

Regional Address Point Repository Synchronization Pilot

A MetroGIS Project for funding year 2007

**Prepared for the MetroGIS Coordinating Committee
December 10, 2008**

Authors:

**Peter Henschel – Carver County GIS Supervisor
Nicole Roepke – Carver County Database Administrator**

Project Summary:

Many organizations have a business need for creating and maintaining address point data. With so many organizations needing address data, would a regional address dataset be a viable solution for the MetroGIS community? In March of 2004 MetroGIS created an Address Workgroup to bring together city, county and regional organizations who share this same interest in creating address point data.

The Address Workgroup created a vision document to help the GIS community understand how a regional address point dataset could exist through cooperation, data sharing and data collection. The vision document can be found at http://www.metrogis.org/data/info_needs/street_addresses/add_wkgrp.shtml. A key component to the success of the regional dataset is a mechanism for the address data repositories to synchronize their data with the regional dataset. Once the synchronization process is setup within an organization, the data updates to the regional repository can be scheduled at any time interval, automating the data uploading process. After the data is synchronized, organizations within MetroGIS can access address data across jurisdictional boundaries knowing they are using data maintained by the address authority.

Through a MetroGIS funded project, Carver County created a methodology for how address point data could be synchronized through a set of tools to allow counties and cities to upload their address point data to a regional dataset. The synchronization process takes the changes from an address point feature class, standardizes the records to conform to a XML Schema that meets the MetroGIS Address Point Specifications, and loads them onto a regional FTP server. A service on the Regional Address Point Repository server will scan the FTP location for files, validate the schema of the file, import the data to the repository, and send an email confirmation.

Table of Contents:

Introduction

Methodology

Standards

Synchronization Process

Explanation of the Functionality Developed

Hardware Specifications

Guidance on Implementing in Different Software or Development

Installation Procedures

Target Users

Lessons Learned

Unresolved Issues

Recommendations for MetroGIS Regional Custodian

Roles and Responsibilities of MetroGIS Regional Custodian

Next Steps

Field Mapping

Data Model

Introduction

Many counties and cities maintain or are in the process of building address point databases either incorporated within GIS or linked to GIS. This address information is useful within entities and to neighboring entities. In order to share address point information in a consistent and universal manner, an XML schema was developed to represent the storage of address data within the Regional Address Point Repository. The XML Schema includes all of the National Street Address Standard fields. It also includes optional fields that are not used by each Address Authority.

Through this synchronization process, address point data will be collected in change sets, converted to an XML file that fits the XML Schema, posted to an FTP location at the Regional Address Point Repository. A service on the Regional Address Point Repository server will scan the FTP location for files, import them to an internal archive location, validate each file against the schema, and finally import the address information into the Regional Address Point Repository Database. Email confirmations can be configured to be sent to those that want confirmation that their data was processed. Additional emails will be sent when data fails validation.

Methodology

- **Standards**

The repository address data standards were derived from the *MetroGIS Address Points Database Specifications* document created by the MetroGIS Address Workgroup. The MetroGIS address standards were based upon the work going on at the national level through URISA.

