FLORIDA SURPLUS LINES SERVICE OFFICE

FSLSO Batch Filing Manual
Insurer Submissions

Schema Version 2.0
Updated November 2017

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 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 the FSLSO Batch Filing submission method.

1.2 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 FSLSO. A batch may have any number of policies of any type from any insurer.
FSLSO	Florida Surplus Lines Service Office
Insurer	Person or company that underwrites an insurance risk and; entity submitting the batch
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).

Term or Acronym	Definition or Expansion
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

1.3 Document Revision History

Date	Version	Description
12/20/2011	v1.0	Create Insurer Batch Manual
12/13/2012	1.1	Added new webservice call [Get Submission Response]

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

2. EXECUTIVE SUMMARY

The Surplus Lines Automation Suite (SLAS) is the technology solution for the reporting and payment of surplus lines policy data and premium taxes. The Surplus Lines Information Portal (SLIP), one of the main components of SLAS, allows agents and insurers to submit policy information electronically. SLAS has been implemented in more than 10 states, accounting for more than 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 2005.

The XML Batch Upload feature in SLIP can be used by agents and third party agency management software vendors. SLIP integrates seamlessly with agency 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 submissions is to configure the agency management system to export the policy data as XML so it can be uploaded manually as a single file. You can take automation further by using the 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.

2.1 FSLSO Batch Filing Manual Terms and Conditions

Use of the FSLSO Batch Filing submission methods (XML or API) constitute agreement with the following Terms and Conditions. The Florida Surplus Lines Service Office reserves the right to change or modify these terms of use without notice other than posting the amended Terms and Conditions in an updated version of this manual. Your continued use of the FSLSO Batch Filing methods after any changes in these Terms and Conditions constitutes your consent to such changes. The FSLSO Batch Filing methods are provided for Florida Surplus Lines Service Office customers for the purposes of submitting surplus lines policy data as required by F.S. 626.921.

TRADEMARK AND COPYRIGHT INFORMATION

All materials related to the FSLSO Batch Filing methods, including, but not limited to, manuals, schemas, text, scripts, data, custom graphics, logos, button icons, images, audio and video clips, links, digital downloads, data compilations, and software is owned, controlled by, or licensed to the Florida Surplus Lines Service Office and is protected by copyright, trademark, and other intellectual property rights. The content in this manual and related to the FSLSO Batch Filing methods may not be copied, reproduced, imitated, republished, modified, uploaded, posted, transmitted, or distributed in any way, including by e-mail or other electronic means, without the express prior written consent of the Florida Surplus Lines Service Office in each instance. You may download material intentionally made available for downloading, provided that you keep intact any and all copyright and other proprietary notices that may appear on such materials.

RESTRICTIONS

Users of the FSLSO Batch Filing methods will under no circumstances: (i) sell, rent, lease, sublicense, transfer, modify, create derivative works of or redistribute copyrighted or trademarked materials; (ii) alter or remove any copyright, patent, trademark or other protective notices included on the related materials; (iii) reverse engineer, decompile or disassemble the XML schema or API, or otherwise attempt to derive any related Source Code; (iv) use the XML schema, API or any related materials for purposes of building a competitive product or service; or (v) allow any third party to access or otherwise use the FSLSO Batch Filing methods (either with or without Your Member Agent Credentials), unless such third party has agreed to these Terms and Conditions.

Users shall not sell, produce for sale, or reproduce in any manner or allow third parties to sell or produce for sale, any material containing all or any part of the SLIP Software, SLIP Website application, XML Batch upload and web service API, FSLSO Tax Estimator, or any other proprietary materials.

INDEMNIFICATION

You agree to indemnify, defend, and hold harmless the Florida Surplus Lines Service Office and its affiliated companies, officers, directors, employees, agents, licensors, and suppliers from and against all losses, expenses, damages, and costs, including reasonable attorneys' fees, resulting from any violation by you of these Terms and Conditions. The Florida Surplus Lines Service Office reserves the right to assume the exclusive defense and control of any matter subject to indemnification by you.

