NCSLSO

NORTH CAROLINA SURPLUS LINES STAMPING OFFICE

INSURER BATCH UPLOAD GUIDE

TABLE OF CONTENTS

1 DOCUMENT METADATA

Section 1, Document Metadata, contains information about this document. Specifically, the Document Metadata section contains the author(s) (section 1.1.), the intended audience (section 1.2.), the glossary of terms and acronyms (section 1.3.), and the document revision history (section 1.4.).

1.1 Authors

Brent Thompson, Programmer Analyst | Infinity Software Development, Inc. | (850) 383-1011 | ThompsonB@infinity-software.com | 1901 Commonwealth Lane, Tallahassee, FL 32303

Seann Taylor, Business Process Analyst | Infinity Software Development, Inc. | (850) 383-1011 | TaylorSe@infinity-software.com | 1901 Commonwealth Lane, Tallahassee, FL 32303

Tiffany Maruniak, Product and Business Development Manager | Florida Surplus Lines Service Office | (800) 562-4496 | tmaruniak@fslso.com | 1441 Maclay Commerce Drive, Suite 200, Tallahassee, FL 32312

1.2 Intended Audience

The executive summary is intended for management and business users interested in understanding the methods for automated submission of policy batch data to the North Carolina Surplus Lines Stamping Office (NCSLSO) via the Surplus Lines Information Portal (SLIP).

The remainder of this document is intended to be read and used by technical staff seeking to develop and implement one of the automated submission methods for an agency management system that will be interacting with SLIP to submit policy data to NCSLSO.

1.3 Glossary of Terms and Acronyms

Term or Acronym	Definition or Expansion
Agent	Term may refer to any Agent, including IPC Agents.
Batch	A group of policies that will be submitted to NCSLA. A batch may have any number of policies of any type from any insurer.
NCSLSO	North Carolina Surplus Lines Stamping Office
NCSLA	North Carolina Surplus Lines Association
Insurer	Person or company that underwrites an insurance risk and; entity submitting the batch

Term or Acronym	Definition or Expansion
ISD	Infinity Software Development, Inc.
RAPID	Regulatory Administration Platform of Insurance Data; RAPID is an internal platform that allows surplus lines office staff to review submitted policies.
SLAS	Surplus Lines Automation Suite; SLAS is a suite of two software applications designed to process policy submission data for the non-admitted insurance market. SLAS is comprised of the Surplus Lines Information Portal (SLIP) and the Regulatory Administrative Platform for Insurance Data (RAPID).
SLIP	Surplus Lines Information Portal; SLIP is an external portal that allows entities in the non-admitted insurance market to submit policy data to their regulating entity.
Web Service	A software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically Web Services Description Language). Other systems interact with the web service in a matter prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards[1].
XML	eXtensible Markup Language; XML is a self-descriptive markup language designed to carry data. XML has no pre-defined tags. Instead, XML provides the structure that allows users to create their own tags.
XML Batch	An XML batch file uploaded by the user via the SLIP interface or via Web Services API

[1] Definition from the W3C Web Services Glossary located online at: http://www.w3.org/TR/ws-gloss/

1.4 Document Revision History

Author	Date	Version	Description
Brent Thompson	4/12/2016	v0.1	First Draft
Design Farm	11/18/2016	v1.0	Updated document styles

2 EXECUTIVE SUMMARY

The Surplus Lines Automation Suite (SLAS) is a groundbreaking solution for the regulation of surplus lines and is the premier automated filing and regulatory system for the surplus lines insurance industry. The Surplus Lines Information Portal (SLIP), one of the main components of SLAS, allows brokerages, brokers, and insurers to submit policy information electronically. SLAS is the solution of choice for regulating over one third of the surplus lines premium in the United States.

SLAS was developed by the Florida Surplus Lines Service Office, which has been using SLIP for electronic entry of policy data since December 2005.

The XML Batch Upload feature in SLIP is useful for agents and vendors of third party agency/agent management software. SLIP integrates seamlessly with agency/agent management systems while providing accurate data validation and reducing the need for duplicate data entry. Agents will not need new software to submit via the XML Batch Upload feature in SLIP.

This document contains the technical information necessary to configure your software for automation. There are two ways to submit batches automatically: manual file upload and API submission. The standard way to automate batch submission is to configure your product to export the policy data as XML so it can be uploaded manually as a single file. You can take automation further by using API submission for complete integration.

The remainder of this document summarizes the two automated batch submission methods and provides the technical details necessary to implement each method.

3 BATCH CREATION GUIDELINES

Regardless of whether you decide to implement the manual file upload or the API batch submission method, there are some common guidelines for automated batch submission. This section provides the common guidelines and requirements for batches.