MetroGIS Address Workgroup

http://www.metrogis.org/data/info_needs/street_addresses/add_wkgp.shtml

URISA Website

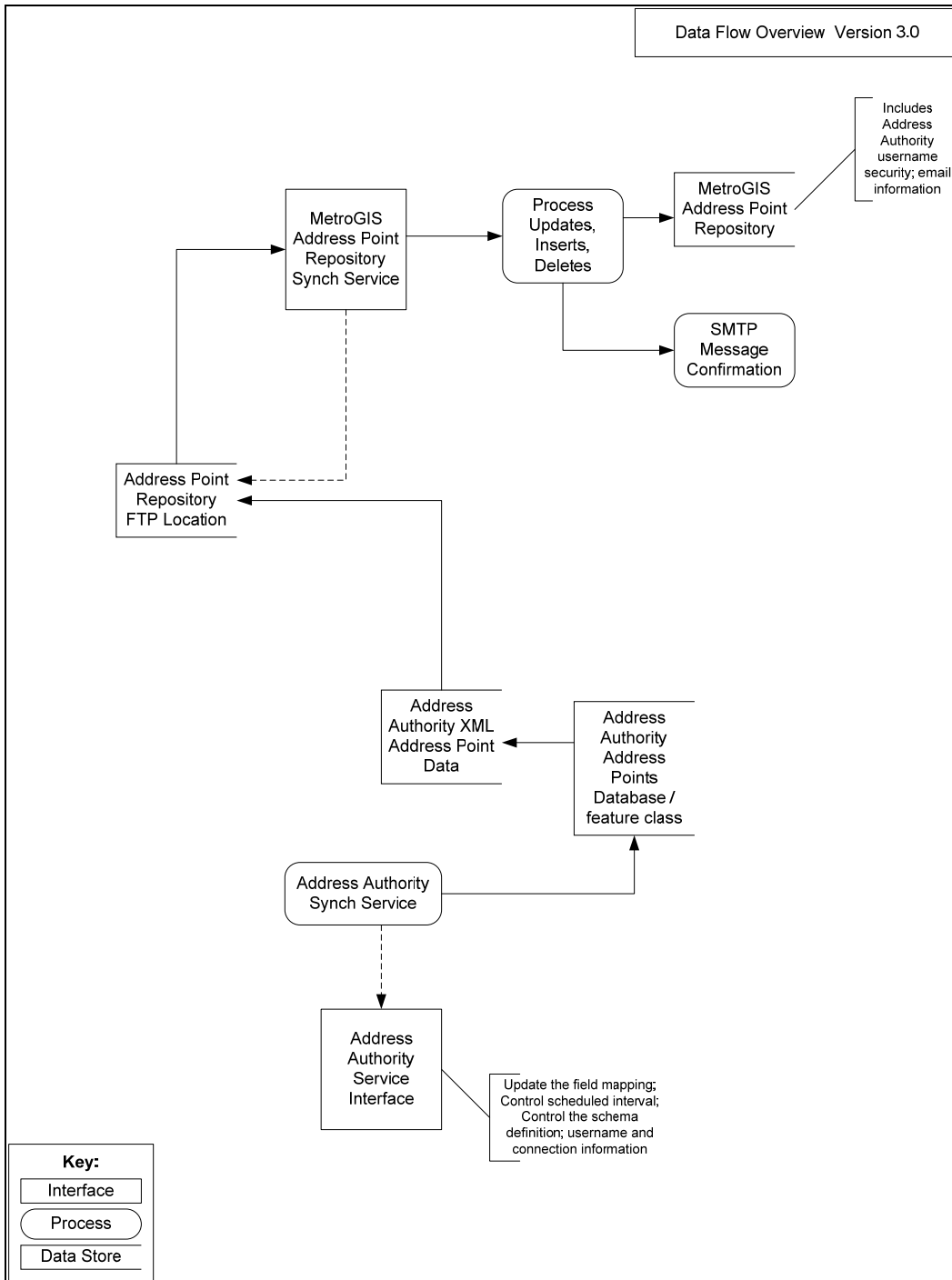
<http://www.urisa.org/about/initiatives/addressstandard>.

- **Synchronization Process**

The synchronization process begins at the Address Authority Synch Service bubble in the diagram below and initiates the selection of records, at the source, that have been changed (included adds, changes, and deletions) since the last synchronization.

This dataset will be collected and output to XML. The synchronization mapping table will be multi-functional, thus providing Address Authorities the potential to re-use the same process to send address change information in a different mapping schema to another destination.

For example, Carver County will be sending data to the Regional Address Point Repository in XML, sending Excel information back to cities within Carver County, and transferring data between division databases at the county.



The XML output will be stored into a file that will be named using the date and a unique code for the Address Authority (GNIS code). This same service will move the file from the file location to the FTP location at the Regional Address Point Repository.