SITE SECURITY

When providing information to the Florida Surplus Lines Service Office FSLSO Batch Filing methods, you agree to: (a) provide true, accurate, current and complete information and (b) maintain and promptly update such information to keep it true, accurate, current and complete. If you provide any information that is, or we have reasonable grounds to suspect that the information is, untrue, inaccurate, not current or incomplete, the Florida Surplus Lines Service Office may without notice suspend or terminate your access to our Site(s) and refuse any and all current or future use of our Site(s) (or any portion thereof). If any portion of our Site(s) requires you to register or open an account you may also be asked to choose a password and a user name. Please select a password that would not be obvious to someone trying to access your account, and change it regularly as an added precaution. You are responsible for maintaining the confidentiality of the password and account, and you are fully responsible for all activities that occur under your password or account identification. You agree to immediately notify the Florida Surplus Lines Service Office of any unauthorized use of your password or account or any other breach of security. Without limiting any rights which it may otherwise have, the Florida Surplus Lines Service Office reserves the right to take any and all action, as it deems necessary or reasonable, to ensure the security of the Sites and your account, including without limitation, terminating your account, changing your password, or requesting additional information to authorize transactions on your account. Notwithstanding the above, the Florida Surplus Lines Service Office may rely on the authority of anyone accessing your account or using your password and in no event and under no circumstances shall the Florida Surplus Lines Service Office be held liable to you for any liabilities or damages resulting from or arising out of (i) any action or inaction of the Florida Surplus Lines Service Office under this provision, (ii) any compromise of the confidentiality of your account or password and (iii) any unauthorized access to your account or use of your password. You may not use anyone else's account at any time, without the permission of the account holder.

Unauthorized access to or use of the FSLSO Batch Filing methods or any Florida Surplus Lines Service Office proprietary materials, product or service is strictly prohibited. Violating the security of any aforementioned site is prohibited and may result in criminal and civil liability. The Florida Surplus Lines Service Office may investigate incidents involving such violations and may involve and will cooperate with law enforcement if a criminal violation is suspected. Examples of security violations include, without limitation, unauthorized access to or use of data or systems including any attempt to probe, scan, or test the vulnerability of the Site(s) or to breach security or authentication measures, unauthorized monitoring of data or traffic, interference with service to any user, host, or network including, without limitation, mail bombing, news bombing, other flooding techniques, deliberate attempts to overload a system, forging any e-mail header, or any part of a message header and using manual or electronic means to avoid any use limitations.

PRIVACY POLICY

The collection of data through the FSLSO Batch Filing methods follows the privacy policy of the Florida Surplus Lines Service Office. To review the FSLSO privacy policy, please click here. Information furnished to the Florida Surplus Lines Service Office under the Surplus Lines Law is confidential and exempt from Section 119.07(1), Florida Statutes, and Section 24(a), Article I of the Florida Constitution if disclosure would reveal information specific to a particular policy or policyholder. The Florida Surplus Lines Service Office may provide such information to the Department in the furtherance of its duties and responsibilities. The exemption does not apply to any proceeding instituted by the Department or Office against an agent or insurer.

2.2 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 FSLSO 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.

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	&	&amp;
Apostrophe	‘	&apos;
Quote	“	&quot;
Less Than	<	&lt
Greater Than	>	&gt;

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 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 FSLSO 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 FSLSO.

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 broker or brokerage license number
4. Check for valid NAIC number.
5. 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 outlines the specific values for the elements in an XML schema that is prepared for submitting batch information to FSLSO. 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 Insurer Schema

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

3.1.8.2 Lloyds Schema

This schema is to be used by Lloyds of London submitting policies; covering both Florida only and Multistate policies and transactions for Brokers, Brokerages, and IPC filings.