3.1 NCSLSO Batch Filing Guidelines

Batches must be submitted in the XML format specified in by the XML schema (see section 3.1.9. Table of XML Fields). No additional files may be included with the batch submission. Batches may be submitted as frequently as necessary. There are no requirements or restrictions on the number of policies that may be included in a batch.

3.2 Batch Definition

The batch file is a single XML file containing policy data in a predefined format. Batches may contain policies of any type.

3.3 Batch File Size

XML batch files are limited to 25 MB in size.

3.4 Batch File Name

The file name is limited to 200 characters. There is no required naming convention, however, it recommended that you create filenames that make it easy to maintain and track your submissions. We suggest that you include the submission date and time in the file name. For example, 20100501_0930_Batch.XML (date_time_Batch.XML or CCYYMMDD_HHMM_Batch.XML) would indicate the batch was created on 05/01/2010 at 9:30 AM.

3.5 HTML (XML) Encoding

Several special characters are reserved and cannot be used directly in XML element or attribute data. Replace them with XML Entity references or XML Encoded text. These special characters act as flags to the parser; they delimit the document's actual content and tell the parser to take specific actions. These special characters, therefore, must be represented in their encoded format:

Character Name	Reserved Character	Entity Reference
Ampersand	&	&amp;
Apostrophe	‘	&apos;
Quote	“	&quot;
Less Than	<	&lt
Greater Than	>	&gt;

3.6 Create a Batch File

The creation of the batch file will require the involvement of a technical resource that is familiar with XML and the data management system in use by your organization. There are several different data management systems in use by insurer throughout the country; therefore, this document cannot provide step-by-step instructions on how to extract policy data from your specific data management system. Rather, this document identifies the structure and formatting requirements of the batch submission in its final form.

The first step in the creation of the batch file is to identify the criteria in which policy data should be extracted from the insurer’s data management system. Typically, insurers extract data based on a specified date range or some other criteria indicating a submission to NCSLSO is required.

Once the criteria to extract policy data is identified for your data management system, a technical resource must create the XML file that contains the policy data. For details on the required format and structure of the XML file, please refer to Section 3 – Batch Creation Guidelines. An XML schema will be provided upon request. The XML schema identifies technical constraints on the content and structure of the XML file and can be used to validate the XML file prior to submission.

Once the XML file is created and/or extracted from the data management system, the files should be submitted as a single XML file to NCSLSO.

Note: The system will not accept Microsoft Excel files saved as XML Data or XML Spreadsheet file types. Please follow the XML format described in this document and identified within the XML Schema to create the XML file.

3.7 Batch File Validation

This section describes the set of validations performed during the batch submission. If the document fails ANY of the validations identified below, the ENTIRE batch file will be rejected.

1. Parse the document and check that the document is well-formed.

2. Check the XML file document against the XSD (XML Schema Definition) file.
    a. Check the length of all data elements to ensure they do not exceed maximum lengths.
    b. Check that values of the specified elements comply with the detailed XML document requirements and the XML schema.
3. Check for a valid broker or brokerage license number
4. Check for valid NAIC number.
5. Accept and/or reject the batch. An email will be sent to the submission contact to confirm the acceptance or rejection of the batch. If the batch has been rejected, the user must correct the batch and resubmit it.

4. TABLE OF XML FIELDS

The tables below outlines the specific values for the elements in an XML schema[2] that is prepared for submitting batch information to NCSLSO. In the table, every element is individually addressed, and a sample of the XML Structure is provided. The XML Structure provides an example of the hierarchy and structural format that the submitted XML will be validated against. The XML structure is followed by a brief description of the element and the element's occurrence and length requirements. The schema is as follows:

4.1 Insurer Schema

This schema is to be used by insurers submitting policies; covering both policies and transactions for Brokers, Brokerages, and IPC filings.

[2] The XML structures are under development and may have changed since the publication of this document. Please contact the NCSLSO at contact@ncsla.com.

Insurer Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Batch Data Set SchemaVersion="1.0" Quarter="4" Year="2011" ReportingState="NC" SubmissionType="INS">	Root element of the XML file. Quarter and year attributes denote the quarter and year of the policies for which you are filing.	1	1	-	-
<Insurer>	Insurer Details for the batch	1	1	-	-
<NAICNumber>AA1234567</NAICNumber>	Insurer NAIC number	1	1	10	10
<Name>Bob's Insurance</Name>	Insurer name	1	1	1	75
<Contact>	Contact	1	1	-	-
<FirstName>Jane</FirstName>	Contact First name	1	1	1	50
<Middle Name>Sarah</Middle Name>	Contact Middle name	0	1	0	30
<LastName>Smith</LastName>	Contact Last name	1	1	1	50
<Name Suffix>Name Suffix1</Name Suffix>	Contact Name Suffix	0	1	0	30
<EmailAddress>EmailAddress@domain.com</EmailAddress>	Contact email address	1	1	5	50
<ContactAddress>	Starting Element				
<Address>Address1</Address	Contact Street Address	1	1	1	75
<Address2>Address21</Address2>	Contact Street Address 2	1	1	0	50
<City>City1</City>	Contact City	1	1	1	50
<StateCode>AL</StateCode>	2 letter state abbreviation	0	1	2	2

