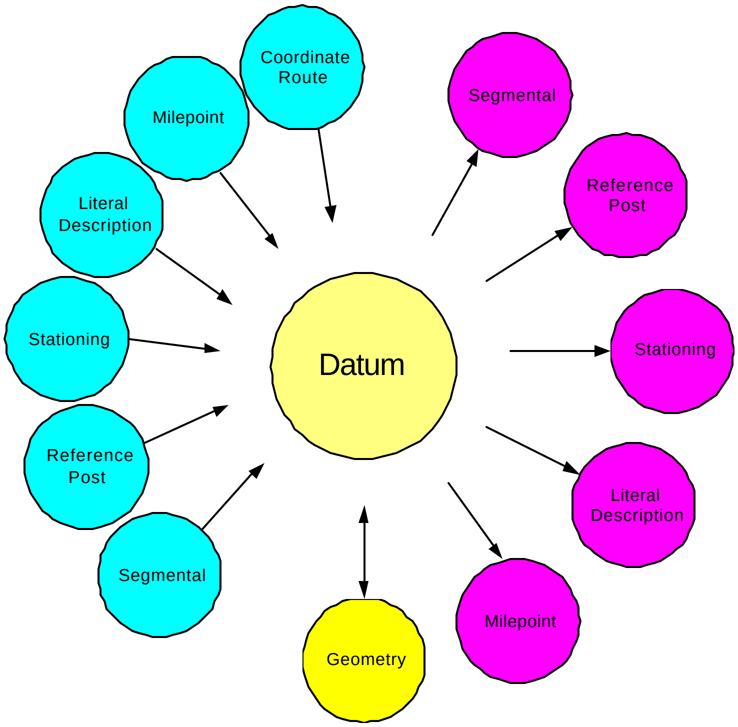
Business Logic comprises the set of code that implements the major functions of the LRS. These functions include code that performs linear transformations, linear overlay operations, and staging of data into the GDW. The Business Logic components provide the major interoperable interfaces to enable other applications on the network to communicate with and use functions of the LRS.

4.1 Linear Transformations

Linear transformations are the major code component of the business logic. These transformations migrate one linear reference location to another, or from one route reference to another within the same LRM. Details of the transformation algorithms are provided in Appendix A.

The general procedure to map a location reference between any two LRMs is a two-stage algorithm. The first stage is to transform the location reference to a datum reference. The second stage is to then convert the datum reference to the LRM. This process is illustrated in Figure 14.

Figure 14 - Transformation Through the Datum



Cartographic locations (geometry) can be mapped to the datum by projecting the geometry to a cartographic representation of the datum. The geometry can then be mapped to an output location reference in any specified LRM. Any input location reference can similarly be mapped to a datum reference and then output as a geometric representation by specifying a cartographic representation of the datum.

4.1.1 Valid Transformations

Table 6 details the valid transformations that are available as part of the LRS. All input LRM values can be mapped through the datum and can be mapped to any of the other LRMs, with the exception of the coordinate route LRM, by performing a datum to LRM transformation.

Table 6 - Valid LRM Transformations

LRM	From Datum	To Datum
Reference Posts	X	X
LRS Milepoint	X	X
Coordinate Route		X
Segmental	X	X
Literal Description	X	X
Stationing	X	X

4.1.2 Transformation Results

The following table details the results from a transformation based on the type of input data (point or linear event).

Table 7 - Result from LRM to LRM Transformations

Input LRM	Event Type	Output LRM	Output Structure
Milepost	Point	Datum	{Anchor Section, Offset}
Milepost	Linear	Datum	Anchor Section Conflation Object
Reference Post	Point	Datum	{Anchor Section, Offset}
Reference Post	Linear	Datum	Anchor Section Conflation Object
Coordinate Route	Point	Datum	{Anchor Section, Offset}
Coordinate Route	Linear	Datum	Anchor Section Conflation Object

Input LRM	Event Type	Output LRM	Output Structure
Segmental	Record	Datum	Anchor Section Conflation Object
Literal Description	Point	Datum	{Anchor Section, Offset}
Literal Description	Linear	Datum	Anchor Section Conflation Object
Stationing	Point	Datum	{Anchor Section, Offset}
Stationing	Linear	Datum	Anchor Section Conflation Object
Datum	Point	Milepoint	{Route Name, Milepoint}
Datum	Linear	Milepoint	{Route Name, FromMilePoint, ToMilePoint}
Datum	Point	Reference Post	{Route Name, ReferencePostName, Offset}
Datum	Linear	Reference Post	{Route Name, FromReferencePostName, FromOffset, ToReferencePostName, ToOffset}
Datum	Point	Segmental	Control Section ID.
Datum	Linear	Segmental	List of control section IDs.
Datum	Point	Literal Description	Text
Datum	Linear	Literal Description	Text
Datum	Point	Stationing	{Project Section ID, Station point}
Datum	Linear	Stationing	{Project Section ID, FromStationPoint, ToStationPoint}

4.1.3 Transformation Interfaces

The transformation algorithms are coded in Java and wrapped with a set of stored procedure interfaces. There will be a variety of programming interfaces for each algorithm that allow different input formats to be used. Depending on the type of transformation, the input parameters can take one of the three following forms:

4.1.3.1 System Format

The system format is internal to the LRS schema and allows the user to specify foreign key values as transformation parameters. For example, the Milepoint to Anchor Section format would be specified as follows

```
LRS.MilepointToDatum([in] TransportSystemID, [in] RouteID,
                      [in] Milepoint,
                      [out] AnchorSection, [out] Offset);
```

In this form, the route parameters have been replaced with the foreign key which uniquely identifies the RouteID for a route in the system. Additional stored procedures allow foreign key values to be calculated. For example, the RouteID key can be determined using the following procedure:

```
    LRS.GetRouteID([in] RouteCategory, [in] RoutePrefix, [in] RouteName,
                               [in] RouteSuffix, [in] GeographicExtent,
                [out] RouteID);
```

4.1.3.2 Application Format

The application format is an external interface which allows the user to specify disaggregated individual parameters. The application format is designed for interoperable systems which maintain route parameters in separate columns in a database table. For example, the Milepoint to Anchor Section format would be specified as follows

```
    LRS.MilepointToDatum([in]TransportSystem, [in] RouteCategory,
                          [in] RoutePrefix, [in] RouteName, [in] RouteSuffix,
                         [in] GeographicExtent, [in] Milepoint,
                        [out] AnchorSection, [out] Offset);
```

This format uses the LRS.GetRouteID function to return the foreign key for the route and then calls the System-format version of the transformation.

4.1.3.3 Interface Format

The interface format is designed to allow user interaction with the system. The interface format uses a parser to extract the route parameters from a standard format input string. For example, the Milepoint to Anchor Section format would be specified as follows

```
    LRS.MilepointToDatum([in] TransportSystem, [in] RouteCategory,
                          [in] “I 80 W”, [in] Milepoint,
                         [out] AnchorSection, [out] Offset);
```

The interface format uses a parser to break the route string into its components and then calls the Application-format version of the transformation.

4.1.4 Point and Linear Transformations

In all cases, point transformations can be coded as linear transformations where the linear extent has no distance (i.e. start and end location references are the same). However, in some cases, algorithm efficiencies can be exploited by treating point and linear transformations independently. Within this document the point and linear transformations are described independently for completeness.

4.2 Staging Algorithms

Data to be staged into the GDW can be categorized into the following three types.

- LRS components
- Business Data from legacy systems
- Other Business Data

LRS components such as the datum, routes, and reference posts are components that have a static format and will be staged at regular intervals. These components will follow exactly the same staging process each time they are staged. The Business data that comes from legacy systems such as Bridge, CRASH, PMIS, etc. also will be staged at regular intervals and will always be in the same format. For these two types of data there will be stored procedures written that allow a GDW administrator to quickly stage a new version of the data to the GDW. The stored procedures will be executable from a database command-line environment such as Oracle's SQL-Plus.

Two options exist to stage business data that is staged on an *ad-hoc* basis or that is presented for staging in a different table structure each time. . First, a staging procedure can be written in PL/SQL which accommodates the data structure and uses the LRM transformation stored procedures to perform a row-by-row stage of the data. If the data format changes each time the data is presented for staging, the stored procedure would need to be modified to accommodate the change. The second option is to use a GeoMedia Client to perform the transformation using the LRM Transformation Wizard and output the transformed data set to the GDW.

4.2.1 Preparing for Staging

Data that is to be staged from a legacy or other system must be prepared before it can be staged. This preparation involves ensuring the following:

- The data has been loaded into an Oracle table in a user-schema (i.e. not the GDW or LRS schema)
- Column names and types of the loaded data have been formatted to ensure compatibility with GeoMedia and Oracle requirements.

If the data is not a directly supported format

- The LRM column structure must coincide with a stored procedure transformation API.

Once the data has been prepared it can be staged by invoking the appropriate staging stored procedure.

4.2.2 Algorithm Structure

Once the data to be staged has been prepared and is accessible to the LRS schema, it can be staged. A GDW administrator who can write into the GDW and use the schema objects stored in the LRS schema will typically perform the staging process.

The typical structure of a staging algorithm is as follows:

1. Select the columns to be output to the staged version of the table
2. Define the input LRM and the columns and formats that describe each row’s location reference
3. Select the rows of the table to stage
4. Select a name for the output table
5. Create the output table in the GDW and add columns to hold the datum reference and geometry.
6. Copy each input row to the output table.
7. Create a Datum reference for each row
8. Create a geometric feature for each row.
9. Insert the table information into the GDW metadata.
10. Insert the table information into the LRSx and SDO metadata.
11. Clean up.

The easiest way to create a new staging algorithm is to copy an existing stored procedure and modify it.

4.3 Linear Overlay

Linear overlay operations are part of the business tier’s functions. The actual overlay process is deferred to the LRSx engine which will perform the overlay operation. The LRSx functions will be wrapped with high-level code to make the overlay operations available to the GeoMedia clients and present a standard format API to the rest of the LRS application.

4.4 Business Logic Management

The code that comprises the business logic is strongly coupled to the LRS schema. Because of this, changes that occur in the structure of the LRS schema will probably require modifications to the business logic. To ensure that the business logic and LRS Schema are concurrent, it is important that efforts are made to manage the administration of the business logic and database tiers together.

5 LRS

The LRS is an Oracle schema that contains the tables, relationships, and code that comprises the LRS. The major access to the LRS will be through Intergraph's GeoMedia Pro GIS client for editing and maintenance and from command-line or interactive SQL environments.

5.1 User Interface Components

This section contains a description and processing overview of the LRS maintenance tools that are used to manage the LRS. Much of the data maintenance for the LRS can be performed though interactive SQL environments that do not require the use of a GIS interface. However, some of the maintenance workflows specifically require the use of an interactive map environment.

5.1.1 Interactive SQL Tools

Most of the LRS maintenance operations require updating rows in tables, adding new rows to existing tables and removing rows from tables. These types of operations can be easily performed using interactive SQL tools or using the default Table View interface of GeoMedia Pro. There are several options available for these types of maintenance operations:

- Custom forms and wizards developed within the GeoMedia Pro environment
- Form and wizard based GUIs developed using Java or Visual Basic.
- Oracle Forms and interactive environments.
- HTML-based GUIs

The GeoMedia-based forms and wizards will limit the use of the maintenance tools to expensive GeoMedia Pro seats. The main advantage of this approach is that a single environment that encompasses the entire set of maintenance workflows can be developed and used for maintaining the LRS. Custom forms and wizards created using Java or Visual Basic will be more cost effective to develop and can be used to perform maintenance functions on the LRS without the overhead of an expensive GeoMedia Pro license. Less sophisticated forms can be developed using Oracle Forms or other similar type of environments. Lastly, HTML-based GUIs can be developed to perform form and wizard based user interfaces that will run on any system connected to the LRS.

5.1.2 GIS-based Editing Tools

There are several maintenance operations that require interactive map-based tools that will be provided as part of Intergraph's GeoMedia Pro-based Network Manager application. The following workflows require use of the conflation editing tools in the Network Manager product:

- Conflating an Anchor Section to the Centerline layer.
- Conflating a Network Link to the Anchor Sections layer.
- Conflating a Project Section to the Anchor Sections layer.
- Conflating a HPMS and GIMS section to the Anchor Sections layer.

The interactive GIS interface can also assist in the following:

- Planning datum entities.
- Positioning a Reference Post
- Adding rows to the Route Link table.
- Defining a Transport System.
- Positioning Project Stations

This last set of workflows can be managed without any specialized editing tools within GeoMedia Pro but would require manual updating of some of the components from a SQL utility.

5.2 Metadata

LRS Metadata stores information about the state of the LRS, its component interoperability, and contact information for LRS component managers, support staff, and users. This data exists in three tables in the LRS Schema.

5.3 Transaction Management

Changes made to subsystem tables must be coordinated to ensure that the editing session is completed and leaves the system in a known stable state. There are two types of transaction management that must be performed within the system. The first type, termed System Transactions, pertain to the physical mechanics of performing an edit on a row in a table. The second type, termed Application Transactions, pertains to the logical effects associated with performing an edit within the system itself and has a wider context that must be managed.

5.3.1 System Transactions

Simple edits to rows in a LRS table must be managed to ensure that the system is stable. Managing these rows requires that the following capabilities are maintained by the system

- Rollback capability to ensure that erroneous edits can be removed.
- Row-level locking to allow multiple concurrent editing sessions on the same tables

- Fail-safe to allow edits to be re-created if the system crashes during an editing session.

These mechanical transaction management requirements are met by the capability of Oracle's fail-safe architecture. Additional transaction capability must also be provided. This capability, called Long Term Transactions (LTT), allows the rollback of editing sessions that can occur over the span of days or weeks so that, for example, a weeks worth of editing or more can be easily removed without de-stabilizing the system. This LTT capability is provided by both Oracle as part of its Enterprise Architecture and by Intergraph Corp. in its GeoMedia Pro 4.0 product.

5.3.2 Application Transactions

Application Transactions correspond to the sets of system transactions that make up the workflows that define the steps necessary to perform routine maintenance operations. The re-alignment of an existing anchor section is one such example. These application transactions are defined within the context of the LRS application, and more specifically associated with each management subsystem in the LRS.

Application Transactions are associated with improvement projects and with changes of state within those improvement projects (a state for a roadway improvement project is a phase, such as planning, design, or construction). The changes to the LRS tables associated with any improvement project are managed by creating an improvement project state and storing the improvement project state identifier with all records in all LRS tables that correspond to that state.

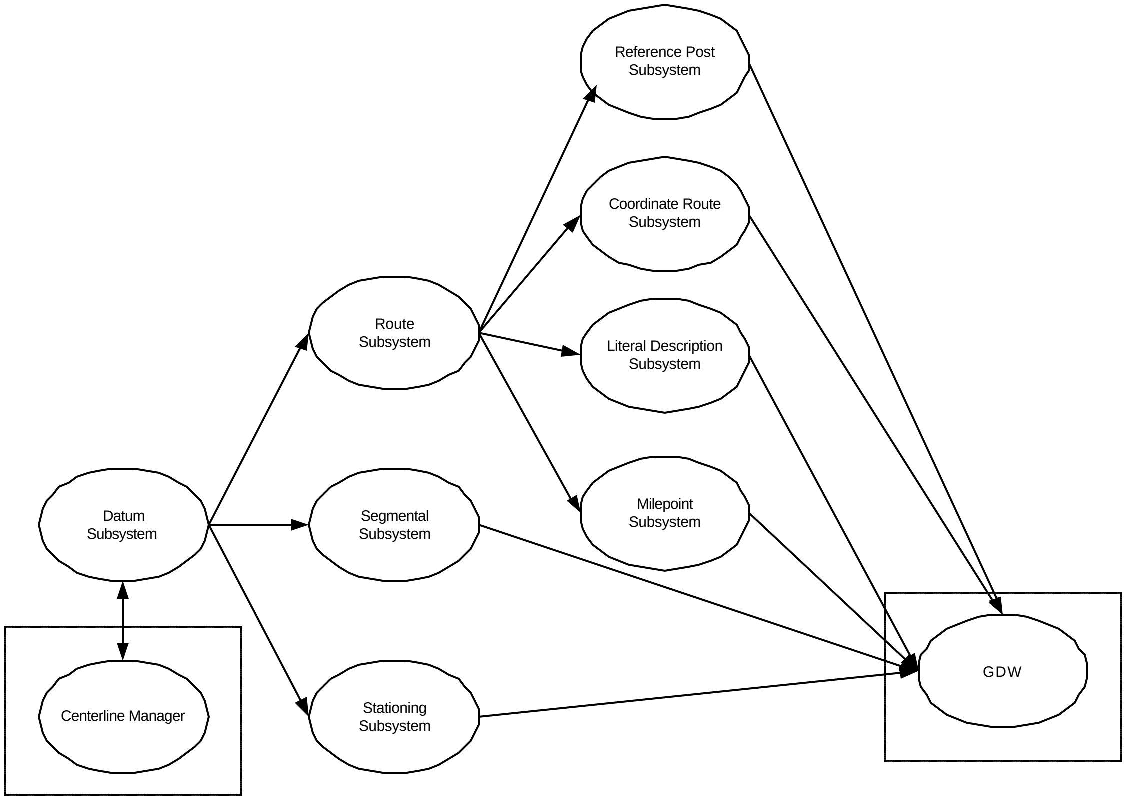
5.4 Events

Many of the operations detailed in this section affect the state of the system and require that the change of state is managed through the layers of the LRS. These changes in state are managed as a set of events. An event is a dynamic object that is raised by one subsystem in response to a change in that system and that has cascading effects throughout one or more other subsystems. Appendix B details the set of events that can be raised.

5.4.1 Event Propagation

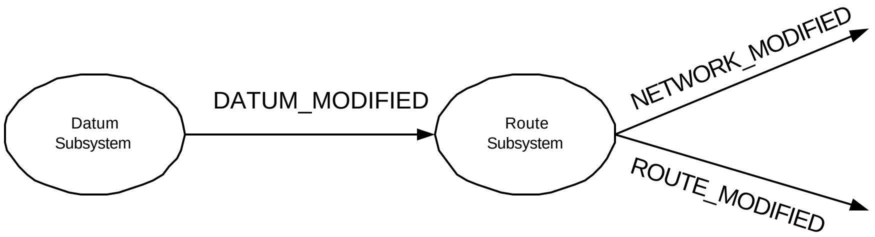
Events are propagated throughout the system from lower layers to higher layers. The event propagation chain is shown in Figure 15.

Figure 15 - Event Propagation

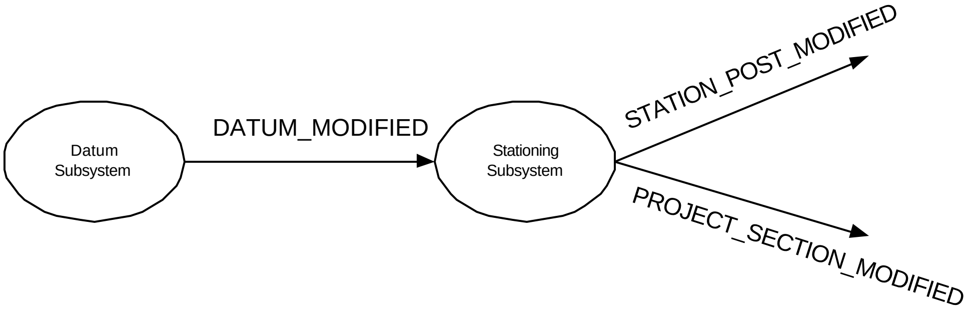


Note that the Centerline manager is not a formal subsystem but has to respond to *requests* for changes from the Datum Subsystem and, once a request has been implemented has to receive notification that changes to the centerlines have been made. Similarly, the GDW is not a formal subsystem but acts as an event consumer. The LRM subsystems do not propagate events among themselves. However, they provide notification that changes to business data stored in the GDW may need to be made.

