



**Florida Surplus Lines Service Office
SLAS Implementation**

Automated Batch Submission
September 14, 2011



1901 Commonwealth Ln | Tallahassee, FL 32309
Phone: 850.383.1011 | Fax: 850.383-1015
www.infinity-software.com



1441 Maclay Commerce Dr. | Ste. 200 | Tallahassee, FL 32312
Phone: 850.562.4496 | Fax: 850.513.9624
www.fslso.com

TABLE OF CONTENTS

1.	Document Metadata	4
1.1	Author(s).....	4
1.2	Intended Audience	4
1.3	Glossary of Terms and Acronyms.....	4
1.4	Document Revision History	6
2.	Executive Summary.....	7
2.1	Contact Information.....	8
3.	Batch Creation Guidelines.....	9
3.1.1	FLSO Batch Filing Guidelines	9
3.1.2	Batch Definition	9
3.1.3	Batch File Size.....	9
3.1.4	Batch File Name.....	9
3.1.5	HTML (XML) Encoding	9
3.1.6	Create a Batch File	10
3.1.7	Batch File Validation	10
3.1.8	XML Schema Table of XML Fields	12
3.1.9	Additional XML Information	13
4.	Manual Batch File Upload	36
4.1	Description	36
4.2	Pre-requisites	36
4.3	Process	36
4.3.1	Create Batch File	37
4.3.2	Log in to SLIP.....	38
4.3.3	Upload and Submit the Batch File.....	38
4.3.4	SLIP Validates the File.....	38
4.3.5	Monitor the Batch Submission Status	38
4.3.6	Batch File is Imported or Rejected.....	39
5.	API Batch Submission.....	40
5.1	Description	40
5.2	Pre-requisites	40
5.3	Process	40
5.3.1	Create Batch File	42
5.3.2	Submit Batch File	42
5.3.3	SLIP Validates the File.....	42
5.3.4	Monitor the Batch Submission Status	42
5.4	Methods	43
5.4.1	Credential Verification Endpoint	43
5.4.1	Upload Batch File Endpoint	45
5.4.2	Check Status Endpoint	46



5.4.3 Get File Upload History Endpoint 48

6. Frequently Asked Questions 50

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 Author(s)

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

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 Florida Surplus Lines Service Office (FSLSO) 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 FSLSO.

1.3 Glossary of Terms and Acronyms

Term or Acronym	Definition or Expansion
<i>Agent</i>	Term may refer to any Agent, including IPC Agents.
<i>AMS</i>	Agency Management System; AMS is a general term for software applications that are used by agencies or agents to manage and maintain policy data. An AMS may be a third-party software application from a vendor, or may be developed internally by the agency/agent. For the purpose of this document, an AMS is any system used by an agency or agent that may be configured to generate policy batch files that can be uploaded to SLIP.
<i>AMS Batch</i>	An XML batch file submitted to SLIP from an AMS via an API.

Term or Acronym	Definition or Expansion
<i>Batch</i>	A group of policies that will be submitted to FLSO. A batch may have any number of policies of any type from any agent.
<i>FLSO</i>	Florida Surplus Lines Service Office
<i>ISD</i>	Infinity Software Development, Inc.
<i>RAPID</i>	Regulatory Administration Platform of Insurance Data; RAPID is an internal platform that allows surplus lines office staff to review submitted policies.
<i>SLAS</i>	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).
<i>SLIP</i>	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.
<i>Web Service</i>	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 .
<i>XML</i>	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.
<i>XML Batch</i>	An XML batch file uploaded by the user via the SLIP interface.

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
Briana G. Hall	5/15/2011	v1.0	First Draft
Briana G. Hall	5/18/2011	v1.1	Internal Revisions
Brent Thompson	9/13/2011	v1.1	Added schema with examples
Brent Thompson	9/14/2011	v1.2	Revisions to schema
Brent Thompson	9/15/2011	v1.3	Revisions to schema
Brent Thompson	9/21/2011	v1.4	Revisions to schema as per suggestions by James Farmer

2. EXECUTIVE SUMMARY

The Surplus Lines Automation Suite (SLAS) is a groundbreaking solution for the regulation of surplus lines and is the #1 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 to submit policy information electronically. SLAS is the solution of choice for regulating over 1/3 of domestic surplus lines premiums.

The Florida Surplus Lines Service Office has been using SLIP for electronic entry of policy data since December 2005. Recently, FLSO introduced a new automated way for agents to enter policy data in batch via SLIP.

SLIP is good news 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 work with SLIP – they can continue to use their existing system.

All agents and IPC Agents have the ability to enter policy data manually into SLIP. However, there are changes you can make to your software to send batches to SLIP automatically. Infinity Software Development, Inc., the developer of SLAS, is reaching out to agents and vendors to talk about how to integrate with 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.

Agents rely on their systems to fully meet their needs. We can help you provide the best value possible. Making the choice to integrate with SLIP reduces workload and provides seamless integration with the Florida Surplus Lines Service Office. SLIP is already in place in Mississippi, Nevada, and Washington, and California. Get ahead of the curve in Florida (and other markets) with the most efficient policy submission experience possible.

Automated batch submission means not having to log out of your agency/agent management system to complete the submission process. Agents that use products that don't integrate with SLIP will have to enter data twice, and track submissions separately. Integration with SLIP brings added value by streamlining your workflow, reducing a pile of duplicate data entry down to just a few clicks.

The rest of this document summarizes the two automated batch submission methods and provides the technical details necessary to implement each method. Upon request, FLSO will provide vendors access to test environments allowing them to test automatic batch submission.

2.1 Contact Information

FSLSO can provide vendors with access to the SLIP testing environment upon request to assist with testing of web services and XML batch functionality. FSLSO will also provide you with the full XML schema.

For more information or assistance, or to request a copy of the XML schema, please contact FSLSO:

Name	James Farmer
Title	Senior Information Architect
Phone	(850)-224-7676 EXT 116
Email	JFarmer@FSLSO.com

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.

3.1.1 FLSO 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.

3.1.2 Batch Definition

The batch file is a single XML file containing policy data in a predefined format. Batches may contain policies of any type, from any agent associated with the agency, as well as any IPC filings.

3.1.3 Batch File Size

XML batch files are limited to 25 MB in size.