Insurer Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Province>AL</ Province >	Province, if a country other than USA	0	1	1	30
<PostalCode>32309</PostalCode>	Contact Zip Code	0	1	5	9
<Country Code>USA</Country Code>	Contact Country Code	1	1	1	30
</ContactAddress>	Ending Element				
<PhoneNumber>	Starting Element				
<Country Code>1</Country Code>	Only required if outside USA	0	1	0	5
<Area Code>888</Area Code>	Area Code	1	1	3	3
<Prefix>123</Prefix>	Phone prefix number	1	1	3	3
<Line>1234</Line>	Phone line number	1	1	4	4
<Extension>12345</Extension>	Extension number if needed	0	1	0	5
</PhoneNumber>	Ending Element				
<Fax>8881234567</Fax>	Fax Number	1	1	10	10
</Contact>	Ending Element				
</Insurer>	Ending Element				
<Brokers>	Starting Element	1	1	-	-
<Broker Xml_BrokerID="0">	Starting Element (The Attribute “XML_BrokerId” must be unique within the	1	1	-	-

Insurer Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
	submission)				
<License Number>123456789</License Number>	Broker License Number	1	1	5	10
<FirstName>Robert</FirstName>	Broker First Name	1	1	1	50
<LastName>Doe</LastName>	Broker Last Name	1	1	1	50
<Policies>	Starting Element List of policies for a given broker for this group of filings				
<Policy Xml_PolicyID="0">	Starting Element (The Attribute "XML_PolicyId" must be unique within the submission)	1	Un-bound	-	-
<Policy Number>ABC1234-1</Policy Number>	Policy number	1	1	1	50
<EffectiveDate> 2014- 01-31 </EffectiveDate>	Policy effective date; (YYYY-MM-DD)	1	1	-	-
<ExpirationDate> 2015- 01-31 </ExpirationDate>	Policy expiration date; (YYYY-MM-DD)	1	1	-	-
<InsuredName>John Doe</InsuredName>	Name of insured	1	1	1	75
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element (The Attribute "XML_TransactionID" must be unique within the submission)	1	Un-bound	-	-
<EffectiveDate>2014-01-31 </EffectiveDate>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<Premium>1234567.89 </Premium>	Transaction Premium	1	1	1	10

Insurer Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</Brokers>	Ending Element				
</Brokers>	Ending Element				
<Brokerages>	Starting Element				
<Brokerage Xml_BrokerageID="0">	Starting Element (The Attribute “XML_BrokerageId” must be unique within the submission)	1	1	-	-
<License Number> 987654321</License Number>	Brokerage License Number	1	1	5	10
<Name>Doe</Name>	Brokerage Name	1	1	1	75
<Policies>	Starting Element List of policies for a given broker for this group of filings				
<Policy Xml_PolicyID="0">	Starting Element (The Attribute “XML_PolicyId” must be unique within the submission)	1	Un-bound	-	-
<Policy Number>ABC1234-1</Policy Number>	Policy number	1	1	1	50
<EffectiveDate> 2014- 01-31 </EffectiveDate>	Policy effective date; (YYYY-MM-DD)	1	1	-	-

Insurer Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<ExpirationDate> 2015- 01- 31 </ExpirationDate>	Policy expiration date; (YYYY-MM-DD)	1	1	-	-
<InsuredName>John Doe</InsuredName>	Name of insured	1	1	1	75
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element (The Attribute "XML_TransactionID" must be unique within the submission)	1	Un-bound	-	-
<EffectiveDate>2014-01-31 </EffectiveDate>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<Premium>1234567.89 </Premium>	Transaction Premium	1	1	1	10
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</Brokerage>	Ending Element				
</Brokerages>	Ending Element				
<IPC >	Starting Element				
<Policies>	Starting Element List of policies for a given broker for this group of filings				

Insurer Schema

XML Structure	Description	Occurrence		Length	
			Min	Max	Min	Max
<Policy Xml_PolicyID="0">	Starting Element Attribute "XML_PolicyId" must be unique within the submission)	1	Un-bound	-	- (The
<PolicyNumber>ABC1234-1</PolicyNumber>	Policy number	1	1	1	50
<EffectiveDate> 2014- 01- 31 </EffectiveDate>	Policy effective date; (YYYY-MM-DD)	1	1	-	-
<ExpirationDate> 2015- 01- 31 </ExpirationDate>	Policy expiration date; (YYYY-MM-DD)	1	1	-	-
<InsuredName>John Doe</InsuredName>	Name of insured	1	1	1	75
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element Attribute "XML_TransactionID" must be unique within the submission)	1	Un-bound	-	- (The
<EffectiveDate>2014-01-31 </EffectiveDate>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<Premium>1234567.89 </Premium>	Transaction Premium	1	1	1	10
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</IPC >	Ending Element				
</Batch Data Set>	Ending Element				

a. Additional XML Information

XML creation software may help you examine and work within the parameters of the XML schema. These tools include Liquid XML Studio, Stylus XML Studio, XML Spy, and others. XML creation software will also validate your file prior to submission.