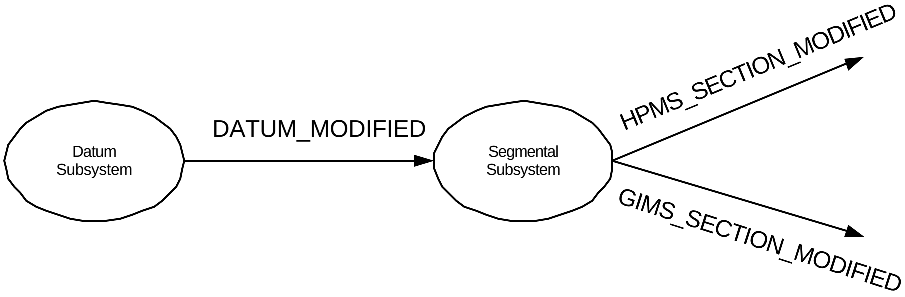
As the Datum Subsystem is the lowest order system in the LRS it does not consume events. Events posted by the Datum subsystem are consumed by the Route, Segmental, and Stationing subsystems.



The Route Subsystem is responsible for evaluating the DATUM_MODIFIED event and posting NETWORK_MODIFIED and ROUTE_MODIFIED events for any changes that need to be propagated.



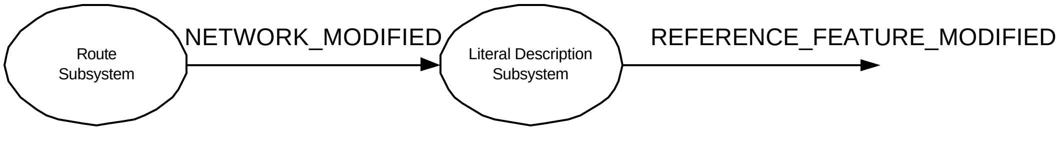
The Stationing Subsystem is responsible for evaluating the DATUM_MODIFIED event and posting PROJECT_SECTION_MODIFIED and STATION_POST_MODIFIED events for any changes that need to be propagated.



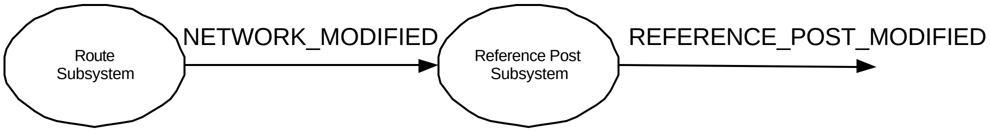
The Segmental Subsystem is responsible for evaluating the DATUM_MODIFIED event and posting HPMS_SECTION_MODIFIED and GIMS_SECTION_MODIFIED events for any changes that need to be propagated.

The Route Subsystem is the second order system in the LRS and only consumes events that have been posted by the Datum Subsystem. Events posted by the Route Subsystem are consumed by the Reference Post, Milepoint, Coordinate Route and Literal Description subsystems.

Changes to the structure of the transportation layers, from either modification to the network or by datum changes being propagated, result in NETWORK_MODIFIED events being sent to the Reference Post subsystem and to the Literal Description subsystems.

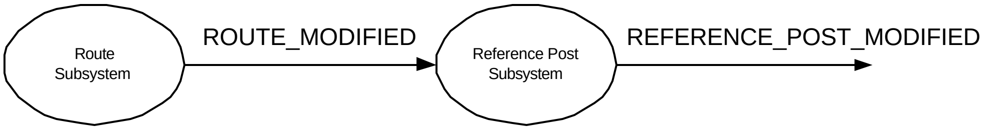


The Literal Description Subsystem must evaluate if any nodes which are also reference features were changed and, if so, propagate a REFERENCE_FEATURE_MODIFIED event.

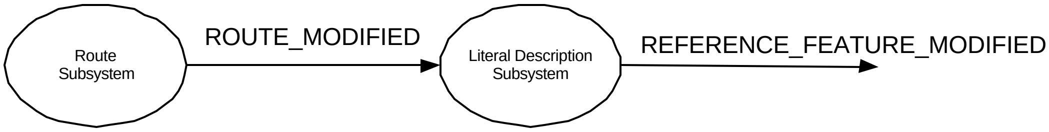


If the modified network affects any transport links which have reference posts associated with them the changes must be evaluated by the Reference Post Subsystem and any causative changes to reference posts propagated as REFERENCE_POST_MODIFIED events.

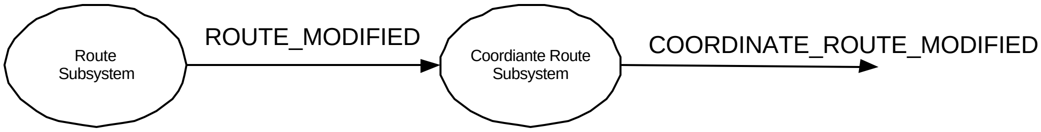
Changes in the definition of one or more routes result in ROUTE_MODIFIED changes which are posted to a number of higher-order subsystem.



The Reference Post Subsystem consumes and evaluates ROUTE_MODIFIED events and propagates those events as REFERFENCE_POST_MODIFIED events if one or ore reference posts are affected by the change.



The Literal Description Subsystem consumes and evaluates ROUTE_MODIFIED events and propagates those events as LIETRAL_DESCRIPTION_MODIFIED events if one or more literal description records are affected by the change. These changes for the Literal Description can occur for changes in reference features in the REFERENCE_FEATURE table and nodes which are also reference features.



The Coordinate Route Subsystem consumes and evaluates ROUTE_MODIFIED events and propagates those events as LITERAL_DESCRIPTION_MODIFIED events if one or more coordinate route records are affected by the change.

5.5 Subsystem Maintenance

This section defines the key operations necessary to maintain the various subsystems in the LRS.

5.5.1 Datum Subsystem

The Datum Subsystem consists of the following tables.

- ANCHOR_POINT
- ANCHOR_POINT_MONUMENT
- ANCHOR_POINT_ELEVATION
- ANCHOR_POINT_CENTERLINE_CONTROL
- ANCHOR_SECTION
- ANCHOR_SECTION_HISTORY
- CENTERLINE

The following key maintenance operations can be performed on the elements of the datum.

- Position a new anchor point
- Position a new anchor section
- Adjust the datum.

Adjusting the datum can take the form of one of the following operations.

- Retire an anchor point
- Re-measure existing anchor section
- Re-align existing anchor section

5.5.1.1 Position New Anchor Point

This operation occurs when a new anchor point is to be added to the datum.

5.5.1.2 Reasons for Change

The new anchor point can be added in response to one of the following occurrences:

- The datum is being extended to include additional infrastructure that was not in the original inventory.

- As a result of an improvement project being started or changing state. For example, a new bypass is going to be constructed.

5.5.1.3 Workflow

1. Ensure that the centerline table contains centerline features that represent the proposed anchor section alignment. The Centerline table is outside the scope of the Datum Subsystem (and LRS system) and so will not be detailed herein.

2. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

3. Add a new anchor point to the ANCHOR_POINT table. Assign the new anchor point the IMPROVEMENT_PROJECT_STATE_ID.

4. Populate the anchor point attributes.

5. Project the anchor point's location to the features of the centerline table. Add a record to the ANCHOR_POINT_CENTERLINE_CTL table to store the results of the projection. Store the resulting location of the projection to the GEOMETRY_GIMSCARTO column of the anchor point.

6. Make additional changes to ANCHOR_POINT_MONUMENT and ANCHOR_POINT_ELEVATION table as necessary.

7. Post a DATUM_CHANGED event to the system.

5.5.1.4 Position New Anchor Section

This operation occurs when a new anchor section is to be added to the datum.

5.5.1.4.1 Reasons for Change

The new anchor section can be added in response to one of the following occurrences:

- The datum is being extended to include additional infrastructure that was not in the original inventory.

- As a result of an improvement project being started or changing state. For example, a new bypass is going to be constructed.

5.5.1.4.2 Workflow

1. Ensure that the centerline table contains centerline features that represent the proposed anchor section alignment. The Centerline table is outside the scope of the Datum Subsystem (and LRS system) and so will not be detailed herein.

2. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

3. Add anchor points to the datum where necessary. Assign each new anchor point the IMPROVEMENT_PROJECT_STATE_ID.

4. Add an anchor section record to the anchor section table. Assign the distance value for the datum and set the IMPROVEMENT_PROJECT_STATE_ID of the row. Populate other attributes of the anchor section.

5. Use GeoMedia Pro to create a conflation object that represents the relationship of the anchor section to the centerline table.

6. Post a DATUM_CHANGE event.

5.5.1.5 Retire Existing Anchor Point

This operation occurs when an exiting anchor point is no longer a part of the datum.

5.5.1.5.1 Reasons for Change

The new anchor point can be added in response to one of the following occurrences:

- The anchor point is no longer connected to any anchor sections and is not planned to be connected to any in the future.

5.5.1.5.2 Workflow

1. Verify that the anchor point is not associated with any anchor section that is not retired.

2. Change the DATE_RETIRED to reflect the date that the anchor point was retired.

3. Post a DATUM_MODIFIED event to the system.

5.5.1.6 Re-Measure Existing Anchor Section

This operation occurs when an existing anchor section has been re-measured to provide greater accuracy or remove some original data entry error. Re-measured anchor sections are treated as if the existing anchor section is being retired and replaced by a new anchor section.

5.5.1.6.1 Reasons for Change

The anchor section can be added in response to one of the following occurrences:

- The anchor section had a measurement error.
- The anchor section distance was estimated and has been re-measured to provide a greater degree of accuracy .

5.5.1.6.2 Workflow

1. Create a copy of the old anchor section. Assign the new anchor section a new anchor section ID.
2. Update the distance of the new anchor section.
3. Add a row into the ANCHOR_SECTION_HISTORY table reflecting the change. The old and new start offsets will be 0 and the old and new end offsets will reflect the old and new anchor section lengths respectively. Set the alignment change flag appropriately.
4. Change the DATE_RETIRED attribute of the old anchor section to the date that the measurement was made.
5. Post a DATUM_MODIFIED event to the system.

5.5.1.7 Re-Align Existing Anchor Section

This operation occurs when an existing anchor section is re-aligned. This event results in the old anchor section being retired and a new anchor section being created.

5.5.1.7.1 Reasons for Change

An anchor section can be added in response to one of the following occurrences:

- As a result of an improvement project being started or changing state.
- As a result of an improvement project being completed and the datum being modified to reflect the change.

5.5.1.7.2 Workflow

1. Ensure that the centerline table contains centerline features that represent the new anchor section alignment. The Centerline table is outside the scope of the Datum Subsystem (and LRS system) and so will not be detailed herein.

2. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

3. Add anchor points to the datum where necessary. Assign each new anchor point the IMPROVEMENT_PROJECT_STATE_ID.

4. Add an anchor section record to the anchor section table. Assign the distance value for the datum and set the IMPROVEMENT_PROJECT_STATE_ID of the row. Populate other attributes of the anchor section.

5. Use GeoMedia Pro to create a conflation object that represents the relationship of the anchor section to the centerline table.

6. Use LRSx stored procedure to create a geometric representation of the anchor section from the conflation object.

7. Use a stored procedure to generate a default datum representation of the anchor section from the anchor points at either end of the anchor section.

8. Add a row into the ANCHOR_SECTION_HISTORY table reflecting the change. Set the ALIGNMENT_CHANGE flag to TRUE for the history record.

9. Change the DATE_RETIRED attribute of the old anchor section to the date that the alignment was made.

10. Post a DATUM_MODIFIED event to the system.

5.5.1.8 Publish an Anchor Section

This operation occurs after an anchor section is added or modified. Staging of anchor sections to the GDW is described elsewhere in this document.

5.5.1.8.1 Workflow

1. Use LRSx stored procedure to create a geometric representation of the anchor section from the conflation object.

2. Use a stored procedure to generate a default datum representation of the anchor section from the anchor points at either end of the anchor section.

5.5.2 Route Subsystem

The Route Subsystem consists of the following tables.

- TRANSPORT_LINK
- TRANSPORT_LINK_AS_XREF
- TRANSPORT_LINK_STATUS
- TRANSPORT_NODE
- TRANSPORT_NODE_STATUS
- TRANSPORT_NODE_OFFSET
- NESTED_NETWORK_XREF
- TRANSPORT_SYSTEM_XREF
- TRANSPORT_SYSTEM
- TRANSPORT_MODE
- TRANSPORT_SYSTEM_MILEPOINT
- ROUTE_LINK
- ROUTE
- ROUTE_SYSTEM
- Individual Route System Schema tables.

The following key maintenance operations can be performed on the elements of the Route Subsystem.

- Add a transport node.
- Retire a transport node.
- Add a transport link.
- Retire a transport link.
- Change a link state.

- Change a node state.
- Determine a transport system.
- Add a signed route.
- Retire a signed route.
- Add a Route-Link
- Retire a Route Link
- Determine a Route Link Distance
- Publish a Transport System Route

5.5.2.1 Add New Transport Node

This operation occurs when a new transport node is to be added to the transport system.

5.5.2.1.1 Reasons for Change

The new transport node can be added in response to one of the following occurrences:

- The network is being extended to include additional infrastructure that was not in the original inventory.
- As a result of an improvement project being started or changing state. For example, a new bypass is going to be constructed.

5.5.2.1.2 Workflow

1. Ensure that the datum has been modified and published to include the new anchor sections.
2. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.
3. Add a new transport node row to the TRANSPORT_NODE table.
4. Determine the position over an anchor section that the node occurs. Add a row to the TRANSPORT_NODE_OFFSET reflecting the relative position of the transport node over the datum.

5. Add a row to the TRANSPORT_NODE_STATUS table and set the state of the node to OPEN (0).

6. Post a NETWORK_CHANGED event to the system.

5.5.2.2 Retire Transport Node

This operation occurs when the transportation system is being modified and a node is no longer connected to any existing or planned transport links.

5.5.2.2.1 Reasons for Change

The transport node would be removed in response to one of the following occurrences:

- The network is being modified to reflect changes in the transportation system.

5.5.2.2.2 Workflow

1. Ensure that the node is connected to only retired transport links. If this is not true, the transport node cannot be retired.

2. Change the DATE_RETIRED of the transport node to reflect the date when the change went into effect.

3. Change the NODE_STATE of the node in the TRANSPORT_NODE_STATUS table to closed.

4. Post a NETWORK_MODIFIED event to the system.

5.5.2.3 Position New Transport Link

This operation occurs when a new transport link is to added to the transportation network.

5.5.2.3.1 Reasons for Change

The new transport link can be added in response to one of the following occurrences:

- The network is being extended to include additional infrastructure that was not in the original inventory.

- As a result of an improvement project being started or changing state. For example, a new bypass is going to be constructed.

5.5.2.3.2 Workflow

1. Ensure that the datum has been modified and published to include the new anchor sections.

2. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

3. Add transport nodes to the network where necessary.

4. Add a transport link record to the TRANSPORT_LINK table. Assign the IMPROVEMENT_PROJECT_STATE_ID of the row. Populate other attributes of the transport link row. Make sure that the DIRECTION value represents the traversability of the link with respect to the implied direction of the link in the FROM and TO transport node direction.

5. Use GeoMedia Pro to create a conflation object that represents the relationship of the transport link to the anchor section table.

6. Use a stored procedure to add rows into the TRANSPORT_LINK_AS_XREF table. This data is derived from the Conflation object created in step 10.

7. Use LRSx stored procedure to create a geometric representation of the anchor section from the conflation object.

8. Use a stored procedure to calculate the datum distance for the transport link.

9. Insert a row into the TRANSPORT_LINK_STATUS table and set the state of the link to OPEN.

10. If the link is to be part of one or more named transport systems, add a row for each such transport system into the TRANSPORT_SYSTEM_XREF table.

11. Post a NETWORK_CHANGED event to the system.

5.5.2.4 Retire Existing Transport Link

This operation occurs when a transport link is no longer to be part of the current transport system.

5.5.2.4.1 Reasons for Change

The new transport link can be retired in response to one of the following occurrences:

- The network is being modified to reflect changes in one or more routes.

5.5.2.4.2 Workflow

1. Set the links status to CLOSED in the TRANSPORT_LINK_STATUS table.

2. Check for any rows referencing the transport link in the TRANSPORT_SYSTEM_XREF table. Retire any rows found by setting the DATE_RETIRED to the retirement date of the transport link.

3. Check for any rows referencing the transport link in the ROUTE_LINK table. Retire any rows found by setting the DATE_RETIRED to the retirement date of the transport link.

4. Retire the transport link by setting the DATE_RETIRED value to the retirement date of the link.

5. Post a NETWORK_MODIFIED event to the system.

5.5.2.5 Change a Transport Link's State

This operation occurs when a transport link is about to be re-opened after being closed or, alternatively, the link is about to be closed.

5.5.2.5.1 Reasons for Change

The state of the transport link can be changed in response to one of the following occurrences:

- The network is being modified to reflect changes in one or more routes and the transport link is being permanently retired.

- The transport link is being temporarily closed to reflect local conditions on the network. For example, a part of the network is closed for maintenance.

5.5.2.5.2 Workflow

1. If the link's status in the TRANSPORT_LINK_STATUS table is CLOSED set it to OPEN. If the status is OPEN set it to CLOSED.

2. Post a NETWORK_CHANGED event to the system.

5.5.2.6 Change a Transport Node's State

This operation occurs when a transport node is about to be re-opened after being closed or, alternatively, the node is about to be closed.

5.5.2.6.1 Reasons for Change

The state of the transport node can be changed in response to one of the following occurrences:

- The network is being modified to reflect changes in one or more routes and the transport link is being permanently retired.
- The transport node is being temporarily closed to reflect local conditions on the network. For example, a part of the network is closed for maintenance.

5.5.2.6.2 Workflow

1. If the node's status in the TRANSPORT_NODE_STATUS table is CLOSED set it to OPEN. If the status is OPEN set it to CLOSED.
2. Post a NETWORK_CHANGE event.

5.5.2.7 Determine a Transportation System

This operation occurs when a new transportation system is being defined in the system.

5.5.2.7.1 Reasons for Change

New transportation systems can be defined in response to one or more of the following occurrences:

- A new transportation classification is to be defined.
- A new route is to be added to the system which has multiple milepoint LRMs.

5.5.2.7.2 Workflow

1. Ensure that the compliment of transportation links and nodes have been added to the network to reflect the definition of the transportation system.
2. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS

component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

3. Add a row into the TRANSPORT_SYSTEM table. Assign the IMPROVEMENT_PROJECT_STATE_ID.

4. Identify each transport link in the TRANSPORT_LINK table that comprises the transport system and create a row for each identified transport link in the TRANSPORT_SYSTEM_XREF table.

5. Post a "TRANSPORT_SYSTEM_CHANGE" event.

6. If the transport system has one or more routes defined, continue. Otherwise stop.

7. For each route that is associated with the new transport system perform the following two steps.

8. Select the set of records in the ROUTE_LINK table for the route and sort by ordinal.

9. For each row returned from 7 calculate the accumulated distance by getting the distance value from the associated transport link. Add a row into the TRANSPORT_SYSTEM_MILEPOINT table for each row in the ROUTE_LINK table that was returned.