A series of services on a server at the Regional Address Point Repository Host will scan the FTP location for files. When a file is detected, it will be copied to an archive location on the repository server. The archived file will be accessed to verify that it is a valid file. Then the

original file from the FTP location will be moved to a processing directory on the repository server. The processing file will be validated against the XML Schema.

Errors in schema validation will be logged and emailed to the configured contact at that Address Authority. In that situation, the processing file will be deleted from the processing directory. If the schema validation is successful, success will be logged, and data validation will begin.

A row-by-row, field-by-field data validation process will use stored procedures to qualify all the data in the submission and ensure that it conforms to the field standards. For example, the XML schema definition includes the Status field as a single character string or Null. If there is a single character in the Status field, it will conform to the schema. However, the status field only allows A, P, or H. If another letter was in the field, that row of data would be a validation exception. Each validation exception is moved from the processing table into an exceptions table. After the validation has been completed, the synchronization processing will begin on the remaining good rows in the processing table. Exceptions will be emailed to the Address Authority so that changes can be made and rows collected in the next submission.

Synchronization processing will involve importing of the data from the XML processing file into a preliminary processing table. From this table, separate stored procedures will be used to update, append, and deactivate records (based on the unique primary key starting with the Address Authority's GNIS code) in the Address Point Repository Database.

This processing will occur within a transaction so that if one portion of the synchronization fails, all changes to the Address Authority's dataset will be rolled back. If there are no errors, the transaction will be committed. A synchronization success SMTP email message will be sent to the Address Authority's configured contact(s).

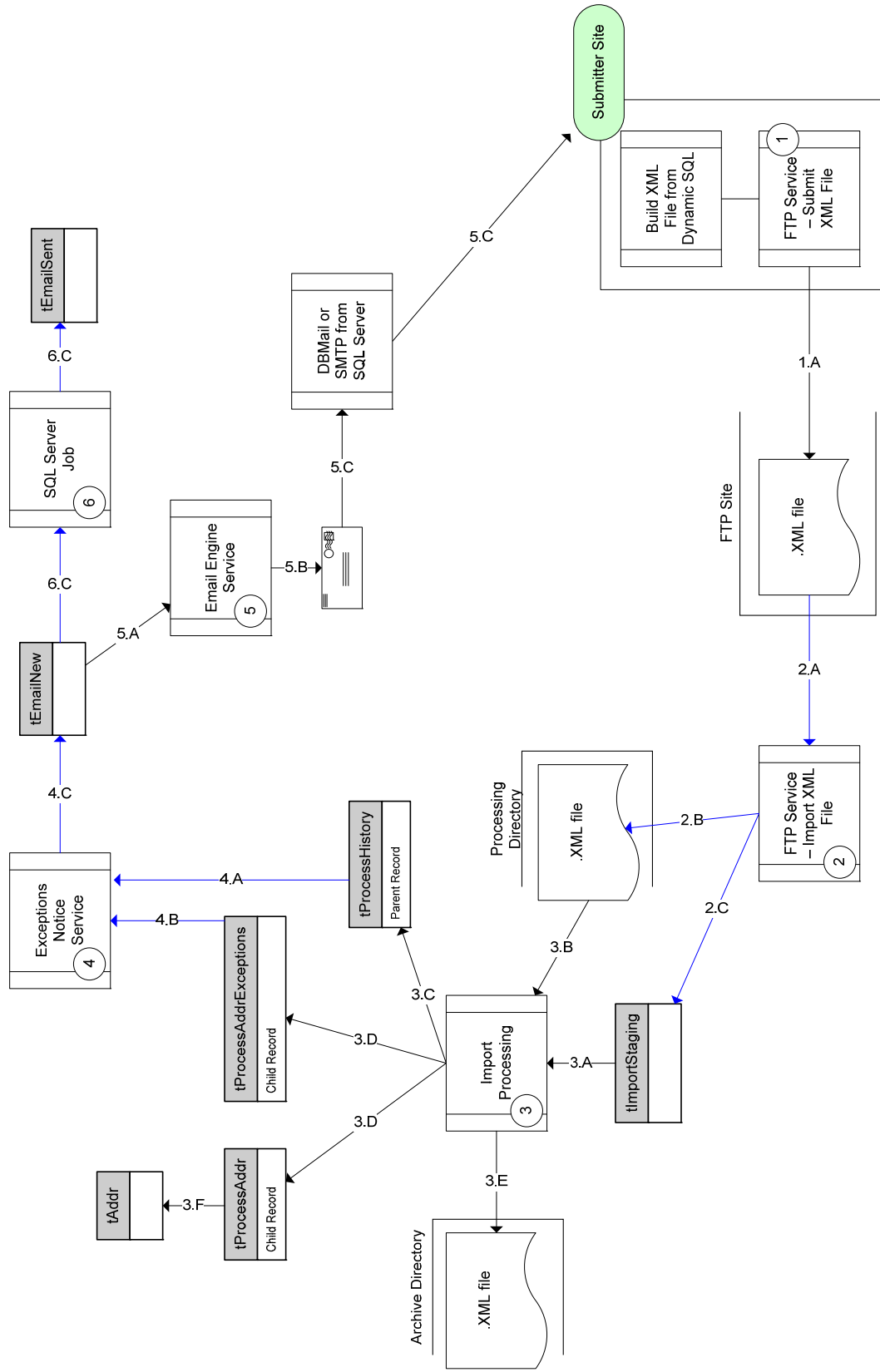
- **Explanation of the Functionality Developed**