The following websites contain valuable information regarding the XML Standard and the UCC XML Standard, as well as some information concerning XML tools.

- http://www.w3.org/XML
- http://www.xml.org
- http://msdn.microsoft.com/xml/default.asp
- http://www.xml.com
- http://www.w3schools.com/xml/default.asp
- http://www.w3schools.com/Schema/default.asp

5. MANUAL BATCH FILE UPLOAD

a. Description

The manual batch file upload method allows an insurer to submit policy and transaction data for multiple policies at once in a single batch process. This process will especially benefit insurers that file a large amount of policy data with NCSLSO since a single XML file may contain information for multiple policies.

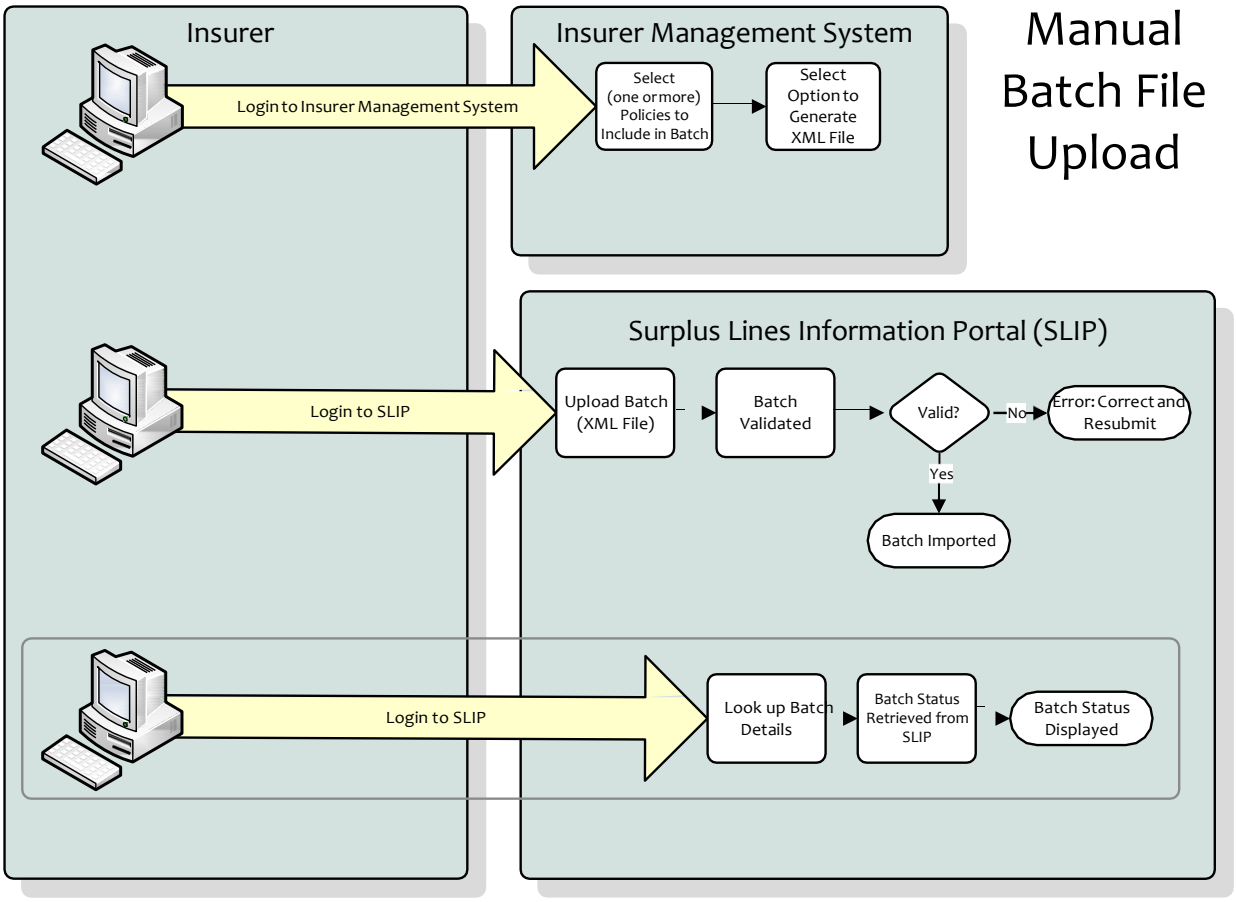
Insurers that store data in a centralized data management system can make use of the manual batch file upload method. The following list provides a high-level list of the steps contained within the manual batch submission process:

i. The insurer will generate an XML file containing the policy data they wish to submit to NCSLSO. Typically, the XML file will include policy data that was added or modified within a specified date range or since the last XML batch submission.

ii. The insurers will log in to SLIP to upload the XML.

b. Process

This section identifies the steps required to create and submit policy information using the manual batch file upload process. The graphic below is a high-level representation of the process flow, and indicates the systems involved.