5.5.2.8 Add a Signed Route

This operation occurs when a new signed route is being defined in the system.

5.5.2.8.1 Reasons for Change

New signed routes can be defined in response to one or more of the following occurrences:

- A new route is to be added to the system.

5.5.2.8.2 Workflow

1. Ensure that the compliment of transportation links and nodes have been added to the network to reflect the definition of the new signed route.

2. Ensure that the transport system for the route is defined.

3. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS

component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

4. Get the name of the schema to add the route to from the ROUTE_SYSTEM table.

5. Add a row into the ROUTE table. Set the route system id and assign a ROUTE_ID to the new row.

6. Add a row to the ASSIGNED_ROUTES table in the schema selected from 3. Set the IMPROVEMENT_PROJECT_STATE_ID to reflect the state of the route.

7. Generate a FULL_NAME for the signed route based on the values added to the new route row in the ASSIGNED_ROUTE table. Populate the FULL_NAME field in the ROUTE table with the name of the route.

8. Based on the IMPROVEMENT_PROJECT_STATE_ID for the new route set the TEMPORAL_STATE_ID of the ROUTE record.

9. Identify the first and each successive transport link in the TRANSPORT_LINK table that comprises the route and create a row for each identified transport link in the ROUTE_LINK table. For each row added to the ROUTE_LINK table add a row to the TRANSPORT_SYSTEM_MILEPOINT table using an accumulated distance value derived from the TRANSPORT_LINK table.

5.5.2.9 Retire a Signed Route

This operation occurs when an existing signed route is being retired in the system.

5.5.2.9.1 Reasons for Change

Signed routes can be retired in response to one or more of the following occurrences:

- The route is no longer part of the route inventory.

5.5.2.9.2 Workflow

1. Get the name of the schema for the route from the ROUTE_SYSTEM table.

2. Get the ROUTE_ID for the route from the ROUTE table.

3. Set the DATE_RETIRED date for the route in the ASSIGNED_ROUTE table in the schema selected in 1.

4. Set the DATE_RETIRED value in the ROUTE table.

5. Post a ROUTE_MODIFIED to the system.

5.5.2.10 Add a Route Link

This operation occurs when an existing signed route is being modified or a new sign route is being created. Additionally, changes in the transport network will cause route links to be retired and new ones created.

5.5.2.10.1 Reasons for Change

Route Links can be added for the following reasons:

- A new signed route is being added.
- An existing signed route is being modified
- The transportation system has changed.

5.5.2.10.2 Workflow

1. Get the name of the schema for the route from the ROUTE_SYSTEM table.

2. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

3. Get the ROUTE_ID for the route from the ROUTE table.

4. Add a row to the ROUTE_LINK table. Populate the row with the TRANSPORT_LINK_ID and DISTANCE attributes from the TRANSPORT_LINK table.

5. Populate the ORDINAL value. If this is an insertion of a section into an existing route ensure that the ordinal values for all records which occur further down the route from the inserted section are correct and that all milepoints in the TRANSPORT_SYSTEM_MILEPOINT table are updated.

6. Insert a row into the TRANSPORT_SYSTEM_MILEPOINT

7. Post a ROUTE_CHANGE event to the system.

5.5.2.11 Retire a Route Link

This operation occurs when an existing signed route is being modified. Additionally, changes in the transport network will cause route links to be retired.

5.5.2.11.1 Reasons for Change

Route Links can be added for the following reasons:

- An existing signed route is being modified
- The transportation system has changed.

5.5.2.11.2 Workflow

1. Get the name of the schema for the route from the ROUTE_SYSTEM table.
2. Get the ROUTE_ID for the route from the ROUTE table.
3. Change the DATE_RETIRED value for the route link row.
4. Change the DATE_RETIRED value for the associated TRANSPORT_SYSTEM_MILEPOINT row.
5. If this is the only change in the route ensure that ordinals in the ROUTE_LINK table are updated and that the milepoint values in the TRANSPORT_SYSTEM_MILEPOINT table are updated.
6. Post a ROUTE_MODIFIED event to the system.

5.5.2.12 Publish Transport System Route

No special operations are required to publish a route. The ROUTE table contains the information necessary to stage the data into the GDW, which is managed by the relevant maintenance processes.

5.5.3 Reference Post Subsystem

The Reference Post Subsystem consists of the following tables.

- REFERENCE_POST

The following key maintenance operations can be performed on the elements of the Reference Post Subsystem.

- Add a Reference Post.

- Re-Position a Reference Post.
- Publish a Reference Post.

5.5.3.1 Re-Position An Existing Reference Post

This operation occurs when an existing reference post is to be re-positioned.

5.5.3.1.1 Reasons for Change

The new reference post can be re-positioned in response to one of the following occurrences:

- As a result of an improvement project being started or changing state. For example, a new bypass is constructed.

5.5.3.1.2 Workflow

1. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

2. If the IMPROVEMENT_PROJECT_STATE_ID is 0 then retire the existing reference post and generate a new reference post and post a REFERENCE_POST_CHANGE event.

3. If the IMPROVEMENT_PROJECT_STATE_ID is not 0 then generate a new reference post.

4. Post a REFERENCE_POST_MODIFIED event to the system.

5.5.3.2 Add New Reference Post

This operation occurs when a new reference post is to be added to the system.

5.5.3.2.1 Reasons for Change

The new reference post can be added in response to one of the following occurrences:

- A new route is being added or an existing route is being extended.
- As a result of an improvement project being started or changing state. For example, a new bypass is constructed.

5.5.3.2.2 Workflow

1. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

2. Ensure that the route associated with the reference post is correctly defined.

3. Add a row to the reference post table. Set the anchor point and anchor point offset values which are collected from the field. Set the improvement project state value, Route ID and other post-specific values.

4. Project the anchor point offset reference onto the transport system and get the transport link associated with the reference post. Use the transport link and route id to get the row in the route link table for the reference post.

5. Post a REFERENCE_POST_CHANGE event.

5.5.3.3 Publish a Reference Post

No special operations are required to publish a reference post. The management of the reference post table is provided by the relevant maintenance operations.

5.5.4 Segmental Subsystem

The Segmental Subsystem consists of the following tables.

- GIMS_SECTION
- HPMS_SECTION

The following key maintenance operations can be performed on the elements of the Segmental Subsystem.

- Position a HPMS or GIMS section
- Publish a HPMS or GIMS section.

5.5.4.1 Position A HPMS or GIMS Section

This operation occurs when an existing HPMS or GIMS section is to be positioned.

5.5.4.1.1 Reasons for Change

The positioning of a section will be in response to one of the following occurrences:

- The datum is being extended to include additional infrastructure that was not in the original inventory.
- As a result of an improvement project being started or changing state. For example, a new bypass is constructed.

5.5.4.1.2 Workflow

1. Ensure that the datum extends over the extent of the section.
2. Use the GeoMedia/Pro conflation editing tools to create a conflation object for the section extent against the anchor sections.
3. Use the LRSx stored procedures to create and store a cartographic representation of the section.
4. Post a GIMS_MODIFIED or HPMS_MODIFIED event to the system.

5.5.4.2 Publish HPMS or GIMS Section

HPMS and GIMS sections are published by using the LRSx stored procedures to generate a cartographic representation of each section during the staging process.

5.5.5 Stationing Subsystem

The Stationing Subsystem consists of the following tables.

- PROJECT_SECTION
- PROJECT_SECTION_NODE
- PROJECT_SECTION_NODE_OFFSET
- STATION_POST

The following key maintenance operations can be performed on the elements of the Stationing Subsystem.

- Position a project section.
- Position a station post.
- Publish a station post
- Publish a project section.

5.5.5.1 Position A Project Section

This operation occurs when an new project section is to be positioned.

5.5.5.1.1 Reasons for Change

The positioning of a project section will be in response to one of the following occurrences:

- As a result of an improvement project being started or changing state. For example, a new bypass is constructed.

5.5.5.1.2 Workflow

1. Ensure that the datum extends over the extent of the project section.

2. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.

3. Ensure that there is a project section node at each end of the project section. If one or both of the nodes are missing add the nodes into the PROJECT_SECTION_NODE table and PROJECT_SECTION_NODE_OFFSET tables with data collected from the field.

4. Add a new project section to the PROJECT_SECTION table.

5. Use the GeoMedia / Pro conflation editing tools to create a conflation object for the project section extent against the anchor sections.

6. Use the LRSx stored procedures to create and store a cartographic representation of the project section.

7. Post a PROJECT_SECTION_CHANGE event to the system.

5.5.5.2 Position A Station Post

This operation occurs when a new station post is to be positioned over a project.

5.5.5.2.1 Reasons for Change

The positioning of a station post will be in response to one of the following occurrences:

- As a result of an improvement project being started or changing state. For example, a new bypass is constructed.

5.5.5.2.2 Workflow

1. Ensure that there is a project section at the position of the station post.
2. Add a new row to the STATION_POST table.
3. Use GeoMedia/Pro to position the station post over the project section and measure the offset from the start of the project section.
4. Use the LRSx stored procedures to create and store a cartographic representation of the station post.
5. Post a STATION_POST_CHANGE event to the system.

5.5.5.3 Publish Station Post and Project Section

Project Sections and Station Posts are published by using the LRSx stored procedures to generate a cartographic representation of each feature during the staging process.

5.5.6 Coordinate Route Subsystem

The Coordinate Route Subsystem does not have any specific tables associated with it..

The following key maintenance operations can be performed on the elements of the Coordinate Route Subsystem.

- Position a point event.
- Position a linear event.

5.5.6.1 Position A Point Event

This operation occurs when a point event such as a vehicle or crash location has to be positioned.

5.5.6.1.1 Reasons for Change

The positioning of a point event can be in response to one of the following occurrences:

- As a result of a phenomena occurring at a discrete location over the extent of the datum. The location of a vehicle at a specific time for example.

5.5.6.1.2 Workflow

1. Ensure that the coordinates for the phenomenon are in the correct coordinate system for the LRS.

2. Use the LRSx stored procedures to project the point to one of the cartographic representations of the datum (we will only have one initially).

3. Store the anchor section and offset for the phenomenon if the projection was successful.

5.5.6.2 Position A Linear Event

This operation occurs when a event such as a snow-plough route has to be positioned.

5.5.6.2.1 Reasons for Change

The positioning of a linear event can be in response to one of the following occurrences:

- As a result of a phenomena occurring over a linear extent of the datum. The route driven by a snow plough for example.

5.5.6.2.2 Workflow

1. Ensure that the coordinate stream for the phenomenon are in the correct coordinate system for the LRS.

2. Use the LRSx stored procedures to project the end-points of the coordinate stream to one of the cartographic representations of the datum (we will only have one initially).

3. Store the anchor section and offset for the end-points of the phenomenon if the projection was successful.

4. Transform the anchor-section offsets for the end-points of the coordinate stream to the Milepoint LRM for the given route.

5. Create a conflation object against the anchor sections that defines the traversal through the datum from the start to the end datum reference for the route.

5.5.7 Literal Description Subsystem

The Literal Description Subsystem consists of the following tables.

- REFERENCE_FEATURE_CATEGORY
- REFERENCE_FEATURE
- REFERENCE_FEATURE_ROUTE_XREF
- TRANSPORT_NODE_ROUTE_XREF

The following key maintenance operations can be performed on the elements of the Literal Description Subsystem.

- Establish (position) a Reference Feature.
- Retire a Reference Feature
- Publish a Reference Feature.

5.5.7.1 Position A Non-Intersection Reference Feature

This operation occurs when a new reference feature (such as a bridge or railroad crossing) is to be positioned.

5.5.7.1.1 Reasons for Change

The positioning of a reference feature will be in response to one of the following occurrences:

- The datum is being extended to include additional infrastructure that was not in the original inventory.
- As a result of an improvement project being started or changing state. For example, a new bypass is constructed.

5.5.7.1.2 Workflow

1. Ensure that there are one or more routes defined at the location of the reference feature.
2. Add a new row to the REFERENCE_FEATURE table.
3. If the change is a result of a change in the state of an improvement project, manage the improvement project tables and get an IMPROVEMENT_PROJECT_STATE_ID value for the operation. If the change is a result of a change in the active (or retired) LRS component, set the IMPROVEMENT_PROJECT_STATE_ID value to 0.
4. Use GeoMedia / Pro to position the reference feature over the transportation link and measure the offset from the start of the transportation link to the reference feature.
5. Add a row into the REF_FEATURE_ROUTE_XREF table for each route that is associated with the reference feature.
6. Post a REFERENCE_FEATURE_CHANGE event to the system.

5.5.7.2 Retire A Reference Feature

This operation occurs when an existing reference feature is to be retired.

5.5.7.2.1 Reasons for Change

The reference feature can be retired in response to one of the following occurrences:

- The physical entity that is associated with the reference feature is no longer there (railroad crossing is removed).
- The physical entity that is associated with a reference feature has had its reference feature name changed (i.e., the railroad crossing 1234 is now railroad crossing 2345).

5.5.7.2.2 Workflow

1. Set the DATE_RETIRED value for the reference feature to the date reflecting the change.
2. Post a REFERENCE_FEATURE_MODIFIED event to the system.

5.5.7.3 Position An Intersection Reference Feature

This operation occurs when a transportation node is to be used as a reference feature.

5.5.7.3.1 Reasons for Change

The positioning of a transportation reference feature will be in response to one of the following occurrences:

- The datum is being extended to include additional infrastructure that was not in the original inventory.
- As a result of an improvement project being started or changing state. For example, a new bypass is constructed.

5.5.7.3.2 Workflow

1. Ensure that the transportation node is positioned.
2. Ensure that there are one or more routes defined at the location of the transport node.
3. For every ON / AT combination of routes which are incident at the transport node add a row into the TRANSPORT_NODE_ROUTE_XREF table.
4. Post a REFERENCE_FEATURE_CHANGE event to the system.

5.5.7.4 Publish a Reference Feature

Reference features do not need to be published. Business data associated with a reference feature such as a bridge or rail-road crossing will be staged separately. Transport Nodes which are reference features are staged as part of the network.

6 ORGANIZATIONAL SPECIFICATIONS

The Iowa DOT must assign LRS subsystem management responsibility to specific DOT offices. The LRS subsystems are Datum, Route, LRS Milepoint, Reference Post, Segmental, Stationing, Coordinate Route, and Literal Description. There are two levels of responsibility for each particular LRS subsystem that must be assigned: individual management processes within a subsystem and the overall management of the subsystem. The Physical Design Team produced recommended office assignments for both levels for all subsystems.

The individual process assignments are itemized in Table 8. The processes are based on the logical design, but modified to accommodate the physical design. The Physical Design Team only analyzed Location Reference Maintenance processes from the Logical Design (see the Logical Design Process Model). The Physical Design Team did not analyze Location Reference Administration, Location Reference Operations, or Location Reference System Monitoring because these are outside the scope of the current project. The Team assigned responsibility to an office based on these office characteristics: understanding of LRS requirements, current skills and workflows related to these requirements, and technology capability.

The Physical Design Team then determined the overall management assignment based on several factors. A primary factor was whether an office is currently assigned a similar individual subsystem task. Another factor was recommended groupings of subsystems from the logical design. Another important consideration was an office’s proactive involvement in the current LRS efforts. Table 9 shows the results of this analysis. In both tables, an ‘X’ indicates who would be ultimately responsible for the overall success of an LRS subsystem.

From the Logical Design Summary Document a key governing body for the LRS is the LRS Operations Work Team. That document proposed its members be the LRS Subsystem Managers and chaired by the LRS Manager. Given Table 9, the LRS Operations Work Team will be composed of managers from Office of Transportation Data, Office of Design, Office of Maintenance, Research Management Division, and the GIS Team in the Information Technology Division as LRS Manager.

The Logical Design Summary Document also describes the requirements for the LRS Board of Directors. This body would provide direction to the LRS Operations Work Team based on DOT priorities and objectives. The LRS Board of Directors would be composed of managers from each DOT Division. The Iowa DOT Divisions, working with the current LRS Steering Team members, should determine appropriate membership. The LRS Board of Directors must be advocates for LRS within their respective Divisions. The current LRS Sponsor and the LRS Steering Team should determine the new LRS Sponsor, who will then chair the LRS Board of Directors.

Some general statements regarding the assignment of LRS management are provided here. First, only responsibilities for production processes are described in these sections. Initial data creation and loading are one-time processes not included here. Second, some of the existing workflow procedures and corresponding technologies do not support the LRS accuracy requirements determined during the Needs Assessment. This was expected, and the pilot phase of this project will reveal additional methods and technologies that existing workflows can adopt over time to meet these accuracy requirements.

The third general assignment regards LRS data publishing. Data publishing includes notifying LRS subsystem users of data changes and staging new versions of LRS subsystem data to the GeoData Warehouse (GDW). While LRS subsystem managers will notify users of data changes, the GDW Manager will initially perform all GDW data staging processes. This assignment will continue until the processes and procedures stabilize. Table 8 indicates who would perform these processes.

Finally, the Logical Design Summary Document outlines the necessary skill and knowledge requirements to manage each subsystem. That document identified general skills and knowledge that all offices should possess: GIS-T and relational database concepts and practices, business process and quality improvement, interoperable architecture design, and marketing within government agencies. Please see the Logical Design Summary Document for individual subsystem skill definitions.

Table 8 - Organization Assignments to LRS Processes

X = Perform this process; S = Support those who perform this process; I = Initial performer (X), but will move to support (S) or not be involved; H = Performs this process but for historic data.