There have been 5 windows services, a SQL Job, and two test configuration Windows applications created that handle the Address Synchronization from start to finish. Each piece is listed below. Refer the Service Interaction Overview diagram in this section for a visual of how these services interact with each other and the database.

1. Address Authority Submit Service

The submission service is made up of a Windows service which calls stored procedures that collect settings information (from its own database) and address point data (from an ArcSDE database) to generate a dataset and from that an XML document. The XML document is stored in a local network directory so that there is a record of the files submitted to the repository. The generated file is placed on the Address Repository FTP site using a username/password pair configured for this submitter.

Service Interaction Overview Version 2.0



2. FTP Import Service AND

3. Import Processing Service

The import processing begins with a list of the files found on the FTP site. The files are put in order by name (it has the date and time of submitter creation) to keep the data updates in the correct order. The files are moved from the FTP site to a local (network) processing directory and a row is added to the import staging table. After all the files found are moved into the processing directory, then the import module is launched. The import module loops through the files in the import staging table, performing these actions on each before moving to the next file.

An xml file read and the XML schema validated. Schema validation failures will be logged to the Address Repository database log table and a row will be added to the Process History table noting the failure status. The failed file will be deleted from the processing directory. If the xml schema validation succeeds, a row will be added to the process history table noting the success status and the next steps are executed.

Address point rows from the xml document will be pushed into a processing table. From there a number of validation stored procedures will run against the processing table to filter out bad data rows. If a row fails data validation, then it will be moved to the processing exceptions table and the data validation failure reason noted. The processing of good data rows will not be held up by a few bad rows.

After the data validation has been completed, the remaining rows in the processing table are ready to be combined with the true address repository data. Inserts, updates, and deactivations (another kind of update) will be handled based on matching of the unique address point id that is attached to each row.

When these steps are complete, the module loops back to the next file in the list until all processing for this set has been complete. At which time, it lays dormant until the wait interval has elapsed.

4. Notification Service

The Notification Service creates email notifications based on the Process History and the Process Exceptions tables and inserts rows into the Email New table where the notices are picked up by the email service for delivery.