Create Batch File

The first step in the manual batch file upload process is to create the batch file. Refer to section 3.1.6. of this document for details about creating the batch file.

For the manual batch file upload process, the Insurer Management System may be configured to create the XML batch file. If it is not configured this way, the user may be required to manually create the XML file.

Log in to SLIP

Using a supported web browser, go to the NCSLA SLIP website (http://www.slip.ncsla.com). Enter your username and password. This will establish a secure connection and validate your identity.

Upload and Submit the Batch File

Go to the Batch Submission page in SLIP. Following the instructions on this page, browse to and select the XML batch file. Submit the file for upload.

SLIP Validates the File

Upon successfully uploading a batch file in SLIP, the system will queue the submission for processing. When the system is ready to process the submission, the validation process will begin.

The first step in the validation process is to validate the format and structure of the XML file as identified in the XML Schema. The next step validates the policy data contained within the XML file itself. If any validation criteria are unsuccessful, the file will be rejected. The XML file format and/or data will have to be corrected and resubmitted.

Whether the file is accepted or rejected, an email will be sent to the user or users associated with Batch account. If the submission was successful, the email will include the filing number and filing date. If this submission was rejected, the email will contain the date and time the file import was attempted and the reason(s) the file was rejected. In both scenarios, the Batch Submission page within SLIP will display the processing status of any submission.

Monitor the Batch Submission Status

After confirming that your batch file was successfully uploaded in SLIP, you may monitor the batch progress in the SLIP Batch Submission page. The page will contain the date the file was submitted and received by NCSLSO. Rejected submissions should be corrected and resubmitted in a timely manner. The following table defines the batch statuses.

External Status	Description
RECEIVED	SLIP has identified and received a batch. The received batch will upload into SLIP automatically and validated.
SUBMISSION REJECTED	The batch has failed validation or was not properly imported into SLIP. NCSLSO has not accepted a batch.
SUBMISSION ACCEPTED	The batch has been successfully imported into SLIP.

Batch File is Imported or Rejected

If the file has been accepted for import, no further action is required. As mentioned in section 4.3.5., you may monitor the batch import process on the Batch Submission page.

If the file has been rejected for import, please review the XML file, correct any errors, and resubmit the batch. If you have questions regarding batch file rejection or resubmission, please contact NCSLSO.

6. API BATCH SUBMISSION

a. Description

The API structure will provide the means for third party applications to submit insurance policy data and documentation, on behalf of an insurer, to NCSLSO. It also provides the means for the Insurer Management System to receive feedback in regard to the acceptance of the submitted data by NCSLSO.

All endpoints will be based upon the hypertext transfer protocol (HTTP) and will use the simple object access protocol (SOAP) to exchange structured data sets, as defined in this document.

b. Prerequisites

Insurers may submit policy data using the API Batch Submission Method if the following requirements are met:

i. A SLIP account is required to submit policy data in batch.