Elementary Process	Trans Data - Sys Mon	Trans Data - Sys Mgmt	IT Division - GIS Team	IT Division - PMIS Support	IT Division - Oracle DBA	Office of Maintenance	Office of Traffic Engineering and Safety	District Offices	Office of Design	Office of Construction	Office of Bridges and Structures	Office of Rail Transportation	DOT Div Directors	Division Managers	GDW Manager	LRS Board of Directors	LRS Working Team	Office of Driver Services (MV)	Research Management Division	Comments
Establish LRS Controls																	**X**			
Establish LRS Parameters																	**X**			
Determine Transport System		**S**											**S**	**X**		**S**	**S**			
Determine First Order Datum Design		**X**	**I**																	
Determine Second Order Datum Design		**X**	**I**																	
Determine Third Order Datum Design		**X**	**I**																	
Plan Linear Datum Survey (P)	**X**	**S**	**I**																	
Prepare Measurement Device (P)	**X**	**S**																		
Place Anchor Point	**X**	**S**																		

Elementary Process	Trans Data - Sys Mon	Trans Data - Sys Mgmt	IT Division - GIS Team	IT Division - PMIS Support	IT Division - Oracle DBA	Office of Maintenance	Office of Traffic Engineering and Safety	District Offices	Office of Design	Office of Construction	Office of Bridges and Structures	Office of Rail Transportation	DOT Div Directors	Division Managers	GDW Manager	LRS Board of Directors	LRS Working Team	Office of Driver Services (MV)	Research Management Division	Comments
Monument Anchor Point	**S**	**S**				**S**		**X**	**S**	**S**										
Measure Anchor Point Span	**X**	**S**																		
Determine Anchor Section Distance		**X**	**I**																	
Adjust Linear Datum		**X**	**I**																	
Position Anchor Point (includes centerline)		**X**																		
Position Anchor Section (includes centerline)		**X**																		
Publish Linear Datum		**X**	**S**		**S**										**I**					
Add Transport Node	**S**	**X**																		
Add Transport Link	**S**	**X**																		
Change Link State	**S**	**X**																		
Change Node State	**S**	**X**																		
Remove Transport Link		**X**																		
Remove Transport Node		**X**																		

Elementary Process	Trans Data - Sys Mon	Trans Data - Sys Mgmt	IT Division - GIS Team	IT Division - PMIS Support	IT Division - Oracle DBA	Office of Maintenance	Office of Traffic Engineering and Safety	District Offices	Office of Design	Office of Construction	Office of Bridges and Structures	Office of Rail Transportation	DOT Div Directors	Division Managers	GDW Manager	LRS Board of Directors	LRS Working Team	Office of Driver Services (MV)	Research Management Division	Comments
Assign Route Name		**X**																**S**		MV to assist with Unofficial routes.
Add Route Link		**X**																**S**		MV to assist with Unofficial routes.
Remove Route Link		**X**																**S**		MV to assist with Unofficial routes.
Remove Signed Route		**X**																**S**		MV to assist with Unofficial routes.
Determine Route Link Distance		**X**																**S**		MV to assist with Unofficial routes.
Publish Transport System Route		**X**			**S**										**I**					
Place Reference Post				**S**		**S**	**S**	**X**												Placed during construction / maintenance.
Position Reference Post	**S**	**S**		**X**		**S**														Trans Data only because videolog collection and GIMS milepoint used to currently support RP positioning.
Publish Reference Post				**X**	**S**	**S**									**I**					
Place Station Marker (Post)						**S**		**X**	**S**	**S**										
Determine Project Section Location								**H**	**X**											

Elementary Process	Trans Data - Sys Mon	Trans Data - Sys Mgmt	IT Division - GIS Team	IT Division - PMIS Support	IT Division - Oracle DBA	Office of Maintenance	Office of Traffic Engineering and Safety	District Offices	Office of Design	Office of Construction	Office of Bridges and Structures	Office of Rail Transportation	DOT Div Directors	Division Managers	GDW Manager	LRS Board of Directors	LRS Working Team	Office of Driver Services (MV)	Research Management Division	Comments
Position Project Section								**H**	**X**											
Position Station (Marker) Post								**H**	**X**											
Publish Project Section					**S**			**H**	**X**						**I**					
Publish Station (Marker) Post					**S**			**H**	**X**						**I**					
Establish Reference Feature (Bridge or RR Crossing)		**S**									**X**	**X**								
Publish Reference Feature		**X**			**S**										**I**					
Determine HPMS Section Location		**X**																		
Position HPMS Section		**X**																		
Publish HPMS Section Location		**X**			**S**										**I**					
Determine Transportation Feature Location (Bridge or RR Crossing)											**X**	**X**								Part of business data maintenance, not part of LRS.

Elementary Process	Trans Data - Sys Mon	Trans Data - Sys Mgmt	IT Division - GIS Team	IT Division - PMIS Support	IT Division - Oracle DBA	Office of Maintenance	Office of Traffic Engineering and Safety	District Offices	Office of Design	Office of Construction	Office of Bridges and Structures	Office of Rail Transportation	DOT Div Directors	Division Managers	GDW Manager	LRS Board of Directors	LRS Working Team	Office of Driver Services (MV)	Research Management Division	Comments
Position Transportation Feature (Bridge or RR Crossing)											**X**	**X**								Part of business data maintenance, not part of LRS.
Publish Transportation Feature Location (Bridge or RR Crossing)											**X**	**X**			**I**					Part of business data maintenance, not part of LRS.
Determine GIMS Segment Extents		**X**																		Part of business data maintenance, not part of LRS.
Position GIMS Segment		**X**																		
Publish GIMS Segment Location		**X**			**S**										**I**					Part of business data maintenance, not part of LRS.

Table 9 - Organization Assigments to LRS Subsystem Management

X = Perform this process; S = Support those who perform this process; I = Initial performer (X), but will move to support (S) or not be involved; H = Performs this process but for historic data. The heavy lines groupings of subsystems based on data, process, and technology similarities.

Elementary Process	Trans Data - Sys Mon	Trans Data - Sys Mgmt	IT Division - GIS Team	IT Division - PMIS Support	IT Division - Oracle DBA	Office of Maintenance	Office of Traffic Engineering and Safety	District Offices	Office of Design	Office of Construction	Office of Bridge and Structures	Office of Rail Transportation	DOT Div Directors	Division Managers	GDW Manager	LRS Board of Directors	LRS Working Team	Office of Driver Services (MVD)	Research Management Division	Comments
LRS Manager			**X**																	
Datum Subsystem Manager	**S**	**X**	**I**			**S**		**S**	**S**	**S**										
Centerline Data		**X**																		
Stationing Subsystem Manager		**S**				**S**		**S**	**X**	**S**										
Route Subsystem Manager	**S**	**X**																**S**		
Individual Route Systems	**S**	**X**																**S**		
Milepoint Subsystem Manager		**X**																**S**		

Elementary Process	Trans Data - Sys Mon	Trans Data - Sys Mgmt	IT Division - GIS Team	IT Division - PMIS Support	IT Division - Oracle DBA	Office of Maintenance	Office of Traffic Engineering and Safety	District Offices	Office of Design	Office of Construction	Office of Bridge and Structures	Office of Rail Transportation	DOT Div Directors	Division Managers	GDW Manager	LRS Board of Directors	LRS Working Team	Office of Driver Services (MVD)	Research Management Division	Comments
Literal Description Subsystem Manager		**X**									**S**	**S**						**S**		
Coordinate Route Subsystem Manager																			**I**	Need an X (Research to determine)
Reference Post Subsystem Manager	**S**	**S**		**S**		**X**	**S**	**S**	**S**											
Segmental Subsystem Manager		**X**																		
GDW Manager			**X**		**S**															

6.1 Office of Transportation Data, System Management

The System Management Section should be responsible for managing and maintaining centerline data and several LRS subsystems: Datum, Route, Segmental, Literal Description, and Milepoint. System Management has the strongest competencies at Iowa DOT in data maintenance for linear and spatially referenced transportation data. Their current workflows include data maintenance for all Iowa public roadways.

System Management currently maintains and should continue to maintain the cartographic centerline data for Iowa DOT. System Management will be enhancing the cartography for GIS purposes (topologically structured, divided roadways additions, and ramp geometry additions). The results of these enhancements should be used as the centerline data for the LRS. The long-term strategy is to have the DOT cartography independent of the GIMS Segmental LRM and tied directly to the datum. Therefore, the centerline maintenance will require maintaining the new relationship between the cartography and the datum.

For Datum Subsystem management, System Management should manage new data at Iowa DOT. The management processes will include datum design, datum collection, datum database maintenance, and datum publishing. All but publishing are briefly described here. Datum design means determining where anchor points and sections will be located. Initially, the process would be performed under the guidance of the GIS Team (see Information Technology Division, GIS Team).

System Management would collect datum data when field collection methods are not used (see the Office of Transportation Data, System Monitoring for field collection responsibilities). The exact collection methods (and related technology) and collection extent is not known at this time (the results of the field pilot portion of the project will guide these decisions). Datum database maintenance means making changes to database records. Datum management will require an increased effort, especially to meet the temporal data requirements of the LRS (data must be available prior to the road opening to traffic).

For Route Subsystem management, System Management would manage new data at Iowa DOT and also enhance existing data already managed by System Management. Transport Links and Nodes (the network) and their relationship to the datum will be new data. The design requires basic traffic direction data to be directly integrated with the Transport Links. System Management maintains some of the required traffic direction data in GIMS, which must be made part of the Route Subsystem. Enhancements to this data are required before it can be fully utilized in the Route Subsystem. Any information that currently resides in GIMS could then be automatically updated from the network data using LRS transform and overlay tools. These network maintenance activities will require an increase in effort. The cartographic centerline topologic processing should be leveraged to help reduce this effort.

The Transport System and LRS Milepoint data will also be new, but the staffing requirements for these will be minimal. Transport Systems change infrequently after the initial load. Milepoint data is derived using automated processes.

Another component of Route Subsystem management is route name maintenance. Route name maintenance is a current System Management activity done in GIMS. However, the LRS requires more rigorous name management. The unique list of route names will be managed in tables. Route names must be partitioned into prefix, name, type and suffix (e.g., North, Maple, Avenue, Southwest), and must be assigned to a geography and a jurisdiction.

A final component of Route Subsystem management is mapping names to roadway segments to create routes. System Management currently maps route names to GIMS segments and orders the segments for routing purposes on the primary roadways. This activity is still required, but the LRS needs to map routes to Transport Links (the result is called Route Links) for all public roadways. The route names on GIMS segments can subsequently be derived using the LRS transform tools.

Another LRS subsystem for which System Management should be responsible is the Literal Description Subsystem. System Management already maintains a literal description on the ends of the GIMS segments for city streets, but only limited descriptions on other road systems. The LRS Literal Description is derived from LRS data using Route Subsystem data and automated processes. The LRS Literal Description LRM can be used to automatically populate the descriptions on the ends of the GIMS segments using the LRS transform tools. Some, but not all, of the effort used to maintain the LRS data should come from the effort that was used to tag the GIMS segments.

System Management should be responsible for running the automated process for loading literal descriptions on roadway intersections (see the Office of Bridge and Structures and the Office of Rail Transportation for responsibilities for non-intersection features used in the Literal Description LRS). Automating the GIMS end point naming task is expected to save some staff effort that should be allocated to new route management processes within System Management.

Finally, System Management should continue to perform HPMS and GIMS segment maintenance. These maintenance efforts will need to include new processes of mapping these segments to the datum.

In summary, System Management should be responsible for 5 subsystems and the centerline data. These responsibilities include new and enhanced data maintenance activities that will require an increase in staff resources. This is especially true when maintenance activities are first implemented. Iowa DOT should explore creating the LRS in phases if additional staffing will be minimal.

System Management staff would require additional training and education. Knowledge of LRS and linear datum concepts and theories would be necessary for all staff involved in the LRS subsystem maintenance. Some special skill sets include geodesy or surveying for the Datum Subsystem, and network analysis, routing, and geocoding for the Route Subsystem.

From a technology perspective, System Management is already targeting technologies that are the same as the LRS technology architecture. They have recently migrated to Oracle, are considering GeoMedia Pro as a client interface, and are willing to explore Oracle Spatial as a spatial data base storage mechanism.

6.2 Office of Transportation Data, System Monitoring

The System Monitoring Section should be responsible for planning and executing field collection of the datum data (anchor point locations, anchor point monuments when necessary and anchor section distances) and network data (transport node locations, transport link directions). System Monitoring has a well-established data collection competency at Iowa DOT. Such skills are required for these critical components of the LRS. The exact collection methods (and related technology) and collection extent of the transportation network are not known at this time (the results of the field pilot portion of the project will guide these decisions).

System Monitoring staff would need to acquire basic knowledge and skills in linear datum theory and concepts, applying business rules for locating anchor points, identifying anchor point monument needs, determining anchor section distance, and determining transport node locations.

System Monitoring would need GeoMedia Pro technology. They will continue to supply videolog data to aid in positioning reference posts until the Office of Maintenance applies more accurate location methods.

6.3 Information Technology Division, GIS Team

The Information Technology Division, GIS Team, should be initially responsible for datum design and datum distance calculation. The GIS Team should be responsible for the processes, procedures and datum stability, while the Office of Transportation Data should collect and maintain the datum data. The reason for this assignment is based on experience. The GIS Team has been directly involved in the development of the LRS and is most familiar with the Datum Subsystem. Because the datum is the foundation of the LRS, it is critical that past datum knowledge be used directly in the implementation of the datum. As the procedures become stable and the knowledge transfer is completed, the responsibility should be transferred to the Office of Transportation Data.

The GIS Team, more specifically the GIS Coordinator, should also take on the LRS Manager responsibility. The LRS Manager role has informally existed at DOT for some time and is currently filled by the GIS Coordinator who is part of the GIS Team. The GIS Coordinator is also the Co-Project Manager for this project. The Coordinator has detailed knowledge about this project, the Iowa DOT LRS vision, strategy, and immediate next steps. This role will demand significant time as the LRS continues into prototyping and implementation. The LRS Manager will need to establish the organizational architecture (Logical Design Summary Document) and heavily promote the LRS within, and external to, the DOT. The GIS Coordinator already has been marketing and informing DOT staff about the LRS. The LRS Manager also manages modifications to the business logic of the LRS. The GIS Coordinator has an in-depth understanding of the business logic components. Finally, assigning this role to the GIS Coordinator also disseminates LRS responsibilities among the different offices involved in LRS management.

The GIS Team also already fulfills the responsibilities of GDW Manager. To assign the LRS component of the GDW to the GIS Team is a logical extension of the current role. They possess in depth knowledge of both the LRS and the GDW. In addition, this assignment helps disseminate LRS responsibilities among the different offices involved in LRS management.

The GIS Team would need to expand its knowledge and skills in data warehousing. Due to existing resource limitations, it is anticipated that additional staff would be required for the GIS Team. The GIS Team already possesses GeoMedia Pro and Oracle technology.

6.4 Information Technology Division, PMIS Support

PMIS Support currently identifies reference post maintenance needs via videolog technologies and supplies that information to the District Offices for placing reference posts in the field. PMIS Support also creates LRS positions for the reference posts in the database. The Physical Design Team recommends that Iowa DOT consider migrating these tasks to the Office of Maintenance. Please see Office of Maintenance in this section for more information.

6.5 Engineering Bureau, Office of Design

The Office of Design should manage the Stationing Subsystem. Management of this subsystem includes the initial data entry and maintenance for the stationing LRM (entering historic projects would be handled by the District Offices). The Office staff possesses a very strong competency in stationing practices and has significant knowledge about improvement project details. The staff is centralized for better quality control. The Office is also responsible for making any alignment modifications discovered during construction, and therefore possesses first-hand knowledge of as-built information. It is not clear at this time what staffing would be required to support the Subsystem, especially if the data is entered in the LRS during the centerline design task.

The Office would need to acquire a working knowledge of the LRS datum and most likely would need to acquire GeoMedia Pro and Oracle technology.

The Office of Design should also continue to provide a support role for the initial placement of station markers (posts or stamps) due to its responsibility to assign these locations during roadway design. The Office of Design should also support the initial placement of anchor point monuments.

6.6 Engineering Bureau, Office of Bridges and Structures

The Office of Bridges and Structures would be most effective at populating the necessary bridge information in the Literal Description Reference Feature tables. The Office is responsible for DOT bridge data and the location of bridges. They have first-hand knowledge of bridge changes and will be responsible for locating bridges using one of DOT's standard LRMs. The Office is also currently moving the bridge data for the primary road system into Oracle.

The DOT can automate the process to populate the LRS tables from the bridge database. This will help minimize additional staffing needs. The Office may need to acquire GeoMedia Pro to enhance the maintenance process. Office staff will need working knowledge and skills on datum and literal description theories and concepts.

The primary impacts to this Office would most likely be the need to meet LRS requirements for choosing points of reference and to meet LRS accuracy and temporal requirements. An example LRS point of reference for a bridge is the southern and then westernmost expansion joint. This point of reference would be used to locate features along the road that the bridge supports (versus roadways the bridge goes over). The point of reference distance from an anchor point must possess accuracy within 3-5 meters. The LRS requirements regarding timely updates suggest that database changes to bridge records must be made as soon as possible. These changes are required prior to the bridge opening to traffic. If the accuracy and temporal requirements cannot be met initially, actions need to be taken to plan and implement them in the future.

6.7 Modal Division, Office of Rail Transportation

The Office of Rail Transportation would be most effective at populating the necessary railroad crossing information in the Literal Description Reference Feature tables. The Office is currently responsible for railroad crossing data and the location of the crossings. They have first-hand knowledge of crossing changes and will be responsible for locating the crossings using one of DOT's standard LRMs.

The DOT can automate the process to populate the LRS tables from the crossing database. This would help minimize impacts to staff resources. The Office would need to acquire Oracle and may need to acquire GeoMedia Pro to enhance the maintenance process. Office staff would need intimate knowledge of datum and literal description theories and concepts.

The primary impacts to the Office would most likely be the need to meet LRS requirements for choosing points of reference and to meet LRS accuracy and temporal requirements. An example LRS point of reference for a crossing is the southern and then westernmost rail at a railway crossing. The point of reference distance from an anchor point must possess accuracy within 3-5 meters. The LRS requirements regarding timely updates suggest rail crossing database changes must be made as soon as possible. These changes are required prior to the crossing opening to traffic. If the accuracy and temporal requirements cannot be met initially, actions need to be taken to plan and implement them in the future.

6.8 District Offices

The District Offices would continue their role in placing both Reference Posts and Station Markers (posts or stamps) in the field for either construction or maintenance. They would also be assigned the new task of placing Anchor Point Monuments in the field for either construction or maintenance (the Office of Transportation Data would identify the need for a monument). The later phases of this project will provide some insight on the number of monuments to be placed.

The current procedures for field marker placement that the District Offices use would need to be modified to include processes for locating Reference Posts, Station Markers, and Anchor Point Monuments. This would most likely require the District Offices to invest in more efficient field data collection technology, such as GPS, to assist in locating these field markers. The field pilot phase of this project will influence this decision and the level of effort to meet these requirements. The District Offices would need training in the Coordinate Route LRM and new collection technologies.

The District Offices should also be responsible for entering Stationing LRM data from past improvement projects. They are the primary users of project stationing. This data entry task would involve extracting stationing information from the design or construction plans and referencing them to the linear datum. This process would replace the current workflows of attempting to manually integrate plan data from various sources. To perform this process, District Offices would need training in LRS, linear datums, the stationing LRM, and the Reference Post LRM. District Offices would also need access to and training in GeoMedia Pro and Oracle.

6.9 Statewide Operations Bureau, Office of Construction

The Office of Construction is responsible for supporting the District Office in the initial placement of station markers (posts or stamps) during an improvement project. The Office of Construction would have a new responsibility for supporting the District Office in the initial placement of anchor point monuments. The Office would need to modify procedures to include the process of positioning a station marker or anchor point monument, and recording their position relative to the LRS. The procedural changes will depend on results of the field pilot phase of this project and the most appropriate strategy for the District Offices.

The Office staff would need to obtain knowledge and skills on linear datum concepts and theories. The Office would not need to acquire any new technologies unless required to support the District Offices.

6.10 Engineering Bureau, Office of Traffic Engineering and Safety

The Office of Traffic Engineering plans the placement of reference posts. The Office should have a support role in how reference posts are designed and placed along the roadway. The Office would need to modify procedures to include how the position of a reference post is measured and recorded relative to the LRS. The procedural changes will depend on results of the field pilot phase of this project and the most appropriate strategy for the District Offices.

The Office staff would need to obtain knowledge and skills on linear datum concepts and theories. The Physical Design Team expects no new technologies are needed to perform this task.

6.11 Statewide Operations Bureau, Office of Maintenance

The Office of Maintenance currently has support roles in managing the Stationing and Reference Post Subsystems.

The Office of Maintenance would continue its support role in the field placement of station markers for the Stationing Subsystem. The Office would assume a new support role in the field placement of anchor point monuments. The Office would need to modify procedures to include how the position of a station marker and anchor point monument are measured and recorded relative to the LRS. The Office staff would need to obtain knowledge and skills on linear datum concepts and theories. The Office most likely would not need to acquire any new technologies unless required by changes to support the District Offices.

Regarding the Reference Post Subsystem, the Physical Design Team recommends the Office of Maintenance assume the overall Reference Post Subsystem management role. In the past, the Office implicitly performed such a role. The Office also should be considered for acquiring the Reference Post management tasks currently performed by the Information Technology Division, PMIS Support staff. PMIS Support currently identifies the need for replacement reference posts in the field (the District Office actually places the posts), and creates positions for the reference posts in the database. This reassignment is recommended for several reasons. First, the Office of Maintenance currently holds a coordination role for Reference Post management tasks. The reassignment would simplify managing the Reference Post Subsystem. Second, the Office establishes maintenance policies and procedures, which includes these tasks. Third, the Office has a direct relationship with the District Offices’ maintenance staff through formal and normal business operations.