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

3.1.8.3 Insurer Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<BatchDataSet SchemaVersion="1.0" Quarter="4" Year="2011" ReportingState="FL" 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
<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	50
<City>City1</City>	Contact City	1	1	1	50
<StateCode>AL</StateCode>	2 letter state abbreviation	0	1	2	2

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
<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>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 submission)	1	1	-	-
<LicenseNumber>A123456</LicenseNumber>	Broker License Number	1	1	7	7
<FirstName>Robert</FirstName>	Broker First Name	1	1	1	50
<LastName>Doe</LastName>	Broker Last Name	1	1	1	50

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<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	Unbound	-	-
<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 (The Attribute "XML_TransactionID" must be unique within the submission)	1	Unbound	-	-
<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				
</Brokers>	Ending Element				
</Brokers>	Ending Element				
<Brokerages>	Starting Element				

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Brokerage Xml_BrokerageID="0">	Starting Element (The Attribute "XML_BrokerageId" must be unique within the submission)	1	1	-	-
<LicenseNumber> L123456</LicenseNumber>	Brokerage License Number	1	1	7	7
<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	Unbound	-	-
<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 (The Attribute "XML_TransactionID" must be unique within the submission)	1	Unbound	-	-
<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				

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
</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				
<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
<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	Unbound	-	-
<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				

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
</Policies>	Ending Element				
</IPC >	Ending Element				
</BatchDataSet>	Ending Element				

3.1.8.4 Lloyds Schema

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<BatchDataSet SchemaVersion="1.0" Quarter="4" Year="2011" ReportingState="FL" SubmissionType="LLOYDS">	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
<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

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>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>8881234567</Fax>	Fax Number	1	1	10	10
</Contact>	Ending Element				
</Insurer>	Ending Element				
<Brokers>	Starting Element	1	1	-	-

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<Broker Xml_BrokerID="0">	Starting Element (The Attribute "XML_BrokerId" must be unique within the submission)	1	1	-	-
<LicenseNumber>A123456</LicenseNumber>	Broker License Number	1	1	7	7
<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	Unbound	-	-
<PolicyType>B</UMR>	'B' or 'O' 'Binder' or 'Open Market'	1	1	1	1
<UMR>B1234ASD</UMR>	The UMR will always begin with the letter 'B' followed immediately by 4 digits (signifying the Lloyd's broker), and contain up to 12 alphanumeric characters for a maximum of 17 characters. Separate transactions should be created in cases of multiple UMRs. For further information and examples, see this FAQ.	1	1	1	17
<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	-	-
<Policyholder>John Doe</ Policyholder >	Name of the policy holder	1	1	1	75
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element (The Attribute "XML_TransactionID" must be unique within the submission)	1	Unbound	-	-

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<EffectiveDateFrom>2014-01-31 </EffectiveDateFrom>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<EffectiveDateTo>2015-01-31 </EffectiveDateTo>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<Premium>1234567.89 </Premium>	Transaction Premium	1	1	1	10
<LPSONumber>65444 </ LPSONumber >	LPSO Number	1	1	5	5
<LPSODate>2015-01-31 </ LPSODate >	LPSO Date; (YYYY-MM-DD)	1	1	-	-
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</Brokers>	Ending Element				
<Brokerages>	Starting Element				
<Brokeager Xml_BrokerageID="0">	Starting Element (The Attribute "XML_BrokerageId" must be unique within the submission)	1	1	-	-
<LicenseNumber> L123456</LicenseNumber>	Brokerage License Number	1	1	7	7
<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	Unbound	-	-
<PolicyType>B</UMR>	'B' or 'O' 'Binder' or 'Open Market'	1	1	1	1