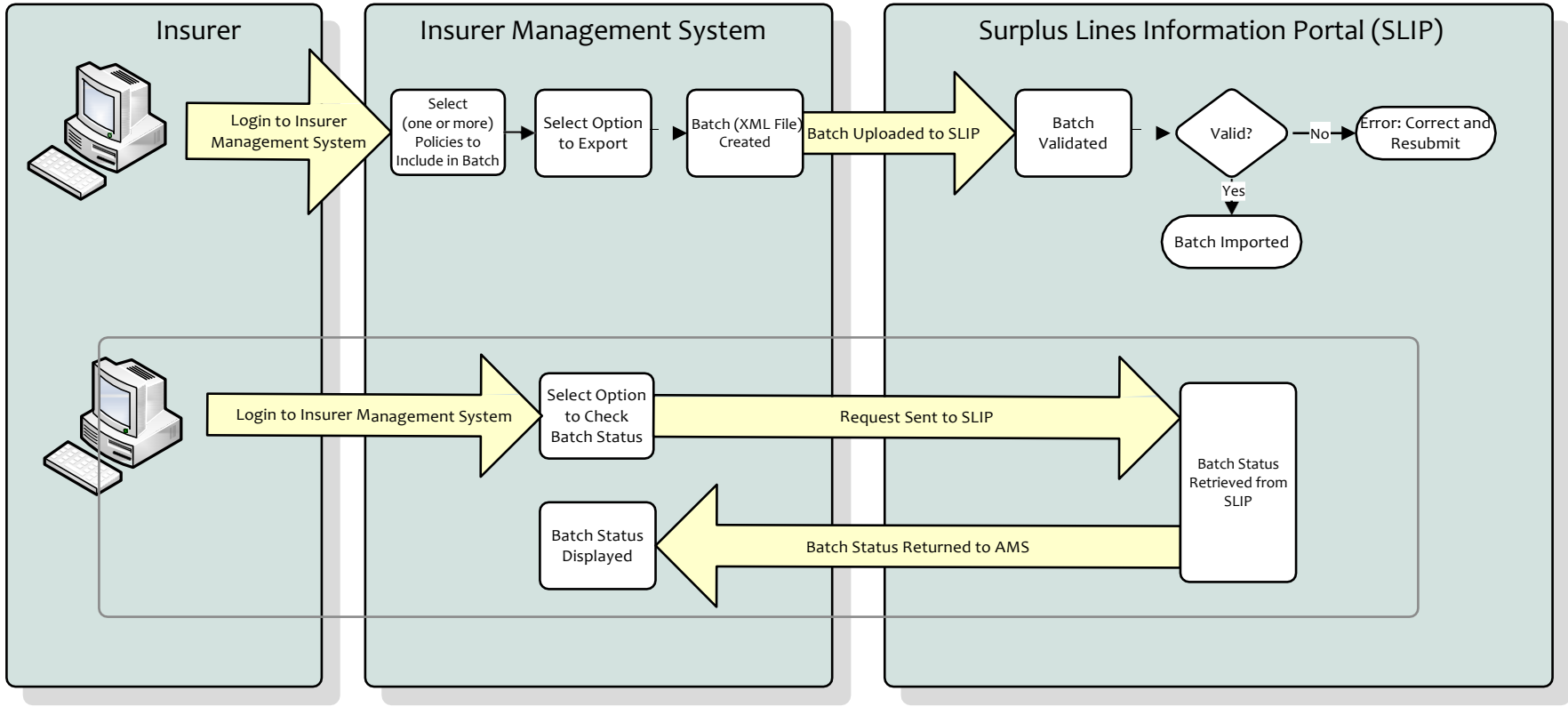
ii. In order for an agency management system to be allowed to interact with the API on behalf of a user, it must first collect a set of credentials from the user. This set of credentials will include the username and API key (user token) value. The user should be able to obtain these values from their SLIP user profile page. The username and key pair are to identify uniquely a user and used to indicate that the user has granted the brokerage management system permission to perform tasks on their behalf. See section VerifyCredentials for the specific method used to verify the credentials.

iii. The SLIP username and password must be supplied within the SOAP body with every request to the API. It is recommended that the insurer management system invoke the credential verification endpoint prior to submitting data to ensure that the credentials are valid.

c. Process

This section identifies the steps required to create and submit policy information using the API batch submission process. The graphic below is a high-level representation of the process flow, and indicates the systems involved.

API Batch Submission



Create Batch File

The first step in the API batch submission process is to create the batch file. Refer to section 3.1.6. of this document for details about creating the batch file. For the API batch submission process, the agency management system will create the batch file automatically.

Submit Batch File

The user will select the option in their agency management system to submit the batch information to SLIP (*see section 5.4.1. Upload Batch File Endpoint for the web service method used*).

SLIP Validates the File

Upon successfully uploading a batch in SLIP, the system will queue the submission for processing. When the system is ready to process the submission, the validation process will begin.

The first step in the validation process is to validate the format and structure of the XML file as identified in the XML Schema. The next step validates the policy data contained within the XML file itself. If any validation criteria are unsuccessful, the file will be rejected. The XML file format and/or data will have to be corrected and resubmitted.

Whether the file is accepted or rejected, an email will be sent to the user or users associated with the Agency License number. If the submission was successful, the email will include the filing number and filing date. If this submission was rejected, the email will contain the date and time the file import was attempted and the reason(s) the file was rejected. In both scenarios, the Batch Submission page within SLIP will display the processing status of any submission.

The user may also use the Check Status Endpoint method to get the status of the batch.

Monitor the Batch Submission Status

After confirming that your batch was successfully uploaded in SLIP, you may monitor the batch progress in the SLIP Batch Submission page or use the Check Status Endpoint method. The page will contain the date the batch was submitted and

received by NCSLSO. Rejected submissions should be corrected and resubmitted in a timely manner. The following table defines the batch statuses.

See section 5.4.2. Check Status Endpoint for the specific method used to monitor the batch status.