In addition to these organizational changes, the Physical Design Team also recommends data, technology, and workflow improvements. These recommendations are primarily driven by the need to meet new LRS requirements and take advantage of more efficient technologies (GIS, relational databases, GPS, etc.).

The data improvement involves inventorying all reference posts. The LRS requires all reference posts be inventoried and their positions along the roadways be referenced to the datum based on LRS accuracy requirements (the distance along an anchor section must be accurate to within 3-5 meters). Currently, only the locations of North / East roadway posts are inventoried (South / West posts are assumed to be directly across from the North / East posts). In order to meet the LRS accuracy requirement, all posts will need to be inventoried.

The Physical Design Team also recommends that newer technologies be used to determine reference post LRS positions. The current technology for collecting post positions (videolog inventory) also does not meet the LRS accuracy requirement. The Iowa DOT will need to consider more accurate location referencing technologies. One solution is to provide District Offices with GPS equipment and have them collect the locations of new posts when they install them, using the Coordinate Route LRM. This solution would require the District Offices to enter the location data into the LRS or pass the data to the Office of Maintenance who would enter the data.

The Physical Design Team also recommends that a new reference post database be designed in the LRS environment to replace the existing mainframe database. The current mainframe reference post database does not support new LRS requirements. The LRS requires that Reference Posts be stored with Transport Link location formats. The current mainframe database does not support this format. The current format is GIMS PI milepoint. This LRM is not an appropriate location format for reference posts. It introduces location error into the reference post locations because it does not provide the same accumulative route locations as required for the reference post system. To make these changes to the existing mainframe system would not be as effective as redesigning the mainframe system in the new LRS environment.

Provided these improvements are implemented, the Office of Maintenance would require training on LRS, the linear datum, and reference post procedures. In addition, staffing required to perform these processes would be met by reallocating resources currently assigned to PMIS Support. Additional resources would be needed to meet the inventory requirements of South / West roadway posts and other additional LRS management responsibilities. The workflow and technology changes would require additional technology (Oracle and GeoMedia Pro) and additional training (any new LRMs that would be implemented to support Reference Post management).

6.12 Research Management Division

The Research Management Division should initially manage the Coordinate Route Subsystem. The Coordinate Route Subsystem has no data maintenance functions because it relies on the maintenance functions of the cartographic centerline layer and other LRS subsystems (Route and Datum). The Coordinate Route Subsystem relies on the newest technologies for field and office location data collection at Iowa DOT (GIS, GPS, cellular). The purpose of this management role is to assess, market, and help move newer technologies into standard practice in DOT offices that produce data. The current situation is in alignment with a Research Management objective to study and then integrate new technologies into the organization.

This assignment is contrary to the recommendations from the Logical Design Summary Document. It states that the Coordinate Route Subsystem be assigned to the same office responsible for the Route Subsystem, Datum Subsystem, or cartography centerline management. However, this proposed assignment is only until the use of these specific technologies stabilizes at Iowa DOT. Research Management must identify a permanent organizational unit once this happens.

6.13 Motor Vehicle Division, Office of Driver Services

The Motor Vehicles Division, Office of Driver Services, should assist in the maintenance of at least the Unofficial Route System in the LRS. Unofficial routes are routes familiar to local officials or the local public, but are not part of the formally posted route systems (e.g., 'Old Miller's Road', or the 'Southwest By-pass"). The staff in this Office is typically the first to encounter and require use of these route types. Office staff would need to be able to enter new route names in the Unofficial Assigned Routes table and enter route paths in the Route Link table. This would be a new process. The Office staff would need to become familiar with the route naming and route link mapping constructs of the LRS. The Office would need Oracle and GeoMedia Pro technology.

6.14 Information Technology Division, Oracle Database Administration

The DOT Oracle Database Administrator (DBA) should play a key support role to the GDW staging processes. Staging processes take raw operational data and format it for GDW purposes. The Oracle DBA should assist in data staging requirement analysis, table definitions, Oracle instance and schema management, security, etc. The Oracle DBA should play a very significant role in the prototype and initial implementation of the LRS. Assignments and direction given to the DOT Oracle Database Administrator will come from the Office assigned the LRS Manager and the GDW Manager roles (Information Technology Division, GIS Coordinator)

The Physical Design Team will work with the Information Technology Division to identify the staff responsible for maintaining code within the LRS business logic tier (e.g., the transformation algorithms). Regardless, this person(s) would receive assignments and direction from the Office assigned the LRS Manager and GDW Manager roles (Information Technology Division, GIS Coordinator).

The Oracle DBA would need intimate knowledge and skills with Oracle, Oracle Spatial, and Oracle Spatial LRSx constructs. The Oracle DBA also would need to expand skills and knowledge in data warehousing concepts and practices. The Oracle DBA would need Oracle and GeoMedia Pro technology.

6.15 Transport System – Multiple Participants

Transport Systems define the extent of the datum and the different ways in which the DOT will subdivide the transportation system to support DOT policy. Example Transport Systems include the Farm-to-Market Transport System and the City of Ames Roadway System

The primary decision-makers regarding new or changing DOT policy are the DOT Division Directors. However, the Division Managers would most likely define the actions to implement the policy. The LRS Board of Directors, with support from the LRS

Working Team, would need to support the Division Managers so these actions include changes to the LRS. The Office of Transportation Data, System Management has been targeted to perform data maintenance to Transport Systems.

7 APPENDIX A: TRANSFORMATION ALGORITHMS

This appendix contains a description of the algorithms necessary to perform transformations from the individual LRMs to the Datum and to provide the reverse transformations from the Datum to the LRMs.

Each algorithm describes the input and outputs that are necessary to initialize the transformation, and a list of tables and column names that are accessed during the transformation process.

7.1 Milepoint LRM To Anchor Section Offset

This algorithm is used to assign an anchor point-offset reference to a location reference described using a Milepoint LRM.

7.1.1 Point Events

7.1.1.1 Inputs

Parameter	**Type**	**Quantity Type**	**Description**
Transportation System Type	Text	N / A	Name of the transportation system that the route belongs to.
Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Route Name	Text	N / A	Name of the signed route.
Milepoint	Number	Miles	Arbitrary non-negative value for milepoint over the signed route.

7.1.1.2 Outputs

Parameter	**Type**	**Quantity Type**	**Description**
Anchor Section ID	Number	N / A	Unique ID of an anchor section in the datum.
Anchor Section Offset	Number	Meters	Offset in meters from the start of the anchor section.

7.1.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME*.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME.*
3. Select *TRANSPORT_SYSTEM_ID* from the **TRANSPORT_SYSTEM** table given *NAME* for the Transport System.
4. Convert offset for milepoint from miles to meters.
5. Select *ROUTE_LINK_ID*, *TRANSPORT_LINK_ID*, for all rows in **ROUTE_LINK** table given *ROUTE_ID* from 2. Order by *ORDINAL*.
6. For each row retuned from 5
    a. Get *ACCUMULATED_BEGIN_DISTANCE*, *ACCUMULATED_END_DISTANCE* from **TRANSPORT_SYSTEM_MILEPOINT** table given *ROUTE_LINK_ID from 5 and TRANSPORT_SYSTEM_ID from 3*.

    b. if offset lies in the range *ACCUMULATED_BEGIN_DISTANCE*.. *ACCUMULATED_END_DISTANCE*

        i. Store *TRANSPORT_LINK_ID*
        ii. Calculate and store absolute offset over row.
        iii. Goto 8.

    c. Iterate otherwise
7. Report milepoint not found.
8. Select *CONFLATION_ANCHOR_SECTION*, *DISTANCE* from **TRANSPORT_LINK** table given *TRANSPORT_LINK_IDs* from 6b.
9. Calculate relative offset over Transport Link from 8.
10. For each segment in *CONFLATION_ANCHOR_SECTION*
    a. Get *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID from segment.*

    b. Calculate length of segment, cumulative lengths of all segments processed and percentage of TransportLink covered by cumulative length.

    c. If relative offset lies on this segment

        i. Set *Anchor Section ID*
        ii. Calculate *Anchor Section Offset*
        iii. Goto 10

    d. Iterate Otherwise
11. Return *Anchor Section ID* and *Anchor Section Offset*

7.1.1.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_SYSTEM	NAME
	TRANSPORT_SYSTEM_ID
ROUTE_LINK	ROUTE_LINK_ID
	ROUTE_ID
	ORDINAL
	TRANSPORT_LINK_ID
TRANSPORT_SYSTEM_MILEPOINT	ROUTE_LINK_ID
	ACCUMULATED_BEGIN_DISTANCE
	ACCUMULATED_END_DISTANCE
	TRANSPORT_SYSTEM_ID
TRANSPORT_LINK	TRANSPORT_LINK_ID
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.1.2 Linear Events

7.1.2.1 Inputs

Parameter	Type	Quantity Type	Description
Transportation System Type	Text	N / A	Name of the transportation system that the route belongs to.
Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Route Name	Text	N / A	Name of the signed route.
Milepoint	Number	Miles	Arbitrary non-negative value for milepoint over the signed route.
To Milepoint	Number	Miles	Arbitrary non-negative value for milepoint over the signed route. Must be lower valued than Milepoint.

7.1.2.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID Conflation	LRSx Conflation Object	N / A	The conflation object contains a list of the anchor section IDs and part thereof that are covered by the linear event..

7.1.2.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given *the system NAME*.

2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID and FULL_NAME.*

3. Select *TRANSPORT_SYSTEM_ID* from the **TRANSPORT_SYSTEM** table given *NAME* for the Transport System.

4. Convert offsets for milepoint from miles to meters.

5. Select *ROUTE_LINK_ID*, *TRANSPORT_LINK_ID*, *ORDINAL*, for all rows in **ROUTE_LINK** table given *ROUTE_ID* from 2. Order by *ORDINAL*.

6. For each row retuned from 5

a. Get *ACCUMULATED_BEGIN_DISTANCE*, *ACCUMULATED_END_DISTANCE* from **TRANSPORT_SYSTEM_MILEPOINT** table given *ROUTE_LINK_ID from 5 and TRANSPORT_SYSTEM_ID from 3*.

b. if Start Offset or End Offset lies in the range *ACCUMULATED_BEGIN_DISTANCE*.. *ACCUMULATED_END_DISTANCE*

   i. Store *TRANSPORT_LINK_ID, ORDINAL* for offset
   ii. Calculate and store absolute offset over row.
   iii. If both start and end offsets have been found Goto 8.

c. Iterate otherwise

7. Report start or end mile point not found.

8. Select *CONFLATION_ANCHOR_SECTION*, *DISTANCE* from **TRANSPORT_LINK** table given *TRANSPORT_LINK_IDs* from 6b for the start location.

9. Calculate relative offset over Transport Link from 8.

10. Create a new empty conflation object and add to it the portion of the anchor section conflation that exists from the start offset over the transport link to the end of the transport link. If the end offset is on the same transport link determine the end of the anchor section from the end-offset, create the conflation object and go to 15.

11. For each route link in the range between the ordinal for the start point and the ordinal for the end point from the list created in 5, select *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_IDs*.

12. Append the CONFLATION_ANCHOR_SECTION to the new conflation object.

13. Perform Steps 8, 9 for the end offset.

14. Determine the portion of the anchor section conflation that exists from the start of the transport link to the end offset over the transport link. Append the potion to the new conflation object.

15. Return the conflation object.

7.1.2.4 Tables and Columns

Table	**Column Name**
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_SYSTEM	NAME
	TRANSPORT_SYSTEM_ID
ROUTE_LINK	ROUTE_LINK_ID
	ROUTE_ID
	ORDINAL
	TRANSPORT_LINK_ID
TRANSPORT_SYSTEM_MILEPOINT	ROUTE_LINK_ID
	ACCUMULATED_BEGIN_DISTANCE
	ACCUMULATED_END_DISTANCE
	TRANSPORT_SYSTEM_ID
TRANSPORT_LINK	TRANSPORT_LINK_ID
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.2 Anchor Section Offset to Milepoint LRM

This algorithm is used to assign a milepoint value to an anchor section-offset given a route name and a transport system to project the position onto.

7.2.1 Point Events

7.2.1.1 Inputs

Parameter	**Type**	**Quantity Type**	**Description**
Transport System Name	Text	N / A	Name of the transportation system that the route belongs to.
Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Route Name	Text	N / A	Name of the signed route.
Anchor Section ID	Number	N / A	Unique ID for an anchor section in the datum.
Offset	Number	Meters	Real valued non-negative offset over the anchor section.

7.2.1.2 Outputs

Parameter	**Type**	**Quantity Type**	**Description**
Milepoint	Number	Miles	Real valued, non-negative offset in miles from the start of the signed route.

7.2.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given *the system NAME*.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID and FULL_NAME.*
3. Select *TRANSPORT_SYSTEM_ID* from the **TRANSPORT_SYSTEM** table given *NAME* for the Transport System.
4. Select *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.* Calculate proportional offset over the anchor section given the meters offset from the start of the anchor section*.*
5. Select *TRANSPORT_LINK_ID* from **TRANSPORT_LINK_AS_XREF** given *ANCHOR_SECTION_ID* and offset calculated from 4 is in the range *FROM_POSITION*..*TO_POSITION.*
6. *Select DISTANCE* from **TRANSPORT_LINK** given *TRANSPORT_LINK_ID* from 5.
7. Calculate the relative offset over the transport link given the anchor section and offset.
8. Select *ROUTE_LINK_ID*, from **ROUTE_LINK** table given *ROUTE_ID* from 2 and *TRANSPORT_LINK_ID* from 5*.*
9. Select *ACCUMULATED_BEGIN_DISTANCE*, *ACCUMULATED_END_DISTANCE* from **TRANSPORT_SYSTEM_MILEPOINT** table given *ROUTE_LINK_ID* from 8 and TRANSPORT_SYSTEM_ID from 3. If no record exists return an error value stating that the anchor section offset value is not on the designated signed route.
10. Calculate the milepoint value by interpolating the milepoint from *ACCUMULATED_BEGIN_DISTANCE* and *ACCUMULATED_END_DISTANCE* values from 8 and the relative offset from 7.
11. Return milepoint.

7.2.1.4 Tables and Columns

Table	**Column Name**
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_SYSTEM	NAME
	TRANSPORT_SYSTEM_ID
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE
TRANSPORT_SYSTEM_AS_XREF	ANCHOR_SECTION_ID
	FROM_POSITION
	TO_POSITION
	TRANSPORT_LINK_ID
TRANSPORT_LINK	TRANSPORT_LINK_ID
	DISTANCE
ROUTE_LINK	ROUTE_LINK_ID
	TRANSPORT_LINK_ID
TRANSPORT_SYSTEM_MILEPOINT	ROUTE_LINK_ID
	TRANSPORT_SYSTEM_ID
	ACCUMULATED_BEGIN_DISTANCE
	ACCUMULATED_END_DISTANCE

7.2.2 Linear Events

7.2.2.1 Inputs

Parameter	**Type**	**Quantity Type**	**Description**
Transport System Name	Text	N / A	Name of the transportation system that the route belongs to.
Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Route Name	Text	N / A	Name of the signed route.
Anchor Section Conflation Object	LRSx Conflation Object	N / A	Conflation of the anchor sections.

7.2.2.2 Outputs

Parameter	**Type**	**Quantity Type**	**Description**
From Milepoint	Number	Miles	Real valued, non-negative offset in miles from the start of the signed route.
To Milepoint	Number	Miles	Real valued, non-negative offset in miles from the start of the signed route.

7.2.2.3 Algorithm

1. Select *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID* for the first segment in the conflation. Calculate proportional offset over the anchor section given the From Position of the segment.

2. Use the Point algorithm to determine the milepoint for the starting position.

3. Perform steps 1 and 2 to calculate the to milepoint using the last segment in the conflation object and the To position on the segment to determine the meters offset over the anchor section.

7.2.2.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_SYSTEM	NAME
	TRANSPORT_SYSTEM_ID
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE
TRANSPORT_SYSTEM_AS_XREF	ANCHOR_SECTION_ID
	FROM_POSITION
	TO_POSITION
	TRANSPORT_LINK_ID
TRANSPORT_LINK	TRANSPORT_LINK_ID
	DISTANCE
ROUTE_LINK	ROUTE_LINK_ID
	TRANSPORT_LINK_ID
TRANSPORT_SYSTEM_MILEPOINT	ROUTE_LINK_ID
	TRANSPORT_SYSTEM_ID
	ACCUMULATED_BEGIN_DISTANCE
	ACCUMULATED_END_DISTANCE

7.3 Reference Post LRM To Anchor Section Offset

This algorithm will convert a reference post location to an anchor section-offset reference.

7.3.1 Point Events

7.3.1.1 Inputs

Parameter	Type	Quantity Type	Description
Route Category	Text	N / A	Name of the category of signed route. “STATE”, “COUNTY”, “E911” etc.
Route Name	Text	N / A	Name of the signed route.
Reference Post Name	Text	N / A	Name of the reference post.
Post Offset	Number	Meters	Arbitrary real valued offset from the reference post.

7.3.1.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID	Number	N / A	Unique ID of an anchor section in the datum.
Offset	Number	Meters	Offset in meters from the start of the anchor section.

7.3.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME*.

2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME.*

3. Select *ROUTE_LINK_ID*, *TRANSPORT_LINK_ID*, *TRANSPORT_LINK_OFFSET* from the **REFERENCE_POST** table given *ROUTE_ID* and *REFERENCE_POST_NAME*.

4. Calculate position of the reference post plus offset over the route link using the *DISTANCE*. If the location is after the end of the route link select the next route link in the sequence specified by *ORDINAL*. If the location is before the start of the route link select the previous route link in the sequence specified by ordinal. Keep going until the correct route link is found.

5. Calculate the offset over the transport link on the route link found in 4 where the location falls.

6. Select *CONFLATION_ANCHOR_SECTION*, *DISTANCE* from **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID* found For each segment in *CONFLATION_ANCHOR_SECTION*

    a. Get *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID from segment.*

    b. Calculate length of segment, cumulative lengths of all segments processed and percentage of TransportLink covered by cumulative length.

    c. If relative offset lies on this segment

        i. Set *Anchor Section ID*
        ii. Calculate *Anchor Section Offset*
        iii. Goto 10

    d. Iterate Otherwise

7. Return *Anchor Section ID* and *Anchor Section Offset*

Note that is the post offset is zero then the algorithm simply returns the Anchor Section / Offset values for the reference post.

7.3.1.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
REFERENCE_POST	ROUTE_LINK_ID
	TRANSPORT_LINK_ID
	TRANSPORT_LINK_OFFSET
	ROUTE_ID
	REFERENCE_POST_NAME
ROUTE_LINK	ROUTE_LINK_ID
	ROUTE_ID
	TRANSPORT_LINK_ID
	DISTANCE
	ORDINAL
TRANSPORT_LINK	DISTANCE
	TRANSPORT_LINK_ID
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.3.2 Linear Events

7.3.2.1 Inputs

Parameter	Type	Quantity Type	Description
Route Category	Text	N / A	Name of the category of signed route. “STATE”, “COUNTY”, “E911” etc.
Route Name	Text	N / A	Name of the signed route.
From Reference Post Name	Text	N / A	Name of the reference post for the start of the event.
From Post Offset	Number	Meters	Arbitrary real valued offset from the start reference post.
To Reference Post Name	Text	N / A	Name of the reference post for the end of the event.
To Post Offset	Number	Meters	Arbitrary real valued offset from the end reference post.

