Exor to AgileAssets Maintenance and PMS Interface

Interface Specification

October 2013

Bentley®
Sustaining Infrastructure

Contents

Version Control

Date	Version	Changed by	Notes
February 2012	Draft, 0.1	RE	Initial Revision
May 13, 2013	Draft, 0.2	JM	Addressed KYTC concerns
September 9, 2013	1.2	RE	Addressed KYTC and AA comments.
October 4, 2013	2.0	DG	Accepted all revisions and comments - Issue
January 24, 2014	2.1	JM	Made Minor as Designed Changes

Reference Documents:
Site Visit to KYTC v0.3.docx – Site Visit Report, Richard Ellis, June 2012

Exor to AgileAssets PMS interface.docx - Design of an interface between the Exor product and the AgileAssets PMS, 2007.

Exor to AgileAssets interface Scope and Requirements.docx – Requirements document for the interface between the KYTC Exor system and the AgileAssets System

1.0 Introduction

In May of 2012, Richard Ellis of Bentley Systems visited KYTC to review the current implementation of the Exor product, and to plan future improvements to it. During this visit it was identified that the creation of an Exor to AgileAssets data exchange interface offered a significant opportunity to improve the quality of information held in the AgileAssets system and reduce the effort currently expended keeping some aspects of the two systems coordinated. This will improve the quality of road information held by the state and reduce the cost of duplicated data entry.

KYTC and Bentley Systems have established a project to undertake the scope and requirements analysis, and to create a design for this interface. That analysis resulted in a Scope and Requirements document that has been agreed between Exor and KYTC.

This report is a software specification document that defines the inputs and outputs the software expects. It will be used to ensure everyone has a common understanding of the data that will be received from the AgileAssets Maintenance System, the way this data will be received and the data that will be returned.

2.0 High Level Requirements

The Scope and Requirements document established that the main objective of this project was to create software that could replicate what KYTC currently does manually to update the AgileAssets System with road network information. This includes:

- Provision of a full list of routes
- Addition and removal of routes
- Updating route spatial representation
- Updating the location of 'Event' linear references and spatial locations
- Updating asset information

This interface should allow both the Maintenance and the PMS systems (currently both implemented using software from AgileAssets) to have location of their business objects on the road network locations maintained without the need to make manual changes to the road network information held within the AgileAssets system.

3.0 Road Network Information

3.1 Full List of Routes and Addition and Removal of Routes

The Exor system must provide a current version of the road network information so that the AgileAssets system can take a current set of network data to establish the two systems with an initial, common version of the road network, and to re-synchronize the network at any time in the future.

To make this data available, tables will be created in the Exor database and access provided to the AgileAssets system.   These views allow the identification of new routes and new length on existing routes.

Route Sections Table defines the extent of each route:

View XAA_ROUTE

Column	Null?	Data Type	Comment
ROUTE_ID	N	NUMBER	Route ID
ROUTE_NAME	N	VARCHAR2 (30 Byte)	Route Name
ROUTE_TYPE	N	VARCHAR2 (4 Byte)	Network type
OFFSET_FROM	N	NUMBER (22,4)	From Milepoint
OFFSET_TO	N	NUMBER (22,4)	To Milepoint
EFFECTIVE_DATE	N	DATE	Effective date

Bentley will construct this view using the following logic:

Column	Derivation
ROUTE_ID	The ID of the Route
ROUTE_NAME	The Unique of the Route
ROUTE_TYPE	The Network Type of the route
OFFSET_FROM	Minimum measure on the route section
OFFSET_TO	Maximum Measure on the route section
EFFECTIVE DATE	Date of the most recent change to the route

Notes:

Route Section is a section of route where the roadway is continuous and not broken by a gap in the roadway. This table will hold multiple sections of the same route if the route has gaps in the linear referencing.

KYTC does not have concurrent routes so no tables are planned to support concurrent routes.