3.1.4 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.

3.1.5 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	&	&
Apostrophe	'	'
Quote	"	"

Less Than	<	<
Greater Than	>	>

3.1.6 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 agency. There are several different data management systems in use by agents 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 agency’s data management system. Typically, agents extract data based on a specified date range or some other criteria indicating a submission to FLSO 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 FLSO.

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.

3.1.7 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 agent or agency license number (for surplus lines agencies).
4. Accept and/or reject the batch. An e-mail 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.

3.1.8 Table of XML Fields

The tables below outline the specific values for the elements in an XML schema that is prepared for submitting batch information to FLSO. 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 four schemas are as follows:

3.1.8.1 [Agent Single State Schema](#)

This schema is to only be used when a Florida surplus lines agency is submitting a policy covered by a nonadmitted insurer with insured exposures in Florida only.

3.1.8.2 [Agent Multi State Schema](#)

This schema is to only be used when a Florida surplus lines agency is submitting a policy covered by a nonadmitted insurer with insured exposures in more than one state.

3.1.8.3 [IPC Single State Schema](#)

This schema is to only be used when an IPC Agent is submitting a policy covered by a nonadmitted insurer with insured exposures in Florida only.

3.1.8.4 [IPC Multi State Schema](#)

This schema is to only be used when an IPC Agent is submitting a policy covered by a nonadmitted insurer with insured exposures in more than one state.

Note: The XML structures are under development and may have changed since the publication of this document. Please contact FLSO for the latest XML structure.

3.1.8.5 Agent Single State Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<code><DataSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://ws.flsso.org/schema/slip_batch" SchemaVersion="1.1" ReportingState="FL" SubmissionType="AgentSS"></code>	Root element of the XML file	1	1	-	-
<code><Brokerage></code>	Brokerage Details for the batch	1	1	-	-
<code><LicenseNumber>L123456</LicenseNumber></code>	Brokerage license number	1	1	7	7
<code><Name>XYZ, INC</Name></code>	Brokerage name	1	1	1	80
<code><Contacts></code>	Starting Element				
<code><Contact ContactType="Both"></code>	Contact of type <i>Both</i> , <i>SubmissionContact</i> , or <i>BillingContact</i>	1	2	-	-
<code><FirstName>Jane</FirstName></code>	Contact First name	1	1	1	50
<code><MiddleName>Sarah</MiddleName></code>	Contact Middle name	0	1	0	30
<code><LastName>Smith</LastName></code>	Contact Last name	1	1	1	50
<code><NameSuffix>NameSuffix1</NameSuffix></code>	Contact Name Suffix	0	1	0	30
<code><EmailAddress>EmailAddress@domain.com</EmailAddress></code>	Contact email address	1	1	5	50



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<ContactAddress>	Starting Element				
<Address>Address1</Address>	Contact Street Address	1	1	1	50
<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
<Province>AL</ Province >	Province, if a country other than USA	0	1	1	30
<PostalCode>PostalCode1</PostalCode>	Contact Zip Code	0	1	5	9
<CountryCode>CountryCode1</CountryCode>	Contact Country Code	1	1	1	30
</ContactAddress>	Ending Element				
<PhoneNumber>	Starting Element				
<CountryCode>+1</CountryCode>	Only required if outside USA	0	1	0	5
<AreaCode>888</AreaCode>	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>1234567891</ Fax >	Fax Number	1	1	10	10
</Contact>	Ending Element				
</Contacts>	Ending Element				
</Brokerage>	Ending Element				



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<BrokerPolicies>	Starting Element	1	1	-	-
<BrokerPolicy>	List of policies for a given broker for this group of filings	1	unbound	-	-
<Broker>	Broker	1	1	-	-
<LicenseNumber>A123456</LicenseNumber>	Broker License Number	1	1	7	7
<BrokerInfo>	Broker Details. These are only necessary if we wish to make UPDATES to the Broker's information. This is optional otherwise.	0	1	-	-
<FirstName>John</FirstName>	Broker First name	1	1	1	50
<MiddleName>M</MiddleName>	Broker Middle name	0	1	0	30
<LastName>Smith</LastName>	Broker Last name	1	1	1	50
<NameSuffix>Jr</NameSuffix>	Broker Name Suffix	0	1	0	30
<EmailAddress>EmailAddress1</EmailAddress>	Broker email address	1	1	5	50
<BrokerAddress>	Starting Element	1	1	-	-
<Address>123 Test Street</Address>	Broker Street Address	1	1	1	50
<Address2>Suite 123</Address2>	Broker Street Address 2	1	1	0	50
<City>Tallahassee</City>	Broker City	1	1	1	50
<StateCode>FL</StateCode>	Broker 2 letter state abbreviation	0	1	2	2
<PostalCode>32309</PostalCode>	Broker Zip Code	0	1	5	9
<CountryCode>USA</CountryCode>	Broker Country Code	1	1	1	30
</BrokerAddress>	Ending Element	-	-	-	-
<MailingAddress>	Starting Element	1	1	-	-



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Address>100 Main Road</Address>	Mailing Street Address	1	1	1	50
<Address2>Suite 200</Address2>	Mailing Street Address 2	1	1	0	30
<City>Tallahassee</City>	Mailing City	1	1	1	50
<StateCode>FL</StateCode>	Mailing 2 letter state abbreviation	0	1	2	2
<PostalCode>32301</PostalCode>	Mailing Zip Code	0	1	5	9
<CountryCode>USA</CountryCode>	Mailing Country Code	1	1	1	30
</MailingAddress>	Ending Element	-	-	-	-
<PhoneNumber>	Starting Element				
<CountryCode>1</CountryCode>	Only required if outside USA	0	1	0	5
<AreaCode>888</AreaCode>	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>123456789</Fax>	Fax Number of Broker	0	1	10	10
</BrokerInfo>	Ending Element				
</Broker>	Ending Element				
<Policies>	Starting Element				
<Policy Xml_PolicyID="0">	Starting Element (The Attribute "XML_PolicyId" must be unique within the submission)	1	Unbound	-	-



