TIGERS STANDARDS

FED/STATE MODERNIZED eFILE

STATE SCHEMAS

November 14, 2009

CONTENTS

1.0 Introduction

1.1 Background

The ANSI ASC X12 Tax Information Interchange Task Group, informally known as "TIGERS," began work on schemas for state efiled Corporate returns under the IRS Modernized eFile (MeF) program in late 2003. Struggling to achieve uniformity across the Corporate returns of the various participating states, the group adopted the strategy of building a "master schema" based on "categories" of data, rather than representing any state's actual tax forms. All state schemas were to be subsets derived from the master schema. Fed/State Modernized eFile went live for Corporate returns in January 2006 utilizing this approach.

Two main problems soon surfaced with this approach. First, the process of mapping state forms and schedules to and from the categories was a burden and a barrier to both states and industry. Secondly, the master schema approach did not accommodate the volatility of tax law and the resulting changes in tax forms.

In June 2007 TIGERS voted to pursue an alternate approach which would allow state flexibility in creating schemas faithful to their forms and schedules. A prototype of this approach was created by a small team of states and industry, and was presented at the August 2007 TIGERS meeting. Out of that meeting, several workgroups of state and industry participants were formed to flesh out the new standard, which was first published in May 2008. Over the past two years, the schemas have been refined by collaborative work of TIGERS participants. Most recently, they have been updated in response to IRS modifications to support the upcoming MeF 1040 program. This document presents the results of this work, and has been voted on and approved by the TIGERS membership as the standard for Fed/State Modernized eFile for both business and individual state income tax returns.

1.2 Objectives

The proposed standard was developed with three main objectives:

a. States will be able to develop data content schemas based on the state's own forms, schedules, and worksheets. The names and sequencing of the data elements and complex types will be determined by the field names and positions on the forms and schedules, and will not have to follow a centrally controlled standard.

b. At the same time, the overall schema structures for the return and the schemas for common components such as headers and binary attachments, will be completely standardized. States will use these structures without alteration to achieve maximum uniformity while remaining forms-based. These standardized structures provide reuse across all states for the software development industry. They also provide a framework for states to utilize in building their schema sets.

c. State schemas will conform to IRS schema design for common components wherever feasible. IRS schemas were used as starting points for the construction of several common component schemas.

2.0 Common Schemas

The TIGERS Common schemas must be used by states exactly as they are provided. The only exception is the FinancialTransaction schema, which can be customized according to the guidelines given in sections 7.3.4/7.4.4. The latest release of the Common schema set is posted on the TIGERS web site, www.statemef.com, for TIGERS development use. A copy of Common is also included with both the individual and business return schema packages. For the development of state schemas, the state must use the copy of Common included with the appropriate individual or business return package, because the release level of the schemas may not be the same.