5. Email Engine Service

The Email Engine Service pulls messages from the Email New table, creates the SMTP message format and submits the emails to an SMTP relay. It can also work with SQL DBMail for SQL 2005 servers. Emails that are sent get an email send date and time stamp updated to the row. This service runs at a higher frequency than the other services. And any email rows inserted into the Email New table will be sent. For example, a trigger exists on the Address Repository Log table that inserts rows into the Email New table when error messages of a certain severity are logged.

6. SQL Server Job

This job just executes a stored procedure that will move send emails to the email sent folder. It could also have steps added to it to clean up or archive other information such as purging process history after so many days or months, etc.

- **Installation Procedures**

The complete installation and configuration documentation is being held until an Address Point Repository Host is found. There may be alterations in the documentation specific to the host. (See Next Steps for details.) After the Address Repository host has been configured, Carver County will point our submitter services to it to validate the implementation. At that point, detailed configuration information will be communicated to the participating submitters and turned over as a resource.

Each service has an installation package that can be executed by hand or via a script. The final compiled installation packages will ship (or be available via download) with the above mentioned implementation documentation. All of the services use a config file to store the database connection information. The service interval settings are all stored in the database. The respective databases will be deployable through SQL server script.

Submitters are encouraged to ask questions and give feedback to Carver County during the initial installation period (anticipated to be within two months of the Host configuration).

- **Hardware Specifications**

Repository & Submitter Site:

- SQL Server
 - You should be able to use your existing SQL Server where SDE data is stored.
 - SQL Server 2000 sp4 or SQL Server 2005 + Standard Edition 32-bit on Windows 2000 or 2003 (respectively) 32-bit
- Windows Application Server
 - You may or may not need this depending on if your organization allows service applications to run on your SQL server.
 - Windows Server 2000 or 2003 Standard Edition 32-bit
 - .NET Framework
 - Latest Microsoft Updates and Hot Fixes

Submitter Site Only:

- ESRI ArcSDE 9.2 – Point Feature Class in a Geodatabase

- **Target Users**

Address Authorities who would like to participate in the regional address point repository. The anticipated users of the synchronization process are counties or cities within MetroGIS. The counties and cities need to be in agreement of who is going to supply the address data to the regional repository.

- **Guidance on Implementing in Different Software or Development**

Carver County utilized Microsoft SQL Server and Visual Basic .NET to build the repository, but this does not limit other systems from implementing the synchronization. A solution could be built using the ideas within Carver County's process to produce the same XML file that is posted to the regional FTP server. The Regional Repository could consume the XML file and function the same way as if they provided the information through the process Carver County has proposed.

Lessons Learned

The solution needs to be as flexible as possible within the same processes. With the use of services, there is no visibility to the working of the application – a test Windows Forms application was built to allow user and admin visibility into the testing and function of the services. Logging is very important and should not be overlooked as a first stop for troubleshooting.

Recommendations for MetroGIS Regional Custodian

- Separate the FTP server from the SQL server.
- Consider the service interval scheduling carefully to minimize wasted processing time. For example, if submitters are generally only ending data up once a day, it will not be necessary to configure the file import processing engine to run every 5 minutes.
- Periodically check that the files you see in the archive directory have indeed been listed in the tProcessHistory table, and that tEmailSent contains confirmation emails sent to the Address Authority whom submitted the original file.
- Understand the logging functions and the data stored in the tLog table. Create a couple of quick views to filter the log table by severity type or date so that you can quickly get a picture of how the services are performing. Install the tLog table trigger that notifies a specified email address when level 9 errors are logged to the table.
- Make sure that only one authority is submitting data for an area. If a city is submitting data and so is the county in which the city is located is also submitting data, there must be coordination on behalf of both parties BEFORE configuring the city into the repository. There should only be one submitter for each point. If both submitters have a single point in their databases and both submit it, there will be two points at that location in the repository. For example, if the City of Chanhassen submits their own data to the repository, then Carver County will exclude any points within Chanhassen from the data that is submitted from here. There is a place in the Where Clause of the configuration to handle this occurrence.
- Web Services should only be allowed to read data, unless a web editor takes the place of one or more submitters. Again, address points should have one originator. A submitter sending up data points would miss any updates (to the same key numbered row) that were created by a separate web service and has the potential for data loss.