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<PolicyNumber>ABC1234-1</PolicyNumber>	Policy number	1	1	1	50
<ExpirationDate> 2015-01-31 </ExpirationDate>	Policy expiration date; (YYYY-MM-DD)	1	1	-	-
<InsuredName>John Doe</InsuredName>	Name of insured	1	1	1	75
<County>Leon</County>	County of risk	1	1	1	30
<PostalCode>32304</PostalCode>	Postal zip code of risk	1	1	5	9
<Comment>Comment4</Comment>	Any comments for the policy	0	1	0	250
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element (The Attribute "XML_TransactionID" must be unique within the submission)	1	Unbound	-	-
<CoverageCode> 4001 </CoverageCode>	Coverage code	1	1	4	4
<TaxStatus>0</TaxStatus>	Tax Status	1	1	1	1
<TransactionType>1</TransactionType>	Transaction Type	1	1	1	2
<EffectiveDate>2011-01-31 </EffectiveDate>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<HurricaneDeductible>100.00</HurricaneDeductible>	Required if Windstorm Coverage is provided (Must be converted to a dollar amount).	0	1	1	10
<WsCoverage>N</WsCoverage>	Windstorm Coverage provided? (Y/N)	0	1	1	1
<WspEligible>N</WspEligible>	Eligible for Windstorm Pool? (Y/N)	0	1	1	1
<AopDeductible>123.21</AopDeductible>	All Other Perils Deductible (deductible for damage from any peril named in the policy, other than hurricane).	0	1	1	10
<PrimaryAmount>4321.21</PrimaryAmount>	Use coverage A (or Coverage C if Coverage Codes 2003 or 2005 are being used).	0	1	1	10
<Insurer Xml_InsurerID="0">	Starting Element				



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
	(The Attribute "XML_InsurerID" must be unique within the submission)				
<Name>First Insurance</Name>	Name of Insurer	0	1	1	75
<NAICNumber> A-10349</NAICNumber>	NAIC Number of Insurer	1	1	1	10
</Insurer>	Ending Element				
<IssueDate>2011-08-01</IssueDate>	The date an insurance company issues a policy or endorsement. This date may be different from the date the insurance becomes effective.	0	1	-	-
<Premium>1234.56</Premium>	The amount charged/returned to the insured minus any fees. In the case of a multi-state risk	1	1	1	10
<PolicyFee>198.43</PolicyFee>	Any fees charged/returned to the insured.	1	1	1	10
<LateExempt>Y</LateExempt>	This field is used to declare a late exemption for a transaction being submitted greater than the required deadline. (Y/N)	0	1	1	1
<LateReason>B</LateReason>	View an FAQ for descriptions of these values	0	1	1	1
<UMR>B1234ASD</UMR>	The policy's Unique Market Reference number if it is a Lloyd's	0	1	6	17
<Comment>Comment10</Comment>	Any comments for this transaction.	0	1	0	250
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</BrokerPolicy>	Ending Element				
</BrokerPolicies>	Ending Element				
</DataSet>	Ending Element				

3.1.8.6 Agent Multi State Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<DataSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://ws.fslso.org/schema/slip_batch" SchemaVersion="1.1" ReportingState="FL" SubmissionType="AgentMS">	Root element of the XML file	1	1	-	-
<Brokerage>	Brokerage Details for the batch	1	1	-	-
<LicenseNumber>L123456</LicenseNumber>	Brokerage license number	1	1	7	7
<Name>XYZ, INC</Name>	Brokerage name	1	1	1	80
<Contacts>	Starting Element				
<Contact ContactType="Both">	Contact of type <i>Both</i> , <i>SubmissionContact</i> , or <i>BillingContact</i>	1	2	-	-
<FirstName>Jane</FirstName>	Contact First name	1	1	1	50
<MiddleName>Sarah</MiddleName>	Contact Middle name	0	1	0	30
<LastName>Smith</LastName>	Contact Last name	1	1	1	50
<NameSuffix>NameSuffix1</NameSuffix>	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	50
<Address2>Address21</Address2>	Contact Street Address 2	1	1	0	30
<City>City1</City>	Contact City	1	1	1	50
<StateCode>AL</StateCode>	2 letter state abbreviation	0	1	2	2



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Province>AL</ Province >	Province, if a country other than USA	0	1	1	30
<PostalCode>PostalCode1</PostalCode>	Contact Zip Code	0	1	5	9
<CountryCode>CountryCode1</CountryCode>	Contact Country Code	1	1	1	30
</ContactAddress>	Ending Element				
<PhoneNumber>	Starting Element				
<CountryCode>1</CountryCode>	Only required if outside USA	0	1	0	5
<AreaCode>888</AreaCode>	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
<Fax>1234567890</ Fax >	Fax Number	1	1	10	10
</PhoneNumber>	Ending Element				
</Contact>	Ending Element				
</Contacts>	Ending Element				
</Brokerage>	Ending Element				
<BrokerPolicies>	Starting Element				
<BrokerPolicy>	List of policies for a given broker for this group of filings	1	unbound	-	-
<Broker>	Broker	1	1	-	-
<LicenseNumber>A123456</LicenseNumber>	Broker License Number	1	1	7	7
<BrokerInfo>	Broker Details. These are only necessary if we wish to make UPDATES to the Broker's information. This is optional	0	1	-	-



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
	otherwise.				
<FirstName>John</FirstName>	Broker First name	1	1	1	50
<MiddleName>M</MiddleName>	Broker Middle name	0	1	0	30
<LastName>Smith</LastName>	Broker Last name	1	1	1	50
<NameSuffix>Jr</NameSuffix>	Broker Name Suffix	0	1	0	30
<EmailAddress>EmailAddress1</EmailAddress>	Broker email address	1	1	5	50
<BrokerAddress>	Starting Element	1	1	-	-
<Address>123 Test Street</Address>	Broker Street Address	1	1	1	50
<Address2>Suite 123</Address2>	Broker Street Address 2	1	1	0	50
<City>Tallahassee</City>	Broker City	1	1	1	50
<StateCode>FL</StateCode>	Broker 2 letter state abbreviation	0	1	2	2
<PostalCode>32309</PostalCode>	Broker Zip Code	0	1	5	9
<CountryCode>USA</CountryCode>	Broker Country Code	1	1	1	30
</BrokerAddress>	Ending Element	-	-	-	-
<MailingAddress>	Starting Element	1	1	-	-
<Address>100 Main Road</Address>	Mailing Street Address	1	1	1	50
<Address2>Suite 200</Address2>	Mailing Street Address 2	1	1	0	50
<City>Tallahassee</City>	Mailing City	1	1	1	50
<StateCode>FL</StateCode>	Mailing 2 letter state abbreviation	0	1	2	2
<PostalCode>32301</PostalCode>	Mailing Zip Code	0	1	5	9
<CountryCode>USA</CountryCode>	Mailing Country Code	1	1	1	30