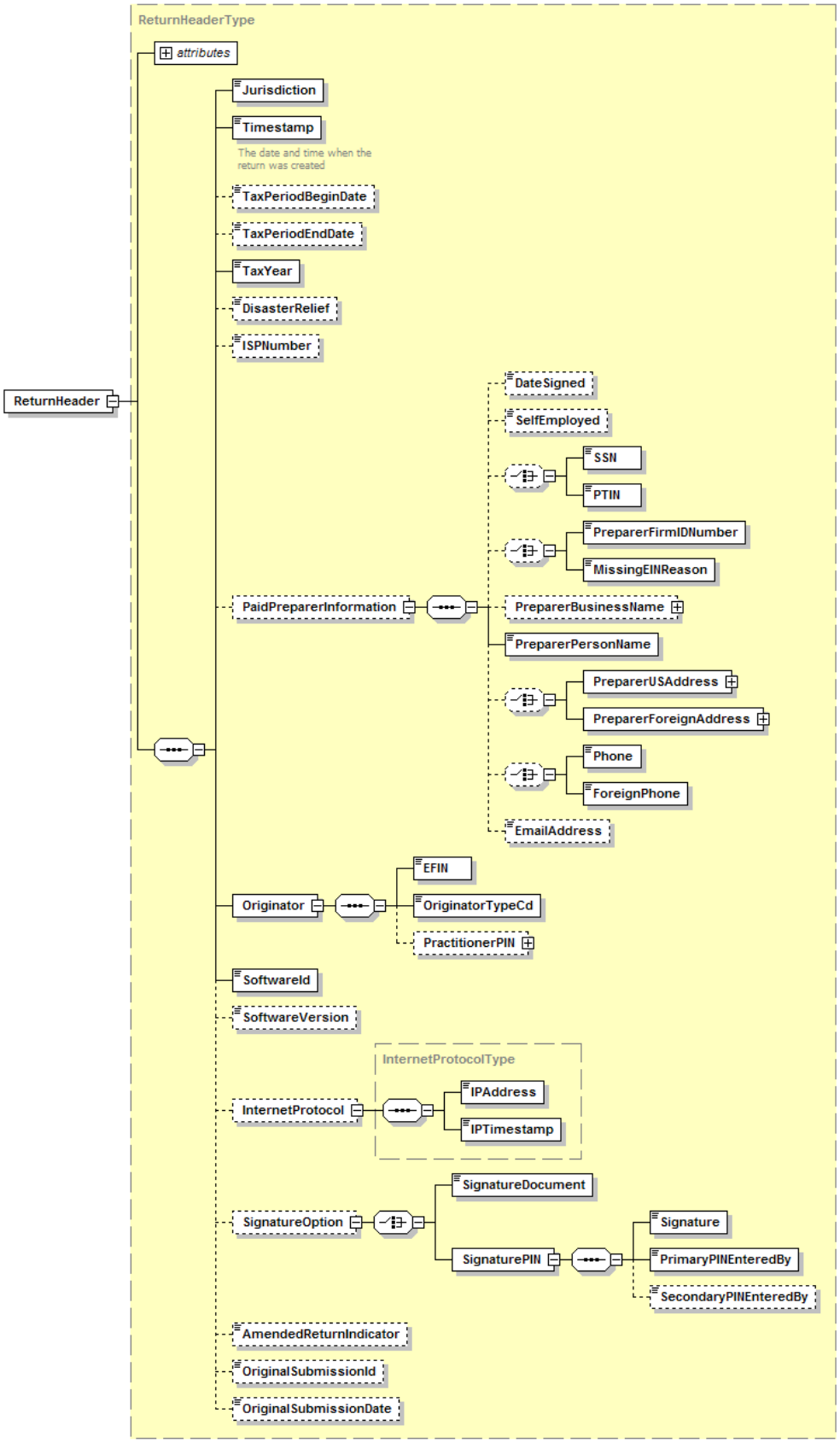
2.1 Header Common Core

The TIGERS Core Header is shown below. The ReturnHeader schema consists of elements taken from the IRS header schema used for 1120 and 1065 filing, with the Filer, Officer, Partner, and FilingType elements removed. It serves as a common core from which both the business and individual return header schemas are extended. This approach removes the need to maintain two sets of identical elements and keep them in synch. It is anticipated that future Fed/State MeF programs will also utilize this common core for the header.

Seven elements differ from the IRS model:

a. The Jurisdiction element is used to indicate standard abbreviation of the state or city to which the return applies.
b. The InternetProtocol complex type was simplified, and uses an edit pattern that supports both Version 4 and Version 6 Internet protocols.
c. Modified the PaidPreparerInformation complex type to make the person name of the preparer mandatory.
d. A state version of the SignatureOption complex type in order to support the option of a signature document, still in use by a number of states.
e. AmendedReturnIndicator is a checkbox element (that is, if it is present, its value is “X”) that has been added to the state headers to indicate whether the filing is that of an amended return. This was needed because a number of states utilize the same form for both original and amended returns.
f. OrigninalSubmissionID – the submission ID of the original submission, in the case where this submission was rejected, and the current return is in fact a re-submission of that return.
g. OriginalSubmissionDate – the data when the state return was originally submitted to the IRS. This is used by several states if the original return was denied by IRS and not received by the state.

Additionally, the elements in the ReturnHeader were re-ordered from the IRS model to place Timestamp, TaxPeriodBeginDate, TaxPeriodEndDate, and TaxYear together, and to place PreparerFirm, Preparer, and Originator together. This ordering was adopted by IRS for the MeF 1040 Header.