Roles and Responsibilities of MetroGIS Regional Custodian

One-time Responsibility

- Configure SMTP mail server or have existing account information available (can use existing SMTP or Exchange server)
- Create secure FTP site for document submittals (not on the SQL server)
- Designate an admin contact to receive any critical logged application failure notifications
- Participate in repository configuration to include
 - Application installation
 - Database configuration
 - Security account creation and configuration
 - 1 hour training on file locations, error log review, and troubleshooting

Ongoing Responsibilities

- Setup of Address Submitter (including awareness of area overlaps)
- Weekly log review
- Respond to critical application failure or submittal questions
- Monitor available file storage space
- Monitor available db storage space
- File share and database backups

Next Steps

The synchronization process has been tested within Carver County's environment, but currently there is no regional host identified. In order for cities and counties to use the synchronization, a regional host must be found.

When the host is determined, Carver County will implement the repository at the host site with the assistance of the host's staff. The installation and configuration documentation will be updated base on the host environment and configuration settings and then will be communicated to the participating submitters and turned over as a resource to the group.

Carver County will point our Address Submitter Service to the host to validate the host implementation and the submitter implementation. Any final documentation or installation package adjustments will be made and then the following items turned over to the host, participating submitters, and the MetroGIS Council.

Final version uncompiled source code solution with all projects

Final version compiled services and respective deployment packages

Final version compiled testing application in a deployment package for submitters

Final version compiled testing application in a deployment package for host

Field Mapping

The examples of mapping spreadsheets should be used as living documents. This is a starting place that can make updating the values in the configuration tables easy and be the first place you come to update the data mapping if you address feature class changes. See Attachment A.

Data Model

Address Repository and Local Address Source databases. See Attachment B.

Attachment B – AddrSbmtrSrc database

tLookupCounty	
PK	<u>CountyID</u>
U1	CountyCode
U2	CountyDescr CountyGISName

tLookupLifecycleStatus	
PK	<u>LifecycleStatusID</u>
U1	LifecycleStatus LifecycleStatusDescr

tLookupException	
PK	<u>ExceptionID</u>
U1	ExceptionDescr ExceptionShortDescr

tLookupProcessStatus	
PK	<u>ProcessStatusID</u>
U1	ProcessStatusDescr

tLookupMailable	
PK	<u>MailableID</u>
U1	MailableDescr MailableMETROGIS

tLookupStreetType	
PK	<u>StreetTypSDEcode</u>
	StreetTypMETROGIS

tAddrFieldMap	
PK	<u>FieldMapID</u>
	DestinationDescr DestinationFieldName DestinationDataType DestinationTypeLen XMLFieldNameAlias SourceField SourceFieldJoinCondition ZLStoNULLConversion DestinationFieldOrder

tLookupErrorSeverity	
PK	<u>SeverityID</u>
U1	SeverityDescr

tLookupMunicipality	
PK	<u>MunicipalityID</u>
U2	MuniCode
U1	MuniName

tLookupStreetDirectional	
PK	<u>StreetDirSDEcode</u>
U1	StreetDirMETROGIS

tLog	
PK	<u>LogID</u>
FK1	LogDate LogLoggedInUser LogSource LogVersion <u>LogSeverityID</u> LogMessage LogStack LogSQLParams

tAddrSubmitHistory	
PK	<u>AddrSubmitHistoryID</u>
	AddrSubmitDestinationDescr AddrSubmitCompleted AddrRecordedLastSubmittedDT