7.3.2.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID Conflation	LRSx Conflation Object	N / A	The conflation object contains a list of the anchor section IDs and part thereof that are covered by the linear event..

7.3.2.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given *the system NAME*.

2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID and FULL_NAME.*

3. Select *ROUTE_LINK_ID*, *TRANSPORT_LINK_ID*, *TRANSPORT_LINK_OFFSET* from the **REFERENCE_POST** table given *ROUTE_ID* and *REFERENCE_POST_NAME* for the start reference post.

4. Calculate position of the reference post plus start offset over the route link using the *DISTANCE*. If the location is after the end of the route link select the next route link in the sequence specified by *ORDINAL*. If the location is before the start of the route link select the previous route link in the sequence specified by ordinal. Keep going until the correct route link is found. Store the *ORDINAL* and *TRANSPORT_LINK_ID* values for the selected route link.

5. Select *CONFLATION_ANCHOR_SECTION*, *DISTANCE* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID*. Calculate the offset over the transport link on the route link found in 4 where the location falls.

6. Create a new empty conflation object and add to it the portion of the anchor section conflation that exists from the start offset over the transport link to the end of the transport link. If the end offset is on the same transport link determine the end of the anchor section from the end-offset, create the conflation object and go to 11.

7. Select TRANSPORT_LINK_ID for each route link in the range between the ordinal for the start point and the ordinal for the end given *ROUTE_ID*, order by *ORDINAL*.

8. For each record returned from 7, select *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID*. Append the *CONFLATION_ANCHOR_SECTION* to the new conflation object.

9. Perform Steps 3, 4, and 5 for the end offset*.*

10. Determine the portion of the anchor section conflation that exists from the start of the transport link to the end offset over the transport link. Append the potion to the new conflation object.

11. Return the conflation object.

Note that is the post offset is zero then the algorithm simply returns the Anchor Section / Offset values for the reference post.

7.3.2.4 Tables and Columns

Table	**Column Name**
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
REFERENCE_POST	ROUTE_LINK_ID
	TRANSPORT_LINK_ID
	TRANSPORT_LINK_OFFSET
	ROUTE_ID
	REFERENCE_POST_NAME
ROUTE_LINK	ROUTE_LINK_ID
	ROUTE_ID
	TRANSPORT_LINK_ID
	DISTANCE
	ORDINAL
TRANSPORT_LINK	DISTANCE
	TRANSPORT_LINK_ID
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.4 Anchor Section Offset To Reference Post LRM

This algorithm will convert an anchor section offset tuple to a reference post and offset value. This algorithm will search for the closest reference post to the datum location. Alternatively, step 8 can be omitted and the algorithm will search for the closest reference post where the offset is positively valued.

7.4.1 Point Events

7.4.1.1 Inputs

Parameter	Type	Quantity Type	Description
Route Category	Text	N / A	Name of the category of signed route. “STATE”, “COUNTY”, “E911” etc.
Route Name	Text	N / A	Name of the signed route.
Anchor Section	Number	N / A	Unique ID of an anchor section..
Offset	Number	Meters	Real valued non-negative offset over the anchor section in the range 0..9656.064

7.4.1.2 Outputs

Parameter	Type	Quantity Type	Description
Reference Post Name	Text	N / A	Name of the reference post closest to the datum location.
Offset	Number	Meters	Offset in meters from the start of the reference post.

7.4.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME*.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME.*
3. Select *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.* Calculate proportional offset over the anchor section given the meters offset from the start of the anchor section*.*
4. Select *TRANSPORT_LINK_ID,* from **TRANSPORT_LINK_AS_XREF** given *ANCHOR_SECTION_ID* and offset calculated from 3 is in the range *FROM_POSITION*..*TO_POSITION.*
5. *Select DISTANCE* from **TRANSPORT_LINK** given *TRANSPORT_LINK_ID* from 4.
6. Calculate the relative offset over the transport link given the anchor section and offset.
7. Select all *ROUTE_LINK_ID, TRANSPORT_LINK_ID* from **ROUTE_LINK** table given *ROUTE_ID*. Order by ORDINAL.
8. For each row returned from 7 select *REFERENCE_POST_NAME, POST_VALUE* from **REFERENCE_POST** given *TRANSPORT_LINK_ID* and *ROUTE_ID*.
9. If there are any reference posts selected calculate the offset to each of the reference posts. Save the minimum valued reference post and offset.
10. Return the minimum valued reference post name and offset.

7.4.1.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
ANCHOR_SECTION	DATUM_DISTANCE
	ANCHOR_SECTION_ID
TRANSPORT_LINK_XREF	TRANSPORT_LINK_ID
	ANCHOR_SECTION_ID
	FROM_POSITION
	TO_POSITION
TRANSPORT_LINK	TRANSPORT_LINK_ID
	DISTANCE
ROUTE_LINK	ROUTE_LINK_ID
	TRANSPORT_LINK_ID
	ROUTE_ID
	ORDINAL
REFERENCE_POST	REFERENCE_POST_NAME
	POST_VALUE
	TRANSPORT_LINK_ID
	ROUTE_ID

7.4.2 Linear Events

7.4.2.1 Inputs

Parameter	Type	Quantity Type	Description
Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Route Name	Text	N / A	Name of the signed route.
Anchor Section Conflation Object	LRSx Conflation Object	N / A	Conflation of the anchor sections.

7.4.2.2 Outputs

Parameter	Type	Quantity Type	Description
From Reference Post Name	Text	N / A	Name of the reference post closest to the start datum location.
From Offset	Number	Meters	Offset in meters from the start reference post.
To Reference Post Name	Text	N / A	Name of the reference post closest to the end datum location.
End Offset	Number	Meters	Offset in meters from the end reference post.

7.4.2.3 Algorithm

1. Select *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID* for the first segment in the conflation. Calculate proportional offset over the anchor section given the From Position of the segment.

2. Use the Point algorithm to determine the reference post and offset for the starting position.

3. Perform steps 1 and 2 to calculate the to reference post and offset using the last segment in the conflation object and the To position on the segment to determine the meters offset over the anchor section.

7.4.2.4 Tables and Columns

Table	**Column Name**
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
ANCHOR_SECTION	DATUM_DISTANCE
	ANCHOR_SECTION_ID
TRANSPORT_LINK_XREF	TRANSPORT_LINK_ID
	ANCHOR_SECTION_ID
	FROM_POSITION
	TO_POSITION
TRANSPORT_LINK	TRANSPORT_LINK_ID
	DISTANCE
ROUTE_LINK	ROUTE_LINK_ID
	TRANSPORT_LINK_ID
	ROUTE_ID
	ORDINAL
REFERENCE_POST	REFERENCE_POST_NAME
	POST_VALUE
	TRANSPORT_LINK_ID
	ROUTE_ID

7.5 Segmental LRM To Anchor Section Offset

This algorithm will return a list of anchor sections and proportions of the anchor sections covered by a segmental record. Note that this algorithm is really a simple selection procedure but is described herein for completeness.

7.5.1 Inputs

Parameter	Type	Quantity Type	Description
CONTROL_SECTION_ID	Number	N / A	Unique identifier for the control section.
SECTION_TYPE	Text		Either “GIMS” or “HPMS”

7.5.2 Outputs

Parameter	Type	Quantity Type	Description
CONFLATION	Conflation Object	N / A	List of unique anchor section Ids and offsets for all datum elements covered by the control section.

7.5.3 Algorithm

1. Select *CONFLATION_ANCHOR_SECTION* from **HPMS_SECTION** or **GIMS_SECTION** table given *CONTROL_SECTION_ID.*

2. Return conflation object.

7.5.4 Tables and Columns

Table	Column Name
HPMS_CONTROL_SECTION GIMS_CONTROL_SECTION	CONTROL_SECTION_ID
	CONFLATION_ANCHOR_SECTION

7.6 Anchor Section Offset To Segmental LRM

This algorithm will return the control section ID of a segmental record.

7.6.1 Inputs

Parameter	**Type**	**Quantity Type**	**Description**
SECTION_TYPE	Text		Either “GIMS” or “HPMS”
Anchor Section	Number	N / A	Unique ID of an anchor section..
Offset	Number	Meters	Real valued non-negative offset over the anchor.

7.6.2 Outputs

Parameter	**Type**	**Quantity Type**	**Description**
CONTROL_SECTION_ID	Number	N / A	Unique ID of a control section.

7.6.3 Algorithm

1. Select *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.* Calculate proportional offset over the anchor section given the meters offset from the start of the anchor section*.*

2. Select *CONTROL_SECTION_ID,* from either **HPMS_CONTROL_SECTION** or **GIMS_CONTROL_SECTION** table where *CONFLATION_ANCHOR_SECTION* contains the *ANCHOR_SECTION_ID* and proportional offset calculated in 1.

3. Return *CONTROL_SECTION_ID.*

7.6.4 Tables and Columns

Table	**Column Name**
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE
HPMS_CONTROL_SECTION GIMS_CONTROL_SECTION	CONTROL_SECTION_ID
	CONFLATION_ANCHOR_SECTION

7.7 Coordinate Route To Anchor Section Offset

This algorithm will return a list of anchor sections and proportions of the anchor sections covered by a coordinate route.

7.7.1 Point Events

7.7.1.1 Inputs

Parameter	Type	Quantity Type	Description
Route Category	Text	N / A	Name of the category of signed route. “STATE”, “COUNTY”, “E911” etc.
Route Name	Text	N / A	Name of the signed route.
X	Number	Map Units	X ordinate of the location for the event on the route.
Y	Number	Map Units	Y ordinate of the location for the event on the route.

7.7.1.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID	Number	N / A	Unique ID of an anchor section in the datum.
Anchor Section Offset	Number	Meters	Offset in meters from the start of the anchor section.

7.7.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME*.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME.*
3. Project location defined by X and Y onto the **ANCHOR_SECTION** table given *GEOMETRY_GIMS_CARTO*. Calculate the *ANCHOR_SECTION_ID* and offset given *ANCHOR_SECTION_ID* and *DATUM_DISTANCE*.
4. Select *TRANSPORT_LINK_ID,* from **TRANSPORT_LINK_AS_XREF** given *ANCHOR_SECTION_ID* and offset calculated from 3 is in the range *FROM_POSITION*..*TO_POSITION.*
5. Verify that a row exists in the **ROUTE_LINK** table with the given *ROUTE_ID* and *TRANSPORT_LINK_ID*. If a row was not found in 5, return an error stating that the point event is not located on the particular route.
6. Otherwise, return the Anchor Section ID and offset.

7.7.1.4 Tables and Columns

Table	**Column Name**
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
ANCHOR_SECTION	GEOMETRY_GIMS
	ANCHOR_SECTION_ID
	DATUM_DISTANCE
TRANSPORT_LINK_AS_XREF	TRANSPORT_LINK_ID
	ANCHOR_SECTION_ID
	FROM_POSITION
	TO_POSITION
ROUTE_LINK	ROUTE_ID
	TRANSPORT_LINK_ID

7.7.2 Linear Events

7.7.2.1 Inputs

Parameter	Type	Quantity Type	Description
Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Route Name	Text	N / A	Name of the signed route.
Start X	Number	Map Units	X ordinate of the starting location for the route.
Start Y	Number	Map Units	Y ordinate of the starting location for the route.
End X	Number	Map Units	X ordinate for the ending location for the route.
End Y	Number	Map Units	Y ordinate for the ending location for the route.

7.7.2.2 Outputs

Parameter	Type	Quantity Type	Description
Conflation	Conflation Object	N / A	List of unique anchor section Ids and offsets for all datum elements covered by the control section.

7.7.2.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME*.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME.*
3. Project location defined by Start X and Start Y onto the **ANCHOR_SECTION** table given *GEOMETRY_GIMS_CARTO*. Calculate the *ANCHOR_SECTION_ID* and offset given *ANCHOR_SECTION_ID* and *DATUM_DISTANCE*.
4. Project location defined by End X and End Y onto the **ANCHOR_SECTION** table given *GEOMETRY_GIMS_CARTO*. Calculate the *ANCHOR_SECTION_ID* and offset given *ANCHOR_SECTION_ID* and *DATUM_DISTANCE*.
5. Select *TRANSPORT_LINK_ID,* from **TRANSPORT_LINK_AS_XREF** given *ANCHOR_SECTION_ID* and offset calculated from 3 is in the range *FROM_POSITION*..*TO_POSITION.*
6. Select *TRANSPORT_LINK_ID,* from **TRANSPORT_LINK_AS_XREF** given *ANCHOR_SECTION_ID* and offset calculated from 4 is in the range *FROM_POSITION*..*TO_POSITION.*
7. Select *ROUTE_LINK_ID*, *ORDINAL*, *DISTANCE* from the **ROUTE_LINK** table given *TRANSPORT_LINK_ID* from 5 and *ROUTE_ID*. This is the starting location for the coordinate route.
8. Select *ROUTE_LINK_ID*, *ORDINAL*, *DISTANCE* from the **ROUTE_LINK** table given *TRANSPORT_LINK_ID* from 6 and *ROUTE_ID*. This is the ending location for the coordinate route.
9. Select the list of *ROUTE_LINK_ID*s between the *ORDINAL* value determined in 7 and that determined in 8.
10. If the *ORDINAL* from 7 is less than the *ORDINAL* value in 8 the coordinate stream is forward with respect to the route. Otherwise the coordinate route is reverse with respect to the route.
11. Calculate the proportion of the route link covered from the start point to the correct end of the first route link using the *DISTANCE* value from the route link. The correct end is determined by the direction determined from 10.
12. Calculate the proportion of the transport link covered by the start route link.
13. Calculate the portion of the *CONFLATION_ANCHOR_SECTION* object for the start transport link that is covered by the portion of the transport link determined in 12.
14. For each whole route link selected in 9 select transport link and append the anchor section conflation object to the list of anchor section conflations being created.
15. Calculate the proportion of the transport link covered by the end route link.
16. Calculate the portion of the *CONFLATION_ANCHOR_SECTION* object for the end transport link that is covered by the portion of the transport link determined in 15. Append the conflation object to the list of conflation objects.
17. Return the conflation object.

7.7.2.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
ANCHOR_SECTION	GEOMETRY_GIMS
	ANCHOR_SECTION_ID
	DATUM_DISTANCE
TRANSPORT_LINK_AS_XREF	TRANSPORT_LINK_ID
	ANCHOR_SECTION_ID
	FROM_POSITION
	TO_POSITION
TRANSPORT_LINK	TRANSPORT_LINK_ID
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ROUTE_LINK	ROUTE_LINK_ID
	ORDINAL
	ROUTE_ID
	TRANSPORT_LINK_ID
	DISTANCE

7.8 Anchor Section Offset To Coordinate Route

This is an invalid LRM transformation.

7.9 Stationing LRM To Anchor Section Offset

This algorithm is used to obtain an anchor point-offset reference to a position described using a stationing input. The stationing input consists of a station id and an offset from the station.

7.9.1 Point Events

7.9.1.1 Inputs

Parameter	Type	Quantity Type	Description
Improvement Project ID	Number	N / A	Unique ID for an improvement project.
Station	Number	N / A	Unique ID for a station post.
Station Post Offset	Number	Meters	Arbitrary value for offset over the project.

7.9.1.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID	Number	N / A	Unique ID of an anchor section in the datum.
Offset	Number	Meters	Offset in meters from the start of the anchor section.

7.9.1.3 Algorithm

```
1. Calculate the overall measurement offset over the project from the Station
   and Station Offset values.

2. Select PROJECT_STATION_FROM, PROJECT_STATION_TO, and
   CONFLATION_ANCHOR_SECTION from the PROJECT_SECTION table given
   IMPROVEMENT_PROJECT_ID.  Order by PROJECT_STATION_FROM.

3. For each row returned from 2

    a. If measurement offset is located on this project section

    b. Calculate the offset of the station and station offset over this
       section id

    c. For each segment in CONFLATION_ANCHOR_SECTION

         i. Get DATUM_DISTANCE from ANCHOR_SECTION table given
            ANCHOR_SECTION_ID

        ii. Calculate length of segment, cumulative lengths of all
            segments processed and percentage of Project Section
            covered by cumulative length.

       iii. If relative offset lies on this segment

             1. Set Anchor Section ID
             2. Calculate Anchor Section Offset
             3. Goto 5

4. Report anchor section not found.

5. Return Anchor Section ID and Anchor Section Offset
```

7.9.1.4 Tables and Columns

Table	Column Name
PROJECT_SECTION	PROJECT_STATION_FROM
	PROJECT_STATION_TO
	DISTANCE
	IMPROVEMENT_PROJECT_ID
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.9.2 Linear Events

7.9.2.1 Inputs

Parameter	Type	Quantity Type	Description
Improvement Project ID	Number	N / A	Unique ID for an improvement project.
From Station	Number	N / A	Unique ID for a station post.
From Station Post Offset	Number	Meters	Arbitrary value for offset over the project.
From Station	Number	N / A	Unique ID for a station post.
From Station Post Offset	Number	Meters	Arbitrary value for offset over the project.

7.9.2.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID Conflation	LRSx Conflation Object	N / A	The conflation object contains a list of the anchor section IDs and part thereof that are covered by the linear event..

7.9.2.3 Algorithm

1. Calculate the overall measurement offset over the project from the from Station and from Station Offset values.
2. Select *PROJECT_STATION_FROM, PROJECT_STATION_TO, and CONFLATION_ANCHOR_SECTION* from the **PROJECT_SECTION** table given *IMPROVEMENT_PROJECT_ID*. Order by *PROJECT_STATION_FROM*.
3. For each row returned from 2, if measurement offset is located on this project section calculate the offset of the station and station offset over this section.
4. Create a new empty conflation object and add to it the portion of the anchor section conflation that exists from the start offset over the project section to the end of the section.
5. Perform steps 1, 2 and 3 for the end offset. If the start and end offsets are on the same section calculate the conflation object between the two offsets and go to 9.
6. Select *CONFLATION_ANCHOR_SECTION* from the **PROJECT_SECTION** table given *IMPROVEMENT_PROJECT_ID* and where the FROM values of each project section are greater than or equal to the end of the starting project section, and the TO values of each project section are less than or equal to the start of the end project section.
7. For each record returned from 6, append the *CONFLATION_ANCHOR_SECTION* to the new conflation object.
8. Determine the portion of the anchor section conflation that exists from the start of the section from 3 for the end location. The portion of the conflation is from the start of the section to the position of the offset. Append the portion to the new conflation object.
9. Return the conflation object.

7.9.2.4 Tables and Columns

Table	**Column Name**
PROJECT_SECTION	PROJECT_STATION_FROM
	PROJECT_STATION_TO
	DISTANCE
	IMPROVEMENT_PROJECT_ID
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.10 Anchor Section Offset to Stationing LRM

This algorithm is used to assign a Station and Offset value to an anchor section-offset given an Improvement Project ID.

7.10.1 Point Events

7.10.1.1 Inputs

Parameter	**Type**	**Quantity Type**	**Description**
Improvement Project ID	Number	N / A	Unique ID for an improvement project.
Anchor Section ID	Number	N / A	Unique ID for an anchor section in the datum.
Offset	Number	Meters	Real valued non-negative offset over the anchor section.

7.10.1.2 Outputs

Parameter	**Type**	**Quantity Type**	**Description**
Station	Number	Meters	Real valued, non-negative offset in meters from the start of the project stationing.
Offset	Number	Meters	Real valued, non-negative offset in meters from the station post.