External Status	Description
RECEIVED	SLIP has identified and received a batch. The received batch will upload into SLIP automatically and validated.
SUBMISSION REJECTED	The batch has failed validation or was not properly imported into SLIP. NCSLSO has not accepted a batch.
SUBMISSION ACCEPTED	The batch has been successfully imported into SLIP.

d. Methods

This section contains the specific methods used in the API batch submission method[3]. Each method is referenced in a step of the API batch submission process, above.

Credential Verification Endpoint

Upon collecting API credentials from the user, it is recommended that the Insurer Management System invoke this endpoint to verify access to the API on the user's behalf. This will ensure that the user's account is active and verify the user's identity.

It is also recommended that the Insurer Management System verify the user's credentials prior to each data submission to ensure that the credentials remain valid.

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

[3] The API methods are under development and may have changed since the publication of this document. Please contact Infinity Software Development for the latest API methods.

Request message:

```
POST /NCSLABatchFiling.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://NCSLA.com/BatchFiling/VerifyInsurerCredentials"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Header>
  <AuthenticationHeader xmlns="http://NCSLA.com/BatchFiling">
   <UserName>string</UserName>
   <APIKey>string</APIKey>
  </AuthenticationHeader>
 </soap:Header>
 <soap:Body>
  <VerifyInsurerCredentials xmlns="http://NCSLA.com/BatchFiling">
   <strUserName>string</strUserName>
   <strPassword>string</strPassword>
  </VerifyInsurerCredentials>
 </soap:Body>
</soap:Envelope>
```

Request parameters:

PARAMETER	DATA TYPE	DESCRIPTION
strUserName	String	The username for the Insurer Slip Account.
strPassword	String	The password for the Insurer Slip Account.

Response message:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
  <VerifyInsurerCredentialsResponse xmlns="http://NCSLA.com/BatchFiling">
   <VerifyInsurerCredentialsResult>
    <StatusCode>string</StatusCode>
    <StatusMessage>string</StatusMessage>
   </VerifyInsurerCredentialsResult>
  </VerifyInsurerCredentialsResponse>
 </soap:Body>
</soap:Envelope>
```

Response parameters:

Parameter	Data Type	Description
StatusCode	String	Indicates whether the credential has been successfully verified. The value “1” indicates success and “0” means failure.
StatusMessage	String	A message describing the status of the credential verification if any error occurred during processing. “Method call successful.” if the credential has been verified successfully.

Upload Batch File Endpoint

The XML file will be submitted to the Upload Batch Filing method. Upon completion of the batch filing, the API will provide the Insurer Management System with a value that uniquely identifies the batch submission attempt (submission number).

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

Request message:

```
POST /NCSLABatchFiling.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://NCSLA.com/BatchFiling/WebservicesBatchInsurerUpload"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Header>
  <AuthenticationHeader xmlns="http://NCSLA.com/BatchFiling">
   <UserName>string</UserName>
   <APIKey>string</APIKey>
  </AuthenticationHeader>
 </soap:Header>
 <soap:Body>
  <WebservicesBatchInsurerUpload xmlns="http://NCSLA.com/BatchFiling">
   <strSlipUserName>string</strSlipUserName>
   <strSlipPassword>string</strSlipPassword>
   <strComments>string</strComments>
   <FileStream>base64Binary</FileStream>
   <FileName>string</FileName>
  </WebservicesBatchInsurerUpload>
 </soap:Body>
</soap:Envelope>
```

Request parameters:

PARAMETER	DATA TYPE	DESCRIPTION
strSlipUserName	String	The username for the SLIP Account.
strSlipPassword	String	The password for the SLIP Account.
FileName	String	The physical name of the file being submitted including file extension.
FileStream	Binary	The content of the policy submission as a base64Binary format
strComments	String	Comments for the batch filing

Response message:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
  <WebservicesBatchInsurerUploadResponse xmlns="http://NCSLA.com/BatchFiling">
   <WebservicesBatchInsurerUploadResult>
    <StatusCode>string</StatusCode>
    <StatusMessage>string</StatusMessage>
    <SubmissionNumber>string</SubmissionNumber>
   </WebservicesBatchInsurerUploadResult>
  </WebservicesBatchInsurerUploadResponse>
 </soap:Body>
</soap:Envelope>
```

Response parameters:

PARAMETER	DATA TYPE	DESCRIPTION
StatusCode	String	Indicates if the request is success or not. The value “1” indicates success and “0” means failure.
StatusMessage	String	A message describing the status of the request processing if any error occurred during processing. “Method call successful.” if the request has been processed successfully.
SubmissionNumber	String	A value assigned for the batch submission.

Check Status Endpoint

The check status endpoint will allow the Insurer Management System to obtain the status of a batch submission.