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
<UMR>B1234ASD</UMR>	The UMR will always begin with the letter 'B' followed immediately by 4 digits (signifying the Lloyd's broker), and contain up to 12 alphanumeric characters for a maximum of 17 characters. Separate transactions should be created in cases of multiple UMRs. For further information and examples, see this FAQ.	1	1	1	12
<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	-	-
<Policyholder>John Doe</ Policyholder >	Name of the policy holder	1	1	1	75
<Transactions>	Starting Element				
<Transaction Xml_TransactionID="0">	Starting Element (The Attribute "XML_TransactionID" must be unique within the submission)	1	Unbound	-	-
<EffectiveDateFrom>2014-01-31 </EffectiveDateFrom>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<EffectiveDateTo>2015-01-31 </EffectiveDateTo>	Transaction effective date; (YYYY-MM-DD)	1	1	-	-
<Premium>1234567.89 </Premium>	Transaction Premium	1	1	1	10
<LPSONumber>64555 </ LPSONumber >	LPSO Number	1	1	5	5
<LPSODate>2014-01-31 </ LPSODate >	LPSO Date; (YYYY-MM-DD)	1	1	-	-
</Transaction>	Ending Element				
</Transactions>	Ending Element				
</Policy>	Ending Element				
</Policies>	Ending Element				
</Brokerage>	Ending Element				

XML Structure	Description	Occurrence		Length	
		Min	Max	Min	Max
</Brokerages>	Ending Element				
</BatchDataSet>	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 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 FSLSO 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:

1. The insurer will generate an XML file containing the policy data they wish to submit to FSLSO. 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 insurers will log in to SLIP to upload the XML.