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
</MailingAddress>	Ending Element	-	-	-	-
<PhoneNumber>	Starting Element				
<CountryCode>1</CountryCode>	Only required if outside USA	0	1	0	5
<AreaCode>888</AreaCode>	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>1234567890</Fax>	Fax Number of Broker	0	1	10	10
</BrokerInfo>	Ending Element				
</Broker>	Ending Element				
<Policies>	Starting Element				
<Policy Xml_PolicyID="0">	Starting Element (The Attribute "XML_PolicyId" must be unique within the submission)	1	Unbound	-	-
<PolicyNumber>ABC1234-1</PolicyNumber>	Policy number	1	1	1	50
<ExpirationDate> 2015-01-31 </ExpirationDate>	Policy expiration date; (YYYY-MM-DD)	1	1	-	-
<InsuredName>John Doe</InsuredName>	Name of insured	1	1	1	75
<County>Leon</County>	County of risk	1	1	1	30
<PostalCode>32304</PostalCode>	Postal zip code of risk	1	1	5	9
<HomeState>FL</HomeState>	Homestate of risk of the policy	1	1	2	2



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Comment>Comment4</Comment>	Any comments for the policy	0	1	0	250
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element (The Attribute "XML_TransactionID" must be unique within the submission)	1	Unbound	-	-
<CoverageCode> 4001 </CoverageCode>	Coverage code	1	1	4	4
<TaxStatus>0</TaxStatus>	Tax Status	1	1	1	1
<TransactionType>1</TransactionType>	Transaction Type	1	1	1	2
<EffectiveDate>2011-01-31 </EffectiveDate>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<HurricaneDeductible> 100.00</HurricaneDeductible>	Required if Windstorm Coverage is provided (Must be converted to a dollar amount).	0	1	1	10
<WsCoverage>N</WsCoverage>	Windstorm Coverage provided? (Y/N)	0	1	1	1
<WspEligible>N</WspEligible>	Eligible for Windstorm Pool? (Y/N)	0	1	1	1
<AopDeductible> 123.21 </AopDeductible>	All Other Perils Deductible (deductible for damage from any peril named in the policy, other than hurricane).	0	1	1	10
<PrimaryAmount> 4321.21 </PrimaryAmount>	Use coverage A (or Coverage C if Coverage Codes 2003 or 2005 are being used).	0	1	1	10
<Insurer Xml_InsurerID="0">	Starting Element (The Attribute "XML_InsurerID" must be unique within the submission)				
<Name>First Insurance</Name>	Name of Insurer	0	1	1	75
<NAICNumber> 10349</NAICNumber>	NAIC Number of Insurer	1	1	1	10
</Insurer>	Ending Element				
<IssueDate>2011-08-01</IssueDate>	The date an insurance company issues a policy or endorsement. This date may be different from the date the insurance becomes	0	1	-	-



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
	effective.				
<Premium>2000.00</Premium>	The amount charged/returned to the insured minus any fees. In the case of a multi-state risk	1	1	1	10
<PolicyFee>35.00</PolicyFee>	Any fees charged/returned to the insured.	1	1	1	10
<LateExempt>Y</LateExempt>	This field is used to declare a late exemption for a transaction being submitted greater than the required deadline. (Y/N)	0	1	1	1
<LateReason>B</LateReason>	View an FAQ for descriptions of these values	0	1	1	1
<UMR>B1234ASD</UMR>	The policy's Unique Market Reference number if it is a Lloyd's	0	1	6	17
<Comment>Comment10</Comment>	Any comments for this transaction.	0	1	0	250
<StateAllocations>	Starting Element	1	Unbound	-	-
<Allocation>	Starting Element				
<StateAllocation>FL</StateAllocation>	State of this allocation	1	1	2	2
<PremiumAllocation>2000.00</PremiumAllocation>	Premium of this allocation	1	1	1	10
<PolicyFeeAllocation>35.00</PolicyFeeAllocation>	Policy Fee of this allocation	1	1	1	10
</Allocation>	Ending Element				
</StateAllocations>	Ending Element				
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</BrokerPolicy>	Ending Element				
</BrokerPolicies>	Ending Element				

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
</DataSet>	Ending Element				

3.1.8.7 IPC Single State Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<DataSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://ws.fslso.org/schema/slip_batch" SchemaVersion="1.1" ReportingState="FL" SubmissionType="IPCSS">	Root element of the XML file	1	1	-	-
<LicenseNumber>L123456</LicenseNumber>	IPC Agent license number	1	1	7	7
<Contacts>	Starting Element				
<Contact ContactType="Both">	Contact of type <i>Both</i> , <i>SubmissionContact</i> , or <i>BillingContact</i>	1	2	-	-
<FirstName>Jane</FirstName>	Contact First name	1	1	1	50
<MiddleName>Sarah</MiddleName>	Contact Middle name	0	1	0	30
<LastName>Smith</LastName>	Contact Last name	1	1	1	50
<NameSuffix>III</NameSuffix>	Contact Name Suffix	0	1	0	30
<EmailAddress>EmailAddress@domain.com</EmailAddress>	Contact email address	1	1	5	50
<ContactAddress>	Starting Element				



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Address>123 Test St</Address>	Contact Street Address	1	1	1	50
<Address2>Suite 200</Address2>	Contact Street Address 2	1	1	0	50
<City>Tallahassee</City>	Contact City	1	1	1	50
<StateCode>FL</StateCode>	2 letter state abbreviation	0	1	2	2
<PostalCode>32309</PostalCode>	Contact Zip Code	0	1	5	9
<CountryCode>USA</CountryCode>	Contact Country Code	1	1	1	30
</ContactAddress>	Ending Element				
<PhoneNumber>	Starting Element				
<CountryCode>+1</CountryCode>	Only required if outside USA	0	1	0	5
<AreaCode>888</AreaCode>	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>1234567891</ Fax >	Fax Number	1	1	10	10
</Contact>	Ending Element				
</Contacts>	Ending Element				
<Policies>	Starting Element				
<Policy Xml_PolicyID="0">	Starting Element (The Attribute "XML_PolicyId" must be unique within the submission)	1	Unbound	-	-