tAddrSetting	
PK	<u>AddrSettingID</u>
U1	AddrSettingName AddrSettingValue



Attachment B – AddrRpstr Database

tAddr	
PK	<u>AddrID</u>
U1	SubmitterAddrID aNumberPre aNumber aNumberSuf aNumberSep ST_Pre_Mod ST_Pre_Dir ST_Pre_Typ ST_Name ST_Pos_Typ ST_Pos_Dir ST_Pos_Mod Occu_Type1 Occu_ID1 Occu_Type2 Occu_Id2 Muni_Code Muni_Name USPS_Place Co_Code Co_Name State_Code Zip Zip4 Loc_Decr Landmark Mailable Status PIN Longitude Latitude Posi_Accu aDirSource aAuthority Edit_Org UpdateDate Comments

tProcessAddr	
PK	<u>ProcessRowID</u>
	ProcSubmitterAddrID ProcANumberPre ProcANumber ProcANumberSuf ProcANumberSep ProcST_Pre_Mod ProcST_Pre_Dir ProcST_Pre_Typ ProcST_Name ProcST_Pos_Typ ProcST_Pos_Dir ProcST_Pos_Mod ProcOccu_Type1 ProcOccu_ID1 ProcOccu_Type2 ProcOccu_Id2 ProcMuni_Code ProcMuni_Name ProcUSPS_Place ProcCo_Code ProcCo_Name ProcState_Code ProcZip ProcZip4 ProcLoc_Decr ProcLandmark ProcMailable ProcStatus ProcPIN ProcLongitude ProcLatitude ProcPosi_Accu ProcADirSource ProcAAuthority ProcEdit_Org ProcUpdateDate ProcComments

tProcessExceptions	
FK2	ProcExcepID
FK1	ProcessHistoryID
	ExcepID ExcepSubmitterAddrID ExcepANumberPre ExcepANumber ExcepANumberSuf ExcepANumberSep ExcepST_Pre_Mod ExcepST_Pre_Dir ExcepST_Pre_Typ ExcepST_Name ExcepST_Pos_Typ ExcepST_Pos_Dir ExcepST_Pos_Mod ExcepOccu_Type1 ExcepOccu_ID1 ExcepOccu_Type2 ExcepOccu_Id2 ExcepMuni_Code ExcepMuni_Name ExcepUSPS_Place ExcepCo_Code ExcepCo_Name ExcepState_Code ExcepZip ExcepZip4 ExcepLoc_Decr ExcepLandmark ExcepMailable ExcepStatus ExcepPIN ExcepLongitude ExcepLatitude ExcepPosi_Accu ExcepADirSource ExcepAAuthority ExcepEdit_Org ExcepUpdateDate ExcepComments

tAddrAuth	
PK	<u>AddrAuthID</u>
U1	AddrGNIS AddrShortDescr AddrDescr AddrAuthWantEmail AddrAuthEmail

tProcessHistory	
PK	<u>ProcessHistoryID</u>
FK1	<u>ProcessOwnerGNISID</u>
-	<u>ProcessHistoryDT</u>
FK2	<u>ProcessStatusID</u>

tLookupException	
PK	<u>ExceptionID</u>
U1	<u>ExceptionDescr</u>
-	<u>ExceptionShortDescr</u>

tEmailSent	
PK	<u>EmailSentID</u>
	EmailSentTo EmailSentFrom EmailSentCC EmailSentSubject EmailSentMessage EmailSentSentDate

tEmailNew	
PK	<u>EmailID</u>
	EmailTo EmailFrom EmailCC EmailSubject EmailMessage EmailSentDate

tLookupProcessStatus	
PK	<u>ProcessStatusID</u>
U1	<u>ProcessStatusDescr</u>

tLookupErrorSeverity	
PK	<u>SeverityID</u>
U1	SeverityDesc

tImportStaging	
PK	<u>ImportID</u>
-	<u>ImportAuthID</u>
-	<u>ImportDT</u>
-	<u>ImportAuthDT</u>
-	<u>ImportXMLFullPath</u>

tLog	
PK	<u>LogID</u>
FK1	LogDate LogLoggedInUser LogSource LogVersion LogSeverityID LogMessage LogStack LogSQLParams

tSetting	
PK	<u>SettingID</u>
U1	SettingName SettingValue