7.10.1.3 Algorithm

1. Select *DATUM_DISTANCE* from the **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.* Calculate proportional offset over the anchor section given the meters offset from the start of the anchor section*.*

2. Select *PROJECT_STATION_FROM, PROJECT_STATION_TO* from the **PROJECT_SECTION** table given *IMPROVEMENT_PROJECT_ID* where *CONFLATION_ANCHOR_SECTION* contains the *ANCHOR_SECTION_ID* and proportional offset calculated in 1.

3. Calculate the relative offset over the project section given the anchor section and offset.

4. Interpolate the position over the project section given *PROJECT_STATION_FROM* and *PROJECT_STATION_TO values* from 2.

5. Calculate and return the integer portion of the position and the decimal portion of the position.

7.10.1.4 Tables and Columns

Table	Column Name
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE
PROJECT_SECTION	PROJECT_STATION_FROM
	PROJECT_STATION_TO
	IMPROVEMENT_PROJECT_ID

7.10.2 Linear Events

7.10.2.1 Inputs

Parameter	Type	Quantity Type	Description
Improvement Project ID	Number	N / A	Unique ID for an improvement project.
Anchor Section Conflation Object	LRSx Conflation Object	N / A	Conflation of the anchor sections.

7.10.2.2 Outputs

Parameter	Type	Quantity Type	Description
From Station	Number	Meters	Real valued, non-negative offset in meters from the start of the project stationing.
From Offset	Number	Meters	Real valued, non-negative offset in meters from the station post.
To Station	Number	Meters	Real valued, non-negative offset in meters from the start of the project stationing.
To Offset	Number	Meters	Real valued, non-negative offset in meters from the station post.

7.10.2.3 Algorithm

1. Select *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID* for the first segment in the conflation. Calculate proportional offset over the anchor section given the From Position of the segment.

2. Use the Point algorithm to determine the station post and offset for the starting position.

3. Perform steps 1 and 2 to calculate the to station post and offset using the last segment in the conflation object and the To position on the segment to determine the meters offset over the anchor section.

7.10.2.4 Tables and Columns

Table	**Column Name**
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE
PROJECT_SECTION	PROJECT_STATION_FROM
	PROJECT_STATION_TO
	IMPROVEMENT_PROJECT_ID

7.11 Literal Description LRM To Anchor Section Offset

This algorithm is used to assign an anchor point-offset to a position described using a literal description input. The literal description can either reference an intersection or a bridge or railroad crossing feature.

There are four permutations of this algorithm each of which have different input requirements. The four permutations are differentiated by the type of reference features:

Algorithm	**On Feature Type**	**Towards Feature Type**
1	Intersection	Intersection
2	Intersection	Reference Feature
3	Reference Feature	Intersection
4	Reference Feature	Reference Feature

Each of these algorithms is detailed separately.

7.11.1 Algorithm 1: Intersection-Intersection

This algorithm processes a literal description defined by two intersections.

7.11.1.1 Point Events

7.11.1.1.1 Inputs

Parameter	Type	Quantity Type	Description
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
At Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
At Route Name	String	Route Name	Name of the cross street for the intersection.
Towards Route Category	String	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Towards Route Name	String	Route Name	Name of the cross street for the intersection.
Offset	Number	Meters	Measured length from the reference feature to the location.

7.11.1.1.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID	Number	N / A	Unique ID of an anchor section in the datum.
Offset	Number	Meters	Offset in meters from the start of the anchor section.

7.11.1.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the On Route.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the on route.
3. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the At Route.
4. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the At route.
5. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the Towards Route.
6. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the Towards route.
7. Select TRANSPORT_NODE_ID from the TRANSPORT_NODE_ROUTE_XREF table given ON_ROUTE_ID and AT_ROUTE_ID for the On Route and At Route values.
8. Select TRANSPORT_NODE_ID from the TRANSPORT_NODE_ROUTE_XREF table given ON_ROUTE_ID and AT_ROUTE_ID for the On Route and Towards Route values.
9. Select TRANSPORT_LINK_ID, DISTANCE, CONFLATION_ANCHOR_SECTION from the TRANSPORT_LINK table given TRANSPORT_NODE_ID_FROM and TRANSPORT_NODE_ID_TO values from 7 and 8.
10. Calculate offset over the transport link for the event.
11. For each segment in CONFLATION_ANCHOR_SECTION
    - a. Get *DATUM_DISTANCE* from the **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.*
    - b. Calculate length of segment, cumulative lengths of all segments processed and percentage of transport link covered by cumulative length.
    - c. If relative offset lies on this segment
        - i. Set *Anchor Section ID*
        - ii. Calculate *Anchor Section Offset*
        - iii. Goto 13
12. Report anchor section not found.
13. Return *Anchor Section ID* and *Anchor Section Offset*

7.11.1.1.4 Tables and Columns

Table	**Column Name**
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ON_ROUTE_ID
	AT_ROUTE_ID
TRANSPORT_LINK	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.11.1.2 Linear Events

7.11.1.2.1 Inputs

Parameter	Type	Quantity Type	Description
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
At Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
At Route Name	String	Route Name	Name of the cross street for the intersection.
Towards Route Category	String	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Towards Route Name	String	Route Name	Name of the cross street for the intersection.
Offset	Number	Meters	Measured length from the reference feature to the location.
Distance	Number	Meters	Measured length from the start of the event to its end.

7.11.1.2.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID Conflation	LRSx Conflation Object	N / A	The conflation object contains a list of the anchor section IDs and part thereof that are covered by the linear event..

7.11.1.2.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the On Route.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the On route.
3. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the At Route.
4. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the At route.
5. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the Towards Route.
6. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the Towards route.
7. Select TRANSPORT_NODE_ID from the TRANSPORT_NODE_ROUTE_XREF table given ON_ROUTE_ID and AT_ROUTE_ID for the On Route and At Route values.
8. Select TRANSPORT_NODE_ID from the TRANSPORT_NODE_ROUTE_XREF table given ON_ROUTE_ID and AT_ROUTE_ID for the On Route and Towards Route values.
9. Select TRANSPORT_LINK_ID, DISTANCE, CONFLATION_ANCHOR_SECTION from the TRANSPORT_LINK table given TRANSPORT_NODE_ID_FROM and TRANSPORT_NODE_ID_TO values from 7 and 8.
10. Calculate the starting offset for the event over the transport link.
11. Calculate the ending offset over the transport link given the starting offset and distance. If the end offset is on the same transport link calculate the portion of the CONFLATION_ANCHOR_SECTION that is covered by the extent of the event. Goto 16.
12. Create a new empty conflation object and add to it the portion of the anchor section conflation that exists from the start offset over the transport link to the end of the transport link determined by the node of the Toward route.
13. Use the Route Link table to move along the selected route. If the TRANSPORT_NODE_ID determined from 8 is the from-node from the link select the ORDINAL from the ROUTE_LINK table that is before the one for the TRANSPORT_LINK determined in 9. If the node is the to-node select the next ORDINAL. Walk forward or backwards in the route until the end of the event is located.
14. For each record returned from 13, select *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID*. Append the *CONFLATION_ANCHOR_SECTION* to the new conflation object. If the route is being traversed in reverse order make sure the reverse the conflation before appending it.
15. Determine the portion of the anchor section conflation that exists from the start of the transport link to the end offset over the transport link. Append the potion to the new conflation object.
16. Return the conflation object.

7.11.1.2.4 Tables and Columns

Table	**Column Name**
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ON_ROUTE_ID
	AT_ROUTE_ID
TRANSPORT_LINK	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ROUTE_LINK	ROUTE__ID
	TRANSPORT_LINK_ID
	ORDINAL
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.11.2 Algorithm 2: Intersection-Reference Feature

This algorithm processes a literal description defined by an intersection and reference feature.

7.11.2.1 Point Events

7.11.2.1.1 Inputs

Parameter	Type	Quantity Type	Description
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
At Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
At Route Name	String	Route Name	Name of the cross street for the intersection.
Towards Feature	String	N / A	Name of a reference feature.
Offset	Number	Meters	Measured length from the reference feature to the location.

7.11.2.1.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID	Number	N / A	Unique ID of an anchor section in the datum.
Offset	Number	Meters	Offset in meters from the start of the anchor section.

7.11.2.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given *the system NAME* for the On Route.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the on route.
3. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the At Route.
4. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the At route.
5. Select TRANSPORT_NODE_ID from the TRANSPORT_NODE_ROUTE_XREF table given ON_ROUTE_ID and AT_ROUTE_ID for the On Route and At Route values.
6. Select *REFERENCE_FEATURE_ID, TRANSPORT_LINK_ID* from the **REFERENCE_FEATURE** table joined with the **REF_FEATURE_ROUTE_XREF** table given *NAME* for the reference feature, *REFERENCE_FEATURE_CATEGORY_ID* and *ROUTE_ID* from 2.
7. Select *TRANSPORT_NODE_ID_FROM*, *TRANSPORT_NODE_ID_TO*, *DISTANCE*, and *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID.*
8. If the TRANSPORT_NODE_ID from 5 is the to-node for the transport link calculate the offset from the from-node.
9. For each segment in CONFLATION_ANCHOR_SECTION
    a. Get *DATUM_DISTANCE* from the **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.*

    b. Calculate length of segment, cumulative lengths of all segments processed and percentage of transport link covered by cumulative length.

    c. If relative offset lies on this segment

        i. Set *Anchor Section ID*
        ii. Calculate *Anchor Section Offset*
        iii. Goto 11
10. Report anchor section not found.
11. Return *Anchor Section ID* and *Anchor Section Offset*

7.11.2.1.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ON_ROUTE_ID
	AT_ROUTE_ID
REFERENCE_FEATURE	REFERENCE_FEATURE_ID
	TRANSPORT_LINK_ID
	NAME
REF_FEATURE_ROUTE_XREF	REFERENCE_FEATURE_ID
	ROUTE_ID
TRANSPORT_LINK	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.11.2.2 Linear Events

7.11.2.2.1 Inputs

Parameter	Type	Quantity Type	Description
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
At Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
At Route Name	String	Route Name	Name of the cross street for the intersection.
Towards Feature	String	N / A	Name of a reference feature.
Offset	Number	Meters	Measured length from the reference feature to the location.
Distance	Number	Meters	Measured length from the start of the event to its end.

7.11.2.2.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID Conflation	LRSx Conflation Object	N / A	The conflation object contains a list of the anchor section IDs and part thereof that are covered by the linear event..

7.11.2.2.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the On Route.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the on route.
3. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the At Route.
4. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the At route.
5. Select TRANSPORT_NODE_ID from the TRANSPORT_NODE_ROUTE_XREF table given ON_ROUTE_ID and AT_ROUTE_ID for the On Route and At Route values.
6. Select *REFERENCE_FEATURE_ID, TRANSPORT_LINK_ID* from the **REFERENCE_FEATURE** table joined with the **REF_FEATURE_ROUTE_XREF** table given *NAME* for the reference feature, *REFERENCE_FEATURE_CATEGORY_ID* and *ROUTE_ID* from 2.
7. Select *TRANSPORT_NODE_ID_FROM*, *TRANSPORT_NODE_ID_TO*, *DISTANCE*, and *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID* from 6.
8. Calculate the starting offset for the event over the transport link.
9. Calculate the ending offset over the transport link given the starting offset and distance. If the end offset is on the same transport link calculate the portion of the CONFLATION_ANCHOR_SECTION that is covered by the extent of the event. Goto 14.
10. Create a new empty conflation object and add to it the portion of the anchor section conflation that exists from the start offset over the transport link to the end of the transport link determined by the node of the Toward route.
11. Use the Route Link table to move along the selected route. If the TRANSPORT_NODE_ID determined from 5 is the from-node from the link select the ORDINAL from the ROUTE_LINK table that is after the one for the TRANSPORT_LINK determined in 9. If the node is the to-node select the previous ORDINAL. Walk forward or backwards in the route until the end of the event is located.
12. For each record returned from 11, select *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID*. Append the *CONFLATION_ANCHOR_SECTION* to the new conflation object. If the route is being traversed in reverse order make sure the reverse the conflation before appending it.
13. Determine the portion of the anchor section conflation that exists from the start of the transport link to the end offset over the transport link. Append the potion to the new conflation object.
14. Return the conflation object

7.11.2.2.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ON_ROUTE_ID
	AT_ROUTE_ID
REFERENCE_FEATURE	REFERENCE_FEATURE_ID
	TRANSPORT_LINK_ID
	NAME
REF_FEATURE_ROUTE_XREF	REFERENCE_FEATURE_ID
	ROUTE_ID
TRANSPORT_LINK	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ROUTE_LINK	ROUTE__ID
	TRANSPORT_LINK_ID
	ORDINAL
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.11.3 Algorithm 3: Reference Feature - Intersection

This algorithm processes a literal description defined by a reference feature and an intersection.

7.11.3.1 Point Events

7.11.3.1.1 Inputs

Parameter	Type	Quantity Type	Description
Reference Feature	String	N / A	Name of a reference feature.
On Route Category	Text	N / A	Name of the category of signed route. “STATE”, “COUNTY”, “E911” etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
Towards Route Category	Text	N / A	Name of the category of signed route. “STATE”, “COUNTY”, “E911” etc.
Towards Route Name	String	Route Name	Name of the cross street for the intersection.
Offset	Number	Meters	Measured length from the reference feature to the location.

7.11.3.1.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID	Number	N / A	Unique ID of an anchor section in the datum.
Offset	Number	Meters	Offset in meters from the start of the anchor section.

7.11.3.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the On Route.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the on route.
3. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the Towards Route.
4. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the Towards route.
5. Select *REFERENCE_FEATURE_ID, TRANSPORT_LINK_ID, TRANSPORT_LINK_OFFSET* from the **REFERENCE_FEATURE** table joined with the **REF_FEATURE_ROUTE_XREF** table given *NAME* for the reference feature, *REFERENCE_FEATURE_CATEGORY_ID* and *ROUTE_ID* from 2.
6. Select TRANSPORT_NODE_ID from the TRANSPORT_NODE_ROUTE_XREF table given ON_ROUTE_ID and AT_ROUTE_ID for the On Route and Towards Route values.
7. Select *TRANSPORT_NODE_ID_FROM*, *TRANSPORT_NODE_ID_TO*, *DISTANCE*, and *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID*.
8. Use the distance of the reference feature from the event to calculate the distance of the event to the end of the transport link at the node specified in 6. Use this distance to calculate the offset over the transport link for the event.
9. For each segment in CONFLATION_ANCHOR_SECTION

   a. Get *DATUM_DISTANCE* from the **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.*

   b. Calculate length of segment, cumulative lengths of all segments processed and percentage of transport link covered by cumulative length.

   c. If relative offset lies on this segment

      iv. Set *Anchor Section ID*
      v. Calculate *Anchor Section Offset*
      vi. Goto 11

10. Report anchor section not found.
11. Return *Anchor Section ID* and *Anchor Section Offset*

7.11.3.1.4 Tables and Columns

Table	**Column Name**
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ON_ROUTE_ID
	AT_ROUTE_ID
REFERENCE_FEATURE	REFERENCE_FEATURE_ID
	TRANSPORT_LINK_ID
	TRANSPORT_LINK_OFFSET
	NAME
REF_FEATURE_ROUTE_XREF	REFERENCE_FEATURE_ID
	ROUTE_ID
TRANSPORT_LINK	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.11.3.2 Linear Events

7.11.3.2.1 Inputs

Parameter	Type	Quantity Type	Description
Reference Feature	String	N / A	Name of a reference feature.
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
Towards Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Towards Route Name	String	Route Name	Name of the cross street for the intersection.
Offset	Number	Meters	Measured length from the reference feature to the location.
Distance	Number	Meters	Measured length from the start of the event to its end.

7.11.3.2.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID Conflation	LRSx Conflation Object	N / A	The conflation object contains a list of the anchor section IDs and part thereof that are covered by the linear event..

7.11.3.2.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the On Route.

2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the on route.

3. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the Towards Route.

4. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the Towards route.

5. Select *REFERENCE_FEATURE_ID, TRANSPORT_LINK_ID, TRANSPORT_LINK_OFFSET* from the **REFERENCE_FEATURE** table joined with the **REF_FEATURE_ROUTE_XREF** table given *NAME* for the reference feature, *REFERENCE_FEATURE_CATEGORY_ID* and *ROUTE_ID* from 2.

6. Select TRANSPORT_NODE_ID from the TRANSPORT_NODE_ROUTE_XREF table given ON_ROUTE_ID and AT_ROUTE_ID for the On Route and Towards Route values.

7. Select *TRANSPORT_NODE_ID_FROM*, *TRANSPORT_NODE_ID_TO*, *DISTANCE*, and *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID*.

8. Use the distance of the reference feature from the event to calculate the distance of the event to the end of the transport link at the node specified in 6. Use this distance to calculate the offset over the transport link for the event.

9. Calculate the ending offset over the transport link given the starting offset and distance. If the end offset is on the same transport link calculate the portion of the CONFLATION_ANCHOR_SECTION that is covered by the extent of the event. Goto 14.

10. Create a new empty conflation object and add to it the portion of the anchor section conflation that exists from the start offset over the transport link to the end of the transport link determined by the node of the Toward route.

11. Use the Route Link table to move along the selected route. If the TRANSPORT_NODE_ID determined from 5 is the from-node from the link select the ORDINAL from the ROUTE_LINK table that is after the one for the TRANSPORT_LINK determined in 9. If the node is the to-node select the previous ORDINAL. Walk forward or backwards in the route until the end of the event is located.

12. For each record returned from 11, select *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID*. Append the *CONFLATION_ANCHOR_SECTION* to the new conflation object. If the route is being traversed in reverse order make sure the reverse the conflation before appending it.

13. Determine the portion of the anchor section conflation that exists from the start of the transport link to the end offset over the transport link. Append the potion to the new conflation object.

14. Return the conflation object

7.11.3.2.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ON_ROUTE_ID
	AT_ROUTE_ID
REFERENCE_FEATURE	REFERENCE_FEATURE_ID
	TRANSPORT_LINK_ID
	TRANSPORT_LINK_OFFSET
	NAME
REF_FEATURE_ROUTE_XREF	REFERENCE_FEATURE_ID
	ROUTE_ID
TRANSPORT_LINK	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ROUTE_LINK	ROUTE__ID
	TRANSPORT_LINK_ID
	ORDINAL
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.11.4 Algorithm 4: Reference Feature – Reference Feature

This algorithm processes a literal description defined by two reference features.

7.11.4.1 Point Events

7.11.4.1.1 Inputs

Parameter	Type	Quantity Type	Description
Reference Feature	String	N / A	Name of a reference feature.
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
Towards Reference Feature	String	N / A	Name of a reference feature.
Offset	Number	Meters	Measured length from the reference feature to the location.

7.11.4.1.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID	Number	N / A	Unique ID of an anchor section in the datum.
Offset	Number	Meters	Offset in meters from the start of the anchor section.