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<PolicyNumber>ABC1234-1</PolicyNumber>	Policy number	1	1	1	50
<ExpirationDate> 2015-01-31 </ExpirationDate>	Policy expiration date; (YYYY-MM-DD)	1	1	-	-
<Insured>	Starting Element				
<InsuredName>Jane Doe</InsuredName>	Name of insured	1	1	1	75
<ContactPerson>John Doe</ContactPerson>	Name of Contact Person	1	1	1	75
<EmailAddress>EmailAddress@domain.com</EmailAddress>	Email address of insured.	1	1	5	
<InsuredAddress>	Starting Element				
<Address> 123 Test ST</Address>	Contact Street Address	1	1	0	50
<Address2>Suite 200</Address2>	Contact Street Address 2	1	1	0	50
<City>Tallahassee</City>	Contact City	1	1	0	50
<StateCode>FL</StateCode>	2 letter state abbreviation	0	1	2	2
<PostalCode>32309</PostalCode>	Contact Zip Code	0	1	0	50
<CountryCode>USA</CountryCode>	Contact Country Code	1	1	0	30
</InsuredAddress>	Ending Element				
<PhoneNumber>	Starting Element				
<AreaCode>888</AreaCode>	Area Code	1	1	0	5
<Prefix>123</Prefix>	Phone prefix number	1	1	0	5
<Line>1234</Line>	Phone line number	1	1	0	5
</PhoneNumber>	Ending Element				
</Insured>	Ending Element				



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<County>Leon</County>	County of risk	1	1	0	30
<PostalCode>PostalCode4</PostalCode>	Postal zip code of risk	1	1	5	9
<Comment>Comment4</Comment>	Any comments for the policy	0	1	0	250
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element (The Attribute "XML_TransactionID" must be unique within the submission)	1	Unbound	-	-
<CoverageCode> 4001 </CoverageCode>	Coverage code	1	1	4	4
<TaxStatus>0</TaxStatus>	Tax Status	1	1	1	1
<TransactionType>1</TransactionType>	Transaction Type	1	1	1	2
<EffectiveDate>2011-01-31 </EffectiveDate>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<HurricaneDeductible>100.00</HurricaneDeductible>	Required if Windstorm Coverage is provided (Must be converted to a dollar amount).	0	1	0	10
<WsCoverage>N</WsCoverage>	Windstorm Coverage provided? (Y/N)	0	1	1	1
<WspEligible>N</WspEligible>	Eligible for Windstorm Pool? (Y/N)	0	1	1	1
<AopDeductible>123.21</AopDeductible>	All Other Perils Deductible (deductible for damage from any peril named in the policy, other than hurricane).	0	1	0	10
<PrimaryAmount>4321.21</PrimaryAmount>	Use coverage A (or Coverage C if Coverage Codes 2003 or 2005 are being used).	0	1	0	10
<Insurer Xml_InsurerID="0">	Starting Element (The Attribute "XML_InsurerID" must be unique within the submission)				
<Name>First Insurance</Name>	Name of Insurer	0	1	0	75
<NAICNumber> 10349</NAICNumber>	NAIC Number of Insurer	1	1	0	10

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
</Insurer>	Ending Element				
<IssueDate>2011-08-01</IssueDate>	The date an insurance company issues a policy or endorsement. This date may be different from the date the insurance becomes effective.	0	1	-	-
<Premium>1234.56</Premium>	The amount charged/returned to the insured minus any fees. In the case of a multi-state risk	1	1	0	10
<LateExempt>Y</LateExempt>	This field is used to declare a late exemption for a transaction being submitted greater than the required deadline. (Y/N)	0	1	1	1
<LateReason>B</LateReason>	View an FAQ for descriptions of these values	0	1	1	1
<UMR>B1234ASD</UMR>	The policy's Unique Market Reference number if it is a Lloyd's	0	1	1	12
<Comment>Comment10</Comment>	Any comments for this transaction.	0	1	0	250
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</BrokerPolicy>	Ending Element				
</BrokerPolicies>	Ending Element				
</DataSet>	Ending Element				

3.1.8.8 IPC Multi State Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<DataSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://ws.fslso.org/schema/slip_batch" SchemaVersion="1.1" ReportingState="FL" SubmissionType="IPCSS">	Root element of the XML file	1	1	-	-
<LicenseNumber>L123456</LicenseNumber>	IPC Agent license number	1	1	7	7
<Contacts>	Starting Element				
<Contact ContactType="Both">	Contact of type <i>Both</i> , <i>SubmissionContact</i> , or <i>BillingContact</i>	1	2	-	-
<FirstName>Jane</FirstName>	Contact First name	1	1	1	50
<MiddleName>Sarah</MiddleName>	Contact Middle name	0	1	0	30
<LastName>Smith</LastName>	Contact Last name	1	1	1	50
<NameSuffix>III</NameSuffix>	Contact Name Suffix	0	1	0	30
<EmailAddress>EmailAddress@domain.com</EmailAddress>	Contact email address	1	1	5	50
<ContactAddress>	Starting Element				
<Address>123 Test St</Address>	Contact Street Address	1	1	0	50
<Address2>Suite 200</Address2>	Contact Street Address 2	1	1	0	50
<City>Tallahassee</City>	Contact City	1	1	0	50
<StateCode>FL</StateCode>	2 letter state abbreviation	0	1	2	2
<PostalCode>32309</PostalCode>	Contact Zip Code	0	1	0	50
<CountryCode>USA</CountryCode>	Contact Country Code	1	1	0	30
</ContactAddress>	Ending Element				