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

Request message:

```
POST /NCSLABatchFiling.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://NCSLA.com/BatchFiling/CheckStatus"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Header>
  <AuthenticationHeader xmlns="http://NCSLA.com/BatchFiling">
   <UserName>string</UserName>
   <APIKey>string</APIKey>
  </AuthenticationHeader>
 </soap:Header>
```

```
 <soap:Body>
  <CheckStatus xmlns="http://NCSLA.com/BatchFiling">
   <SubmissionNumber>string</SubmissionNumber>
  </CheckStatus>
 </soap:Body>
</soap:Envelope>
```

Request parameters:

PARAMETER	DATA TYPE	DESCRIPTION
SubmissionNumber	String	The submission number returned as the result of the batch submission.

Response message:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
  <CheckStatusResponse xmlns="http://NCSLA.com/BatchFiling">
   <CheckStatusResult>
    <SubmissionNumber>string</SubmissionNumber>
    <SubmissionStatus>string</SubmissionStatus>
    <StatusCode>string</StatusCode>
    <StatusMessage>string</StatusMessage>
   </CheckStatusResult>
  </CheckStatusResponse>
 </soap:Body>
</soap:Envelope>
```

Response parameters:

Parameter	Data Type	Description
StatusCode	String	Indicates if the request is success or not. The value “1” indicates success and “0” means failure.
StatusMessage	String	A message describing the status of the request processing if any error occurred during processing. “Method call successful.” if the request has been processed successfully.
SubmissionNumber	String	The submission number returned as the result of the batch submission.
SubmissionStatus	String	Status of the submission, either Accepted, Rejected, or Submitted (waiting)

Get File Upload History Endpoint

During submission processing, notifications may be generated that can provide the user with feedback or recommendations for elements within the submitted data.

The following is a sample SOAP 1.1 request and response. The placeholders shown need to be replaced with actual values.

Request message:

```
POST /NCSLABatchFiling.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://NCSLA.com/BatchFiling/GetInsurerFileUploadHistory"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Header>
  <AuthenticationHeader xmlns="http://NCSLA.com/BatchFiling">
   <UserName>string</UserName>
   <APIKey>string</APIKey>
```

```
  </AuthenticationHeader>
 </soap:Header>
 <soap:Body>
  <GetInsurerFileUploadHistory xmlns="http://NCSLA.com/BatchFiling">
   <slipUsername>string</slipUsername>
   <slipPassword>string</slipPassword>
  </GetInsurerFileUploadHistory>
 </soap:Body>
</soap:Envelope>
```

Request parameters:

PARAMETER	DATA TYPE	DESCRIPTION
slipUsername	String	The username for the SLIP Account.
slipPassword	String	The password for the SLIP Account.

Response message:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Body>
  <GetInsurerFileUploadHistoryResponse xmlns="http://NCSLA.com/BatchFiling">
   <GetInsurerFileUploadHistoryResult>
    <UploadHistory>
     <xsd:schema>schema</xsd:schema>xml</UploadHistory>
    <StatusCode>string</StatusCode>
    <StatusMessage>string</StatusMessage>
   </GetInsurerFileUploadHistoryResult>
  </GetInsurerFileUploadHistoryResponse>
 </soap:Body>
```

```
</soap:Envelope>
```

Response parameters:

PARAMETER	DATA TYPE	DESCRIPTION
UploadHistory	XML	Contains history of all batch uploads done by this Batch account (either via SLIP or API), and the status of each submission.
StatusCode	String	Indicates if the request is success or not. The value “1” indicates success and “0” means failure.
StatusMessage	String	A message describing the status of the request processing if any error occurred during processing. “Method call successful.” if the request has been processed successfully.

7. FREQUENTLY ASKED QUESTIONS

The following list identifies frequently asked questions from technical resources concerning the XML Batch Upload:

1. Do I need a SLIP account to submit a Batch file?
   Answer: Yes, a SLIP account is required to submit policy data in batch.

2. Can I use Excel to export a file to Batch?
   Answer: The data contained within a batch submission must be in XML format. XML is a different way of storing data than Excel. XML is the leading standard for data exchange providing several inherent benefits, including data validation, structural enforcement, and platform independence. Please work with your technical staff to prepare your file appropriately.

3. What is the "XML_TransactionId" contained within the transaction element used for in the XML Batch Upload Method?
   Answer: The Transaction ID is a unique non-negative integer value provided by the filer used to uniquely identify a policy transaction submitted using the manual batch file upload.

4. How can I generate a batch file from our data management system?
   Answer: You will need to work with your IT staff to identify the best method to export data from your data management system in the required format.

5. Can I use the manual batch file upload method and also use the manual data entry method?
   Answer: Yes, the system can handle this, but it is recommended you use only one method to avoid the possibility of duplicating filing submissions.

6. Can I edit a policy transaction that has been submitted through the manual batch file upload method?
   Answer: Yes, you can edit all transactions in SLIP, regardless of submission method.

7. How often can I upload a batch?
   Answer: There is no restriction on how often a batch may be uploaded.