7.11.4.1.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the On Route.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the on route.
3. Select *REFERENCE_FEATURE_ID, TRANSPORT_LINK_ID, TRANSPORT_LINK_OFFSET* from the **REFERENCE_FEATURE** table joined with the **REF_FEATURE_ROUTE_XREF** table given *NAME* for the reference feature, *REFERENCE_FEATURE_CATEGORY_ID* and *ROUTE_ID*.
4. Select *REFERENCE_FEATURE_ID, TRANSPORT_LINK_ID, TRANSPORT_LINK_OFFSET* from the **REFERENCE_FEATURE** table joined with the **REF_FEATURE_ROUTE_XREF** table given *NAME* for the towards reference feature, *REFERENCE_FEATURE_CATEGORY_ID* and *ROUTE_ID*.
5. Select *ROUTE_LINK_ID*, *ORDINAL*, *DISTANCE* from **ROUTE_LINK** table given *TRANSPORT_LINK_ID* From 3 and *ROUTE_ID*. This is the location for the reference feature.
6. Select *ROUTE_LINK_ID*, *ORDINAL*, *DISTANCE* from **ROUTE_LINK** table given *TRANSPORT_LINK_ID* From 4 and *ROUTE_ID*. This is the location for the towards feature.
7. If the *ORDINAL* from 5 is less than the *ORDINAL* value in 6 the direction is forward with respect to the route. Otherwise the direction is reverse with respect to the route.
8. Calculate the offset from the reference feature over the route link given the direction relative to the route direction discovered in 7. If the offset occurs on the previous route link select the route link with the previous ordinal value. If the offset occurs on the next route link select the route link with the next ordinal value. Keep going until the correct route link and offset are found.
9. Select DISTANCE, CONFLATION_ANCHOR_SECTION for the route link found from 8.
10. Calculate the proportion of the transport link covered by the route link at the offset.
11. For each segment in *CONFLATION_ANCHOR_SECTION*
    a. Get *DATUM_DISTANCE* from **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID*
    b. Calculate length of segment, cumulative lengths of all segments processed and percentage of Project Section covered by cumulative length.
    c. If relative offset lies on this segment
        i. Set *Anchor Section ID*
        ii. Calculate *Anchor Section Offset*
        iii. Goto 13
12. Report anchor section not found.
13. Return *Anchor Section ID* and *Anchor Section Offset*

7.11.4.1.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
REFERENCE_FEATURE	REFERENCE_FEATURE_ID
	TRANSPORT_LINK_ID
	TRANSPORT_LINK_OFFSET
	NAME
REF_FEATURE_ROUTE_XREF	REFERENCE_FEATURE_ID
	ROUTE_ID
ROUTE_LINK	ROUTE_ID
	ORDINAL
	DISTANCE
	TRANSPORT_LINK_ID
TRANSPORT_LINK	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.11.4.2 Linear Events

7.11.4.2.1 Inputs

Parameter	Type	Quantity Type	Description
Reference Feature	String	N / A	Name of a reference feature.
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
Towards Reference Feature	String	N / A	Name of a reference feature.
Offset	Number	Meters	Measured length from the reference feature to the location.
Distance	Number	Meters	Measured length from the start of the event to its end.

7.11.4.2.2 Outputs

Parameter	Type	Quantity Type	Description
Anchor Section ID Conflation	LRSx Conflation Object	N / A	The conflation object contains a list of the anchor section IDs and part thereof that are covered by the linear event..

7.11.4.2.3 Algorithm

1. Select *ROUTE_SYSTEM_ID* from **ROUTE_SYSTEM** given the system *NAME* for the On Route.
2. Select *ROUTE_ID* from table **ROUTE** given *ROUTE_SYSTEM_ID* and *FULL_NAME*. This is the route id for the on route.
3. Select *REFERENCE_FEATURE_ID, TRANSPORT_LINK_ID, TRANSPORT_LINK_OFFSET* from the **REFERENCE_FEATURE** table joined with the **REF_FEATURE_ROUTE_XREF** table given *NAME* for the reference feature, *REFERENCE_FEATURE_CATEGORY_ID* and *ROUTE_ID*.
4. Select *REFERENCE_FEATURE_ID, TRANSPORT_LINK_ID, TRANSPORT_LINK_OFFSET* from the **REFERENCE_FEATURE** table joined with the **REF_FEATURE_ROUTE_XREF** table given *NAME* for the towards reference feature, *REFERENCE_FEATURE_CATEGORY_ID* and *ROUTE_ID*.
5. Select *ROUTE_LINK_ID*, *ORDINAL*, *DISTANCE* from **ROUTE_LINK** table given *TRANSPORT_LINK_ID* From 3 and *ROUTE_ID*. This is the location for the reference feature.
6. Select *ROUTE_LINK_ID*, *ORDINAL*, *DISTANCE* from **ROUTE_LINK** table given *TRANSPORT_LINK_ID* From 4 and *ROUTE_ID*. This is the location for the towards feature.
7. If the *ORDINAL* from 5 is less than the *ORDINAL* value in 6 the direction is forward with respect to the route. Otherwise the direction is reverse with respect to the route.
8. Calculate the start offset from the reference feature over the route link given the direction relative to the route direction discovered in 7. If the offset occurs on the previous route link select the route link with the previous ordinal value. If the offset occurs on the next route link select the route link with the next ordinal value. Keep going until the correct route link and offset are found.
9. Calculate the ending offset over the transport link given the starting offset and distance. If the end offset is on the same transport link calculate the portion of the CONFLATION_ANCHOR_SECTION that is covered by the extent of the event. Goto 14.
10. Create a new empty conflation object and add to it the portion of the anchor section conflation that exists from the start offset over the transport link to the end of the transport link.
11. Use the Route Link table to move along the selected route in the direction specified by 7. Walk forward or backwards in the route until the end of the event is located.
12. For each record returned from 11, select *CONFLATION_ANCHOR_SECTION* from the **TRANSPORT_LINK** table given *TRANSPORT_LINK_ID*. Append the *CONFLATION_ANCHOR_SECTION* to the new conflation object. If the route is being traversed in reverse order make sure the reverse the conflation before appending it.
13. Determine the portion of the anchor section conflation that exists from the start of the transport link to the end offset over the transport link. Append the potion to the new conflation object.
14. Return the conflation object

7.11.4.2.4 Tables and Columns

Table	Column Name
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	FULL_NAME
	ROUTE_SYSTEM_ID
	ROUTE_ID
REFERENCE_FEATURE	REFERENCE_FEATURE_ID
	TRANSPORT_LINK_ID
	TRANSPORT_LINK_OFFSET
	NAME
REF_FEATURE_ROUTE_XREF	REFERENCE_FEATURE_ID
	ROUTE_ID
ROUTE_LINK	ROUTE_ID
	ORDINAL
	DISTANCE
	TRANSPORT_LINK_ID
TRANSPORT_LINK	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
	CONFLATION_ANCHOR_SECTION
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE

7.12 Anchor Section Offset to Literal Description

This algorithm is used to assign a Literal Description value to an anchor section-offset reference.

7.12.1 Point Events

7.12.1.1 Inputs

Parameter	Type	Quantity Type	Description
Anchor Section ID	Number	N / A	Unique ID for an anchor section in the datum.
Offset	Number	Meters	Real valued non-negative offset over the anchor section.

7.12.1.2 Outputs

Parameter	Type	Quantity Type	Description
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
At Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
At Route Name	String	Route Name	Name of the cross route at the originating intersection.
Towards Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Towards Route Name	String	Route Name	Name of the cross route at the destination intersection.
Distance	Number	Meters	Measured length from the reference feature to the location.

7.12.1.3 Algorithm

1. Select *DATUM_DISTANCE* from the **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.* Calculate proportional offset over the anchor section given the meters offset from the start of the anchor section*.*
2. Select *TRANSPORT_LINK_ID* from **TRANSPORT_LINK_AS_XREF** given *ANCHOR_SECTION_ID* and offset calculated from 1 is in the range *FROM_POSITION*..*TO_POSITION.*
3. *Select DISTANCE, TRANSPORT_NODE_ID_FROM, TRANSPORT_NODE_ID_TO* from **TRANSPORT_LINK** given *TRANSPORT_LINK_ID.*
4. Calculate distance offset from the start of the route in meters for the anchor section offset.
5. Select all *ON_ROUTE_ID*, *AT_ROUTE_ID* given *TRANSPORT_NODE_ID* as the from-node of the transport link from the **TRANSPORT_NODE_ROUTE_XREF** table.
6. For each row returned select the combination of *ON_ROUTE_ID* and *AT_ROUTE_ID* that maximizes the preference set of routes.
7. Select *FULL_NAME*, *ROUTE_SYSTEM_ID* from **ROUTE** given *ON_ROUTE_ID* from 6.
8. Select *NAME* from **ROUTE_SYSTEM** given *ROUTE_SYSTEM_ID*.
9. Select *FULL_NAME*, *ROUTE_SYSTEM_ID* from **ROUTE** given *AT_ROUTE_ID* from 6.
10. Select *NAME* from **ROUTE_SYSTEM** given *ROUTE_SYSTEM_ID*.
11. Select all *AT_ROUTE_ID* given *TRANSPORT_NODE_ID* as the to-node of the transport link and *ON_ROUTE_ID* from 6 from the **TRANSPORT_NODE_ROUTE_XREF** table.
12. For each row returned select the combination of *ON_ROUTE_ID* and *AT_ROUTE_ID* that maximizes the preference set of routes.
13. Select *FULL_NAME*, *ROUTE_SYSTEM_ID* from **ROUTE** given *ON_ROUTE_ID* from 12.
14. Select *NAME* from **ROUTE_SYSTEM** given *ROUTE_SYSTEM_ID*.
15. Format a literal description string given On Route Category from 8 and On Route from 7, At Route Category from 10 and At Route from 9, and Towards Route Category from 14 and Towards Route from 13.

7.12.1.4 Tables and Columns

Table	Column Name
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE
TRANSPORT_LINK_AS_XREF	TRANSPORT_LINK_ID
	ANCHOR_SECTION_ID
	FROM_POSITION
	TO_POSITION
TRANSPORT_LINK	TRANSPORT_LINK_ID
	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ON_ROUTE_ID
	AT_ROUTE_ID
ROUTE	ROUTE_ID
	FULL_NAME
	ROUTE_SYSTEM_ID
ROUTE_SYSTEM	ROUTE_SYSTEM_ID
	NAME

7.12.2 Linear Events

7.12.2.1 Inputs

Parameter	Type	Quantity Type	Description
Anchor Section Conflation Object	LRSx Conflation Object	N / A	Conflation of the anchor sections.

7.12.2.2 Outputs

Parameter	Type	Quantity Type	Description
On Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
On Route Name	String	Route Name	Name of the route that the location lies on.
At Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
At Route Name	String	Route Name	Name of the cross route at the originating intersection.
Towards Route Category	Text	N / A	Name of the category of signed route. "STATE", "COUNTY", "E911" etc.
Towards Route Name	String	Route Name	Name of the cross route at the destination intersection.
Distance	Number	Meters	Measured length from the reference feature to the location.

7.12.2.3 Algorithm

16. Select *DATUM_DISTANCE* from the **ANCHOR_SECTION** table given *ANCHOR_SECTION_ID.* Calculate proportional offset over the anchor section given the meters offset from the start of the anchor section*.*

17. Select *TRANSPORT_LINK_ID* from **TRANSPORT_LINK_AS_XREF** given *ANCHOR_SECTION_ID* and offset calculated from 1 is in the range *FROM_POSITION*..*TO_POSITION.*

18. *Select DISTANCE, TRANSPORT_NODE_ID_FROM, TRANSPORT_NODE_ID_TO* from **TRANSPORT_LINK** given *TRANSPORT_LINK_ID.*

19. Calculate distance offset from the start of the route in meters for the anchor section offset.

20. Select all *ON_ROUTE_ID*, *AT_ROUTE_ID* given *TRANSPORT_NODE_ID* as the from-node of the transport link from the **TRANSPORT_NODE_ROUTE_XREF** table.

21. For each row returned select the combination of *ON_ROUTE_ID* and *AT_ROUTE_ID* that maximizes the preference set of routes.

22. Select *FULL_NAME*, *ROUTE_SYSTEM_ID* from **ROUTE** given *ON_ROUTE_ID* from 6.

23. Select *NAME* from **ROUTE_SYSTEM** given *ROUTE_SYSTEM_ID*.

24. Select *FULL_NAME*, *ROUTE_SYSTEM_ID* from **ROUTE** given *AT_ROUTE_ID* from 6.

25. Select *NAME* from **ROUTE_SYSTEM** given *ROUTE_SYSTEM_ID*.

26. Select all *AT_ROUTE_ID* given *TRANSPORT_NODE_ID* as the to-node of the transport link and *ON_ROUTE_ID* from 6 from the **TRANSPORT_NODE_ROUTE_XREF** table.

27. For each row returned select the combination of *ON_ROUTE_ID* and *AT_ROUTE_ID* that maximizes the preference set of routes.

28. Select *FULL_NAME*, *ROUTE_SYSTEM_ID* from **ROUTE** given *ON_ROUTE_ID* from 12.

29. Select *NAME* from **ROUTE_SYSTEM** given *ROUTE_SYSTEM_ID*.

30. Format a literal description string given On Route Category from 8 and On Route from 7, At Route Category from 10 and At Route from 9, and Towards Route Category from 14 and Towards Route from 13.

7.12.2.4 Tables and Columns

Table	**Column Name**
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE
TRANSPORT_LINK_AS_XREF	TRANSPORT_LINK_ID
	ANCHOR_SECTION_ID
	FROM_POSITION
	TO_POSITION
TRANSPORT_LINK	TRANSPORT_LINK_ID
	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
	DISTANCE
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ON_ROUTE_ID
	AT_ROUTE_ID
ROUTE	ROUTE_ID
	FULL_NAME
	ROUTE_SYSTEM_ID
ROUTE_SYSTEM	ROUTE_SYSTEM_ID
	NAME

7.13 Transformation Table Coverage

Table 10 provides a list of the tables and columns accessed by the transformation algorithms.

Table 10 - Table and Columns used by Transformation Algorithms

Table	Column
ANCHOR_SECTION	ANCHOR_SECTION_ID
	DATUM_DISTANCE
	GEOMETRY_GIMS_CARTO
TRANSPORT_LINK_AS_XREF	ANCHOR_SECTION_ID
	TRANSPORT_LINK_ID
	FROM_POSITION
	TO_POSITION
TRANSPORT_LINK	TRANSPORT_LINK_ID
	DISTANCE
	CONFLATION_ANCHOR_SECTION
	TRANSPORT_NODE_ID_FROM
	TRANSPORT_NODE_ID_TO
ROUTE_LINK	ROUTE_LINK_ID
	ROUTE_ID
	TRANSPORT_LINK_ID
	DISTANCE
	ORDINAL
TRANSPORT_SYSTEM_MILEPOINT	ROUTE_LINK_ID
	TRANSPORT_SYSTEM_ID
	ACCUMULATED_BEGIN_DISTANCE

Table	**Column**
	ACCUMULATED_END_DISTANCE
REFERENCE_POST	FULL_NAME
	POST_VALUE
	TRANSPORT_LINK_ID
	TRANSPORT_LINK_OFFSET
	ANCHOR_SECTION_ID
	ANCHOR_SECTION_OFFSET
	ROUTE_ID
HPMS_CONTROL_SECTION	CONFLATION_ANCHOR_SECTION
	CONTROL_SECTION_ID
GIMS_CONTROL_SECTION	CONFLATION_ANCHOR_SECTION
	CONTROL_SECTION_ID
TRANSPORT_SYSTEM	NAME
	TRANSPORT_SYSTEM_ID
ROUTE_SYSTEM	NAME
	ROUTE_SYSTEM_ID
ROUTE	ROUTE_ID
	FULL_NAME
	ROUTE_SYSTEM_ID
TRANSPORT_NODE_ROUTE_XREF	TRANSPORT_NODE_ID
	ROUTE_ID
REFERENCE_FEATURE	REFERENCE_FEATURE_ID
	NAME
	TRANSPORT_LINK_ID
	TRANSPORT_LINK_XREF

Table	Column
	REFERENCE_FEATURE_ CATEGORY-ID
REF_FEATURE_ROUTE_XREF	REFERENCE_FEATURE_ID
	ROUTE_ID

8 APPENDIX B: LRS SUBSYSTEM EVENTS

This appendix describes the events that are generated by each specific subsystem.

8.1 Datum Subsystem Events

The following events can be raised[4] by the Datum Subsystem

8.1.1 DATUM_CHANGE

This event is raised when a new anchor section or anchor point is added to the datum. This is a notification that the datum has been extended and requires that the datum is re-published before the changes can be utilized by other subsystems.

8.1.2 DATUM_MODIFIED

This event is raised when an existing anchor section has been retired. This event requires that all subsystems with data tied to the anchor sections be rationalized.

8.2 Route Subsystem Events

The following events can be raised by the Route Subsystem

8.2.1 NETWORK_CHANGED

This event is raised when a new transport link or transport node is added to the datum. It is also posted when a row is added to the NESTED_NETWORK_XREF table or a link or node has its open/closed state changed. This is a notification that the network has been extended and requires that the network is re-published before the changes can be utilized by other subsystems.

8.2.2 NETWORK_MODIFIED

This event is raised when an existing transport link or transport node has been retired. This event requires that all subsystems with data tied to the transport link or transport node tables be rationalized.

8.2.3 TRANSPORT_SYSTEM_CHANGE

This event is raised when a new transport system is added to the network.

[4] Raising an event is a notification process whereby a subsystem creates an event due to some change in state and passes the event to the system for processing. It is up to the system to determine if and when the event is evaluated, verified and processed. Two basic models for handling events are a) immediate and b) queued. Most systems, the LRS included, will distinguish between the types of event and take immediate action on some types and queue others.

8.2.4 ROUTE_CHANGE

This event is raised when a new route is added to the network. This is a notification that a new route has been added and requires that the route is published before the it can be utilized by other subsystems.

8.2.5 ROUTE_MODIFIED

This event is raised when an existing route is retired, part of an existing route is retired, or a new part of an existing route is created. . This event requires that all subsystems with data tied to the route tables be rationalized.

8.3 Reference Post Subsystem Events

The following events can be raised by the Reference Post Subsystem

8.3.1 REFERENCE_POST_CHANGE

This event is raised when a new reference post is added. This is a notification that additional reference posts are available and requires that the reference posts are re-published before the changes can be utilized by other subsystems.

8.3.2 REFERENCE_POST_MODIFIED

This event is raised when an existing reference post is retired or re-positioned. This event requires that all subsystems with data tied to the reference post table be rationalized.

8.4 Segmental Subsystem Events

The following events can be raised by the Segmental Subsystem

8.4.1 HPMS_MODIFIED

This event is raised when an existing HPMS segment is modified. This event requires that all subsystems with data tied to the HPMS_SECTION table be rationalized.

8.4.2 GIMS_MODIFIED

This event is raised when an existing GIMS segment is modified. This event requires that all subsystems with data tied to the GIMS_SECTION table be rationalized.

8.5 Stationing Subsystem Events

The following events can be raised by the Stationing Subsystem

8.5.1 PROJECT_SECTION_CHANGE

This event is raised when a new project section is added. This is a notification that additional project sections are available and requires that the project sections are re-published before the changes can be utilized by other subsystems.

8.5.2 STATION_POST_CHANGE

This event is raised when a new station post is added. This is a notification that additional station posts are available and requires that the station posts are re-published before the changes can be utilized by other subsystems.

8.6 Coordinate Route Subsystem Events

There are no events associated with the coordinate route subsystem.

8.7 Literal Description Subsystem Events

The following events can be raised by the Literal Description Subsystem

8.7.1 REFERENCE_FEATURE_CHANGE

This event is raised when a new reference feature is added. This is a notification that additional reference features are available and requires that the reference features are re-published before the changes can be utilized by other subsystems.

8.7.2 REFERENCE_FEATURE_MODIFIED

This event is raised when an existing reference feature is retired or re-positioned. This event requires that all subsystems with data tied to the reference feature table be rationalized.