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<PhoneNumber>	Starting Element				
<CountryCode>+1</CountryCode>	Only required if outside USA	0	1	0	5
<AreaCode>888</AreaCode>	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>1234567891</ Fax >	Fax Number	1	1	10	10
</Contact>	Ending Element				
</Contacts>	Ending Element				
<Policies>	Starting Element				
<Policy Xml_PolicyID="0">	Starting Element (The Attribute "XML_PolicyId" must be unique within the submission)	1	Unbound	-	-
<PolicyNumber>ABC1234-1</PolicyNumber >	Policy number	1	1	1	50
<ExpirationDate> 2015-01-31 </ExpirationDate>	Policy expiration date; (YYYY-MM-DD)	1	1	-	-
<Insured>	Starting Element				
<InsuredName>Jane Doe</InsuredName>	Name of insured	1	1	1	75
<ContactPerson>John Doe</ContactPerson>	Name of Contact Person	1	1	1	75
<EmailAddress>EmailAddress@domain.com</EmailAddress>	Email address of insured.	1	1	5	50
<InsuredAddress>	Starting Element				



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Address>123 Test ST</Address>	Contact Street Address	1	1	1	50
<Address2>Suite 200</Address2>	Contact Street Address 2	1	1	0	50
<City>Tallahassee</City>	Contact City	1	1	1	50
<StateCode>FL</StateCode>	2 letter state abbreviation	0	1	2	2
<PostalCode>32309</PostalCode>	Contact Zip Code	0	1	0	50
<CountryCode>USA</CountryCode>	Contact Country Code	1	1	0	30
</InsuredAddress>	Ending Element				
<PhoneNumber>	Starting Element				
<AreaCode>888</AreaCode>	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
</PhoneNumber>	Ending Element				
</Insured>	Ending Element				
<County>Leon</County>	County of risk	1	1	1	30
<PostalCode>PostalCode4</PostalCode>	Postal zip code of risk	1	1	5	9
<Comment>Comment4</Comment>	Any comments for the policy	0	1	0	250
<HomeState>FL</HomeState>	Homestate of risk of the policy	1	1	2	2
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element (The Attribute "XML_TransactionID" must be unique within the submission)	1	Unbound	-	-



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<CoverageCode> 4001 </CoverageCode>	Coverage code	1	1	4	4
<TaxStatus>0</TaxStatus>	Tax Status	1	1	1	1
<TransactionType>1</TransactionType>	Transaction Type	1	1	1	2
<EffectiveDate>2011-01-31 </EffectiveDate>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<HurricaneDeductible> 100.00</HurricaneDeductible>	Required if Windstorm Coverage is provided (Must be converted to a dollar amount).	0	1	1	10
<WsCoverage>N</WsCoverage>	Windstorm Coverage provided? (Y/N)	0	1	1	1
<WspEligible>N</WspEligible>	Eligible for Windstorm Pool? (Y/N)	0	1	1	1
<AopDeductible> 123.21 </AopDeductible>	All Other Perils Deductible (deductible for damage from any peril named in the policy, other than hurricane).	0	1	1	10
<PrimaryAmount> 4321.21 </PrimaryAmount>	Use coverage A (or Coverage C if Coverage Codes 2003 or 2005 are being used).	0	1	1	10
<Insurer Xml_InsurerID="0">	Starting Element (The Attribute "XML_InsurerID" must be unique within the submission)				
<Name>First Insurance</Name>	Name of Insurer	0	1	1	75
<NAICNumber> 10349</NAICNumber>	NAIC Number of Insurer	1	1	1	10
</Insurer>	Ending Element				
<IssueDate>2011-08-01</IssueDate>	The date an insurance company issues a policy or endorsement. This date may be different from the date the insurance becomes effective.	0	1	-	-
<Premium>2000.00</Premium>	The amount charged/returned to the insured minus any fees. In the case of a multi-state risk	1	1	1	10
<LateExempt>Y</LateExempt>	This field is used to declare a late exemption for a transaction being submitted greater than the required deadline. (Y/N)	0	1	1	1



SURPLUS LINES
AUTOMATION SUITE

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<LateReason>B</LateReason>	View an FAQ for descriptions of these values	0	1	1	1
<UMR>B1234ASD</UMR>	The policy's Unique Market Reference number if it is a Lloyd's	0	1	1	12
<Comment>Comment10</Comment>	Any comments for this transaction.	0	1	0	250
<StateAllocations>	Starting Element				
<Allocation>	Starting Element				
<StateAllocation>FL</StateAllocation>	State of this allocation	1	1	2	2
<PremiumAllocation>2000.00</PremiumAllocation>	Premium of this allocation	1	1	1	10
</Allocation>	Ending Element				
</StateAllocations>	Ending Element				
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</BrokerPolicy>	Ending Element				
</BrokerPolicies>	Ending Element				
</DataSet>	Ending Element				

3.1.9 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>

4. MANUAL BATCH FILE UPLOAD

4.1 Description

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

Agents 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:

1. The agent will generate an XML file containing the policy data they wish to submit to FLSO. 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.
2. The agent will log in to SLIP to upload the XML.