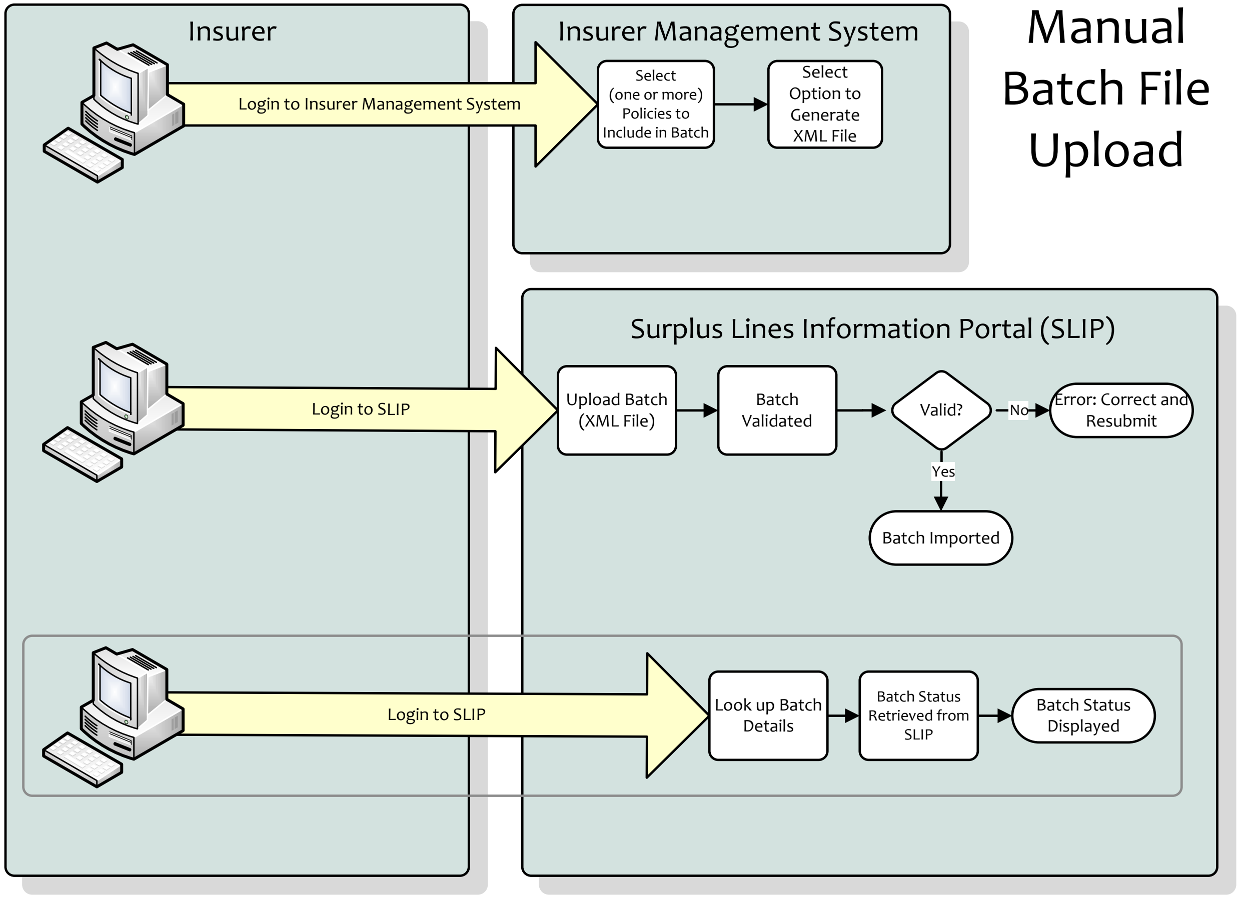
4.2 Pre-requisites

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

1. Insurers send a test batch XML file to FSLSO to ensure proper formatting. Contact James Farmer (jfarmer@fslso.com) at 1.800.562.4496 (ext. 116) at FSLSO to submit a test file..
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.


Manual
Batch File
Upload
Insurer
Login to Insurer Management System
Insurer Management System
Select (one or more) Policies to Include in Batch
Select Option to Generate XML File
Login to SLIP
Surplus Lines Information Portal (SLIP)
Upload Batch (XML File)
Batch Validated
Valid?
No
Error: Correct and Resubmit
Yes
Batch Imported
Login to SLIP
Look up Batch Details
Batch Status Retrieved from SLIP
Batch Status Displayed

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 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.

4.3.2 Log in to SLIP

Using a supported web browser, go to the FSLSO SLIP website (log onto FSLSO 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.

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.

4.3.3 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.

4.3.4 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 FSLSO. 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. FSLSO has not accepted a batch.
SUBMISSION ACCEPTED	The batch has been successfully imported into SLIP.

4.3.5 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 FSLSO.

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 insurer, to FSLSO. It also provides the means for the Insurer Management System to receive feedback in regards to the acceptance of the submitted data by FSLSO.