The AgileAssets system also requires a table of spatial data. This table will hold one geometry per route and have the following format:

The table name is **XAA_ROUTE_SDO**

Column	**Null?**	**Data Type**	**Comment**
ROUTE_ID	N	NUMBER(38)	Route ID
GEOLOC	N	MDSYS.SDO_GEOMETRY, GTYPE 3306.	Shape as an Oracle spatial geometry

3.2 Route Spatial Representation

Every change in spatial representation of the route must be tracked

A row will be added to the XAA_SPATIAL_AUDIT table for every change. On update, two rows will be added to the XAA_SPATIAL_AUDIT table indicating a ‘delete’ and an ‘add’. The AgileAssets system can use this information to retrieve the appropriate shapes from the route spatial table discussed above.

A reshape is assumed to have an effective date of today.

The XAA_SPATIAL_AUDIT table will have the following structure:

Column	Null?	Data Type	Comment
ROUTE_ID	N	Number(38)	Route ID
GEOLOC	N	MDSYS.SDO_GEOMETRY	Shape as an Oracle spatial geometry
OPERATION	N	VARCHAR2(6)	Either ADD or DELETE
OP_DATE	N	Date	Date the operation occurred
EFF_DATE	N	Date	Start date of the Geometry
END_DATE	Y	Date	End date of the Geometry

4.0 Event Information

4.1 Updating the Location of 'Event' Linear References

The AgileAssets System needs to know if any of the locations of objects stored in the AgileAssets System has changed due to a network operation on the road network. To communicate this, the AgileAssets System will publish a set of historic network locations and a date of those locations. EXOR will respond with the current route locations for those objects.

The format of both the data received from the AgileAssets System and the data returned will be:

Column Name	Null?	Data Type	Comment
HISTORIC_DATE	N	Date	Date of the locations coming from the AgileAssets System (filled by AgileAssets)
LOC_IDENT	N	INTEGER	AgileAssets Location ID # (filled by AgileAssets)
ROUTE_NAME	N	VARCHAR2 (30 Byte)	Route Name **in the form of an Exor System UNIQUE** (filled by AgileAssets)
OFFSET_FROM	N	NUMBER (22,4)	From Milepoint (filled by AgileAssets)
OFFSET_TO	N	NUMBER (22,4)	To Milepoint (filled by AgileAssets)
SOURCE_TABLE	N	VARCHAR2 (32 Byte)	Raw Data Table Name - Informational Item that lists the AgileAssets source data table name
NEW_DATE	N	Date	Date of the locations coming from the Exor System (filled by Exor)
NEW_ROUTE_NAME	Y	VARCHAR2 (30 Byte)	New Route Name (filled by EXOR)
NEW_OFFSET_FROM	Y	NUMBER (22,4)	New From Milepoint (filled by EXOR)
NEW_OFFSET_TO	Y	NUMBER (22,4)	New To Milepoint (filled by EXOR)
PROCESS_MSG	Y	VARCHAR(100)	Messages and Errors produced during the process. Identified possible errors are: "invalid old location" "Route location closed". (filled by EXOR)

The data will be read from one table and written to another. The AgileAssets System will write to a table named **XAA_LOC_IDENT**. Exor will read from this table and write to a table named **XAA_NET_REF**.

AgileAssets system will populate the HISTORIC_DATE, LOC_IDENT, ROUTE_NAME, OFFSET_FROM, OFFSET_TO and SOURCE_TABLE columns.

If the old location has become two new locations separated by a section of road (that may be of zero length) then two, or more, rows will be returned to list all new locations. For example, if the middle mile of a 3 mile route is re-aligned to be 0.2 mile shorter, so that milepoints from 1 and 2 become a new datum from 1 to 1.8 and milepoints 2 through 3 become 1.8 to 2.8, the locations that span the realignment would return two (or more) records: one for the section before the realignment, giving the new network location at that point and one for the section now realigned, with null values for the route, start and end measure, and a possibly a third, for the section after the realignment (if it exists) with the new location of this section.

When the historic location no longer exists (because the road has been closed) , null will be returned for the NEW_ROUTE_NAME, NEW_OFFSET_FROM and NEW_OFFSET_TO and an message inserted into the PROCESS_MSG column with a message similar to “Route location closed”.

If the historic location didn’t exist at the date specified then NEW_ROUTE_NAME, NEW_OFFSET_FROM and NEW_OFFSET_TO will be null and a process message similar to “invalid old location” will be placed in the PROCESS_MSG column.