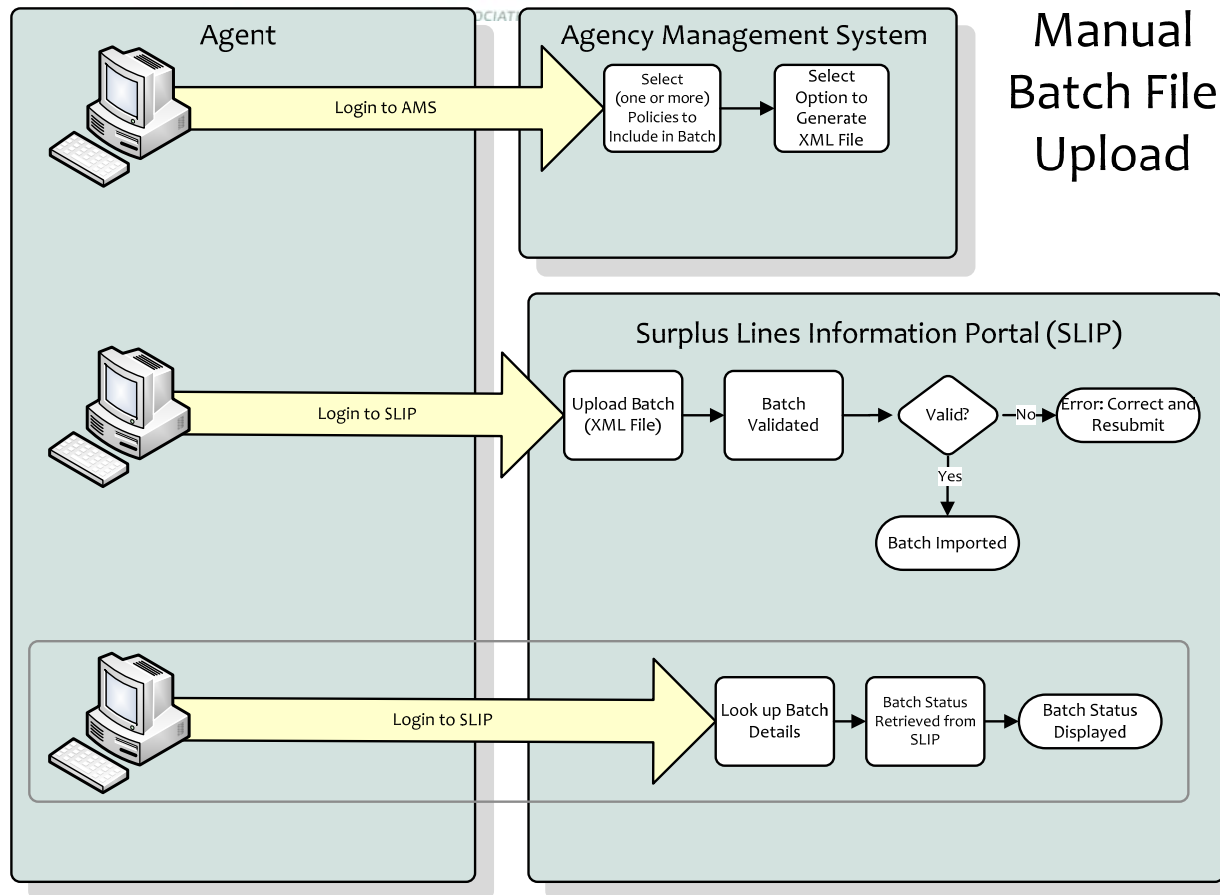
4.2 Pre-requisites

Agents may submit policy data using the manual batch file upload method if the following requirements are met:

1. A SLIP Batch Filing account is required to submit policy data in batch. This account is distinct and separate from your general SLIP account. Contact James Farmer at 1.800.562.4496 (ext. 116) at FLSO to receive a SLIP Batch Filing account.
2. SLIP is designed to work for Microsoft Internet Explorer 7+.

4.3 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.



4.3.1 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 AMS 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.

4.3.2 Log in to SLIP

Using a supported web browser, go to the FLSO SLIP website (log onto FLSO to find the correct login page based on your user type: <http://www.fslso.com/software/the.slips.aspx>). Enter your username and password. This will establish a secure connection and validate your identity.

FLSO initially creates the agent as an administrator on the account, who in turn has the ability to create other users for that agent.

4.3.3 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.

4.3.4 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 e-mail 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.

4.3.5 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 SLA California. 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. FLSO has not accepted a batch.
SUBMISSION ACCEPTED	The batch has been successfully imported into SLIP.

4.3.6 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 FLSO.

5. API BATCH SUBMISSION

5.1 Description

The API structure will provide the means for third party applications to submit insurance policy data and documentation, on behalf of an agent or agency, to FLSO. It also provides the means for the AMS to receive feedback in regards to the acceptance of the submitted data by FLSO.

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.

5.2 Pre-requisites

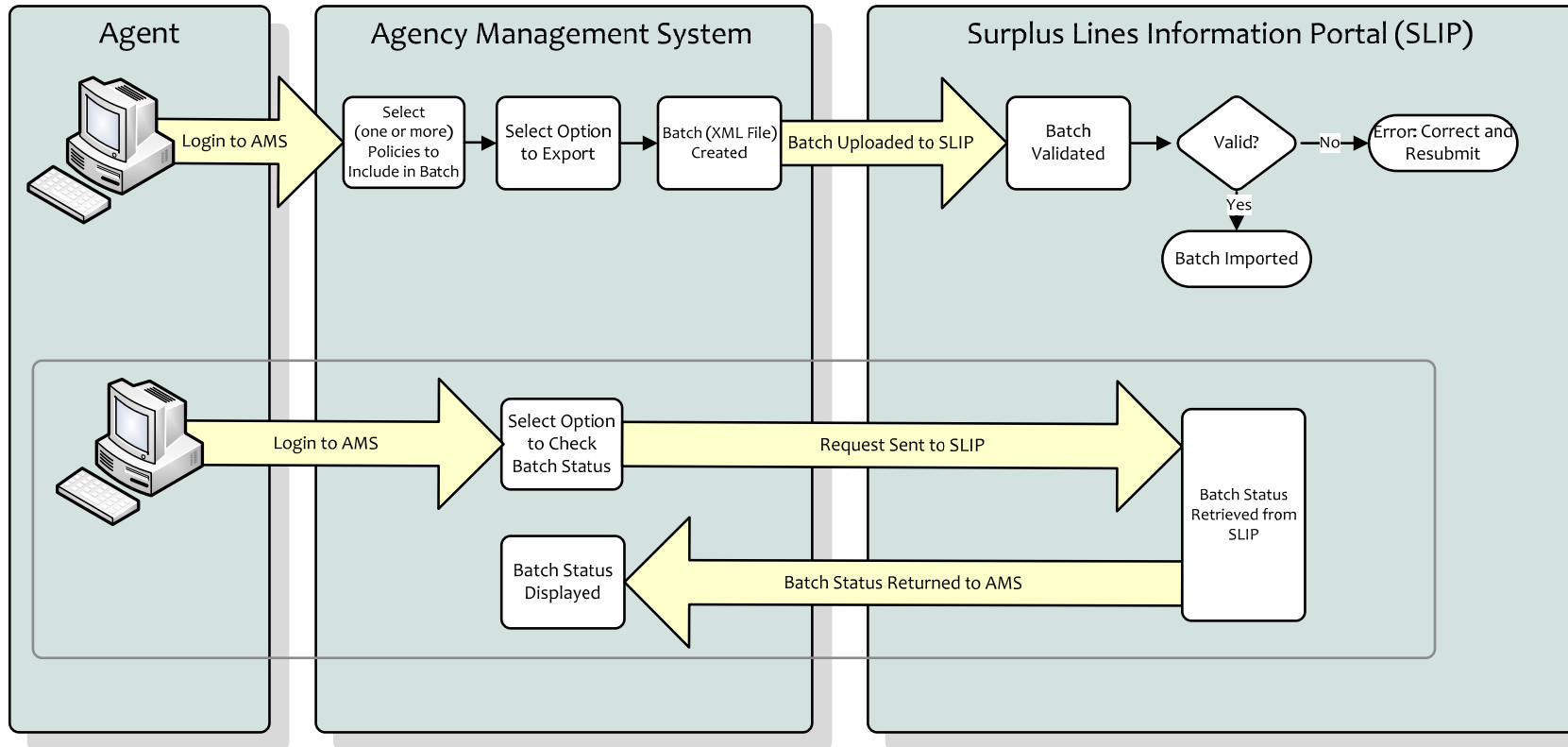
Agents may submit policy data using the API Batch Submission Method if the following requirements are met:

1. A SLIP Batch Filing account is required to submit policy data in batch. This account is distinct and separate from your general SLIP account. Contact James Farmer at 1.800.562.4496 (ext. 116) at FLSO to receive a SLIP Batch Filing account.
2. 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.
3. The Batch username and password must be supplied within the SOAP body with every request to the API. It is recommended that the brokerage management system invoke the credential verification endpoint prior to submitting data to ensure that the credentials are valid.

5.3 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



5.3.1 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.

5.3.2 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*).

5.3.3 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 e-mail 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.

5.3.4 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 FLSO. 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. FLSO has not accepted a batch.
SUBMISSION ACCEPTED	The batch has been successfully imported into SLIP.

5.4 Methods

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

***Note:** 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.*

5.4.1 Credential Verification Endpoint

Upon collecting API credentials from the user, it is recommended that the AMS 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 AMS 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.

Request message:

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

<?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://fslso.com/BatchFiling">
```

```

<UserName>string</UserName>
<APIKey>string</APIKey>
</AuthenticationHeader>
</soap:Header>
<soap:Body>
  <VerifyCredentials xmlns="http://fslso.com/BatchFiling">
    <strWSUserName>string</strWSUserName>
    <strWSPassword>string</strWSPassword>
  </VerifyCredentials>
</soap:Body>
</soap:Envelope>

```

Request parameters:

PARAMETER	DATA TYPE	DESCRIPTION
strWSUserName	String	The username for the Batch Account.
strWSPassword	String	The password for the Batch 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>
    <VerifyCredentialsResponse xmlns="http://fslso.com/BatchFiling">
      <VerifyCredentialsResult>
        <StatusCode>string</StatusCode>
        <StatusMessage>string</StatusMessage>
      </VerifyCredentialsResult>
    </VerifyCredentialsResponse>
  </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.

5.4.1 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 AMS 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 /FSLSOBatchFiling.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://fslso.com/BatchFiling/WebservicesBatchUpload"

<?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://fslso.com/BatchFiling">
      <UserName>string</UserName>
      <APIKey>string</APIKey>
    </AuthenticationHeader>
  </soap:Header>
  <soap:Body>
    <WebservicesBatchUpload xmlns="http://fslso.com/BatchFiling">
      <strWSUserName>string</strWSUserName>
      <strWSPassword>string</strWSPassword>
      <strComments>string</strComments>
      <FileStream>base64Binary</FileStream>
      <FileName>string</FileName>
    </WebservicesBatchUpload>
  </soap:Body>
</soap:Envelope>
```

Request parameters:

PARAMETER	DATA TYPE	DESCRIPTION
strWSUserName	String	The username for the Batch Account.
strWSPassword	String	The password for the Batch 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>
    <WebservicesBatchUploadResponse xmlns="http://fslso.com/BatchFiling">
      <WebservicesBatchUploadResult>
        <StatusCode>string</StatusCode>
        <StatusMessage>string</StatusMessage>
        <ConfirmationNumber>string</ConfirmationNumber>
      </WebservicesBatchUploadResult>
    </WebservicesBatchUploadResponse>
  </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.
ConfirmationNumber	String	A value assigned for the batch submission.

5.4.2 Check Status Endpoint

The check status endpoint will allow the AMS 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 /FSLSOBatchFiling.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
```

Content-Length: [length](#)
 SOAPAction: "http://fslso.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://fslso.com/BatchFiling">
      <UserName>string</UserName>
      <APIKey>string</APIKey>
    </AuthenticationHeader>
  </soap:Header>
  <soap:Body>
    <CheckStatus xmlns="http://fslso.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://fslso.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)

5.4.3 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 /FSLSOBatchFiling.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://fslso.com/BatchFiling/GetFileUploadHistory"

<?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://fslso.com/BatchFiling">
      <UserName>string</UserName>
      <APIKey>string</APIKey>
    </AuthenticationHeader>
  </soap:Header>
  <soap:Body>
    <GetFileUploadHistory xmlns="http://fslso.com/BatchFiling">
      <strWSUserName>string</strWSUserName>
      <strWSPassword>string</strWSPassword>
    </GetFileUploadHistory>
  </soap:Body>
</soap:Envelope>
```

Request parameters:

PARAMETER	DATA TYPE	DESCRIPTION
strWSUserName	String	The username for the Batch Account.
strWSPassword	String	The password for the Batch Account.

Response message:

```

HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetFileUploadHistoryResponse xmlns="http://fslso.com/BatchFiling">
      <GetFileUploadHistoryResult>
        <UploadHistory>
          <xsd:schema>schema</xsd:schema>xml</UploadHistory>
          <StatusCode>string</StatusCode>
          <StatusMessage>string</StatusMessage>
        </GetFileUploadHistoryResult>
      </GetFileUploadHistoryResponse>
    </soap12:Body>
  </soap12: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.

6. 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 Batch Filing* account is required to submit policy data in batch. This account is distinct and separate from your general SLIP account. Contact James Farmer at 1.800.562.4496 (ext. 116) at FLSO to receive a SLIP Batch Filing account.

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.