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

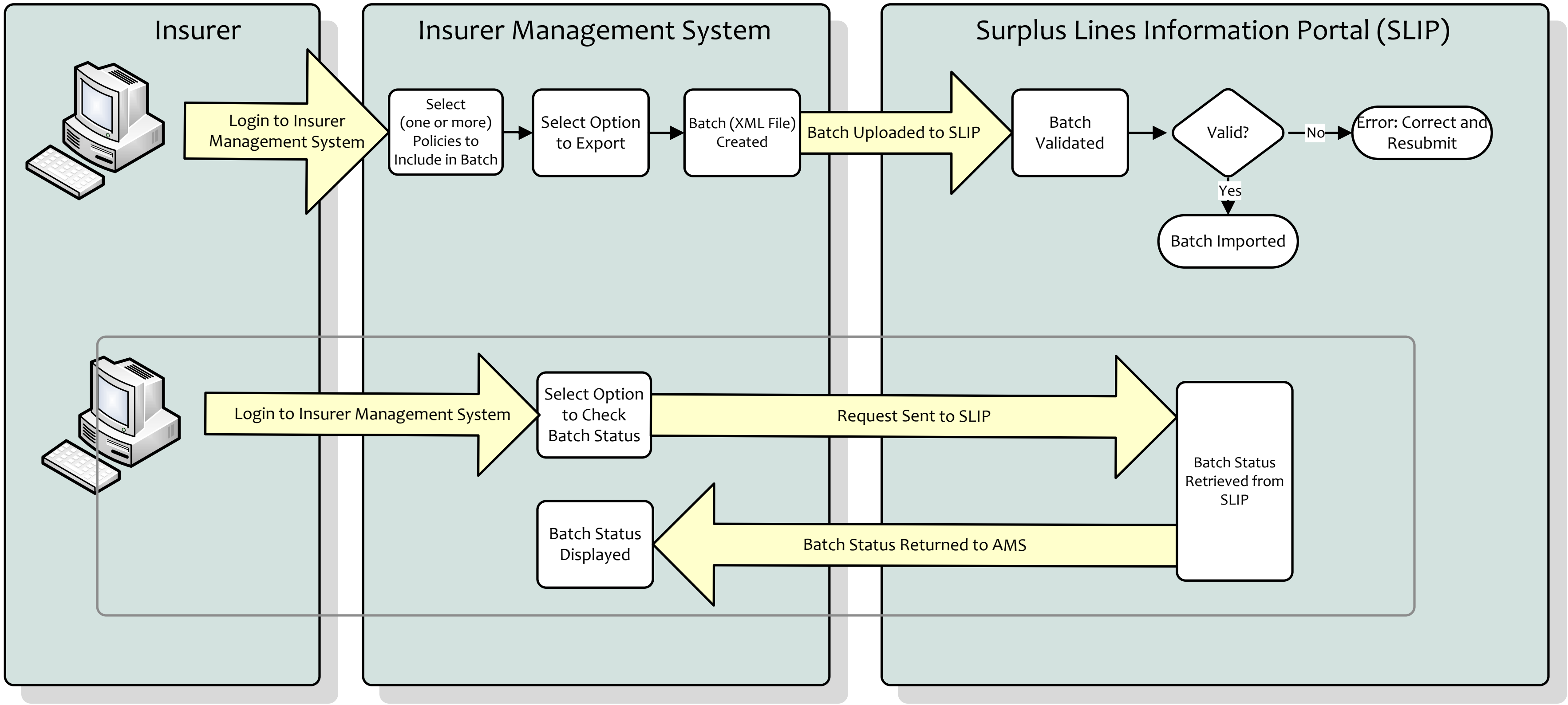
Insurers may submit policy data using the API Batch Submission Method if the following requirements are met:

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

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


Insurer
Insurer Management System
Surplus Lines Information Portal (SLIP)
Login to Insurer Management System
Select (one or more) Policies to Include in Batch
Select Option to Export
Batch (XML File) Created
Batch Uploaded to SLIP
Batch Validated
Valid?
No
Error: Correct and Resubmit
Yes
Batch Imported
Login to Insurer Management System
Select Option to Check Batch Status
Request Sent to SLIP
Batch Status Retrieved from SLIP
Batch Status Returned to AMS
Batch Status Displayed

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 FSLSO. 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. FSLSO 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 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.

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/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://fslso.com/BatchFiling">
      <UserName>string</UserName>
      <APIKey>string</APIKey>
    </AuthenticationHeader>
  </soap:Header>
  <soap:Body>
    <VerifyInsurerCredentials xmlns="http://fslso.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://fslso.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.

5.4.2 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 /FSLSOBatchFiling.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://fslso.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://fslso.com/BatchFiling">
      <UserName>string</UserName>
      <APIKey>string</APIKey>
    </AuthenticationHeader>
  </soap:Header>
  <soap:Body>
    <WebservicesBatchInsurerUpload xmlns="http://fslso.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://fslso.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.

5.4.3 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 /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)

5.4.4 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/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://fslso.com/BatchFiling">
      <UserName>string</UserName>
      <APIKey>string</APIKey>
    </AuthenticationHeader>
  </soap:Header>
  <soap:Body>
    <GetInsurerFileUploadHistory xmlns="http://fslso.com/BatchFiling">
      <slipUsername>string</slipUsername>
      <slipPassword>string</slipPassword>
    </GetInsurerFileUploadHistory>
  </soap:Body>
</soap:Envelope>
</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://fslso.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.

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 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.*