In some rare cases it will not be possible to establish a new location. In these cases a process message similar to “New location unavailable” will be placed in the PROCESS_MSG column.

To generate the new information, Bentley will construct a process to generate the data required. The interface will perform the following processing:

- The process will read each line of the XAA_LOC_IDENT table and identify if the route has been altered since the date specified. This can be done quickly by checking the XAA_ROUTE table constructed above. If the date of the most recent change in the route is prior to the date specified in the XAA_LOC_IDENT then the route and location have not changed and the information can be written to the XAA_NET_REF table.
- If the route does not exist in the XAA_ROUTE table then the only information written to the XAA_NET_REF table is the information provided and the PROCESS_MSG “Route location does not exist” ”.
- If the date of the most recent change in the route is after the date specified in the XAA_LOC_IDENT table, then the new location information is written to the XAA_NET_REF table.

- If the route does not include the measure specified at the date specified, then the information written to the XAA_NET_REF table is the information provided and the PROCESS_MSG "Invalid old location".
- If the route does not include the measure specified at the the current date, then the information written to the XAA_NET_REF table is the information provided and the PROCESS_MSG "Route Location Closed".

5.0 Asset Information

5.1 Updating Asset Information

Initially there are nine sets of asset information that must be communicated from Exor to the AgileAssets system.   The nine types are:

AL - Auxiliary Lane
FS – Functional Class
LN – Lanes
RA – Adequacy Rating
RW – Right of way Width
SH – Shoulders
SL – Speed Limit
SS – State System
TF – Traffic Count Information

The interface will initially be configured to communicate this set of asset information to the AgileAssets system, but other asset information must be able to be added to the list by changing the configuration of the software and without the need for changes to the application software.

All asset tables will have the same five columns, and then the specific asset attribute columns required. The fixed columns will be:

```
Name                 Null?  Type
-------------------- ------ -------------------------
ROUTE                No     VARCHAR2(50)
ROUTE_TYPE           No     VARCHAR2(4)
FROM_POINT           No     NUMBER
TO_POINT             No     NUMBER
ASSET_ID             No     NUMBER
```

The flexible attribute columns will then be whatever is specified for the interface configuration. For example, The Auxiliary Lane table would have the following format:

```
Name               Null?  Type
------------------ ------ -------------------------
ROUTE               No    VARCHAR2(50)
ROUTE_TYPE          No    VARCHAR2(50)
FROM_POINT          No    NUMBER
TO_POINT            No    NUMBER
ASSET_ID            No    NUMBER
AUXLANE                   NUMBER
```

```
AUXLNWID                  VARCHAR2(50)
AUXSURF                   VARCHAR2(50)
```

6.0 Interface Execution

In the sections above, various parts of the interface have been defined. All of these components must be implemented to complete the interface. There is a small amount of interdependency, so the order is important.

The interface will be executed either by a job that runs it at preset times, or manually, or both. The system administrator at KYTC will be responsible for creating the job based on administration documentation provided by Bentley. Bentley will provide a user interface for the process to be run from inside the Exor application.

All output and input tables will reside in database schema outside of the core Exor data. This will enable the data read from and sent to the AgileAssets system to be separated from core Exor data.

A data link will need to be forged between the AgileAssets system and the Exor system. The direction and mode of this link have yet to be confirmed but by using a separate schema and views reading data across database link(s), this can be defined as required and offsite development and testing will be more practical.

The interface components will be run in the following order:

- Full List of Routes and Addition and Removal of Routes
- Route Spatial Representation
- Updating the Location of 'Event' Linear References
- Updating Asset Information

7.0 Documentation Requirements

Documentation is required to administer and run the interface. It needs to include detail on the configuration of the interface including the configuration of the interface metadata, and detail on how to execute the interface both manually and via a job.

8.0 Conclusion

This document is the result of a series of conversations between Bentley Systems, KYTC and AgileAssets with the objective of establishing an interface between the AgileAssets Maintenance System (including a PMS) and the Exor system KYTC uses to manage the road network and key asset data. From these discussions, Bentley Systems has created a software specification that meets the agreed scope of this project and the requirements that need to be met in order for the project to be successful.

Using this specification, Bentley Systems will develop and deploy software that reads and produces the data specified