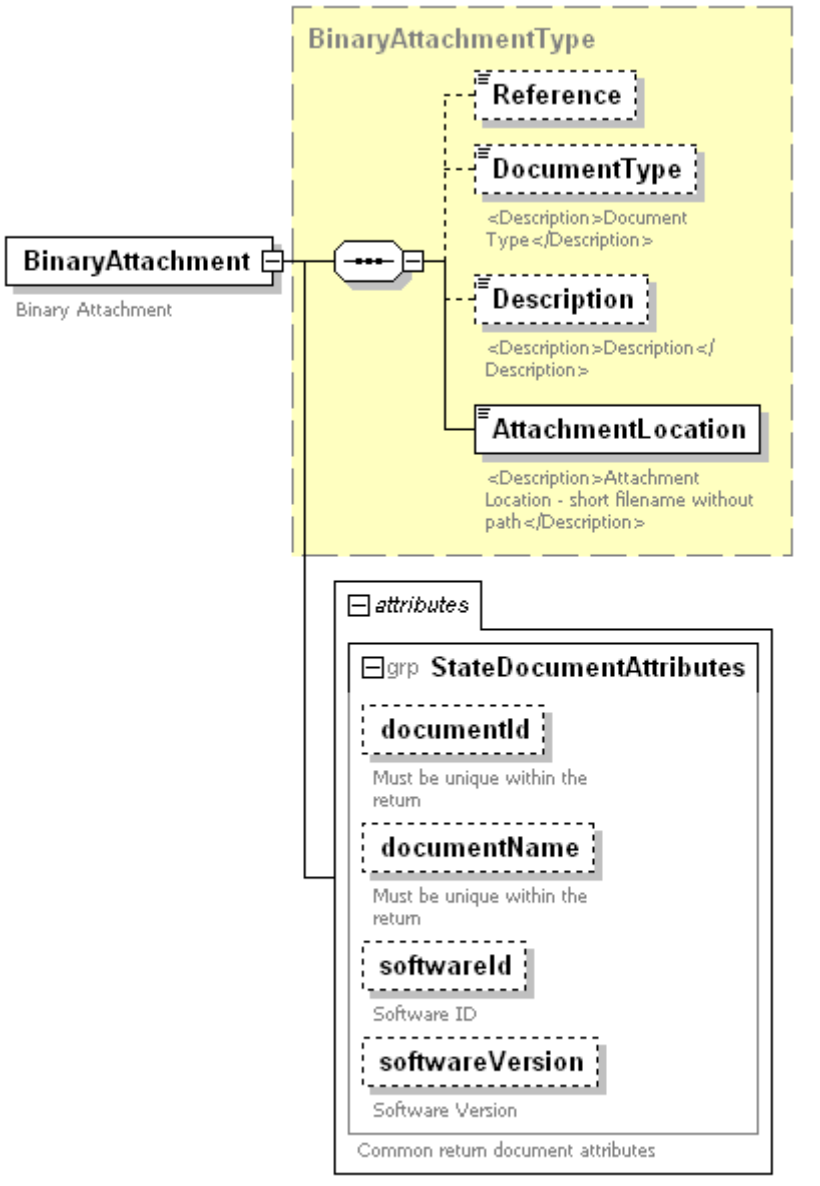


2.2 BinaryAttachments

The schema for binary attachments is shown below. It is used in both business and individual filings. This schema is identical to that used by IRS, except for the following.

a. The “Reference” element was added to contain the form or schedule, or line item to which the binary attachment pertains.

b. The IRS attributes, DocumentName, DocumentID, SoftwareID, and SoftwareVersion, were all made optional.



Generated by XmlSpy www.altova.com

The one mandatory element, AttachmentLocation, gives the filename of the attached PDF binary file, such as “CorporateBalanceSheet.pdf.”

2.3 FinancialTransaction

The schema for a Financial Transaction is shown below. This schema was developed for the Modernized eFile programs. It uses the data elements from the standard Automated Clearing House (ACH) records required to initiate an automated ACH debit or credit payment transaction. It has also been modified to handle the new International ACH Transaction (IAT) which was mandated by NACHA for use with money flowing from or to an international account, effective September 18, 2009. Key features of the FinancialTransaction schema include the following.

a. This schema is used by both business and individual filings.

b. This schema includes both the ability to make a payment and the ability to specify an account into which a refund should be deposited. This is because a filer may be claiming a refund on the current return, while making estimated payments toward a future return.

c. The ACHDebit complex type is used for the StatePayment element containing information for a payment of balance due on the current return. It is also used in the EstimatedPayments element to contain information for up to four quarterly estimated or declarations payments.

d. The AddendaRecord complex within ACHDebit allows a single financial transaction to be allocated to multiple taxpayers or periods, and may be used to allocate a bulk payment. Additionally sub-amounts, such as tax, penalty, and interest, may be broken out. This complex corresponds to the Addenda or “7” record in an ACH CCD+ or CTX transaction.

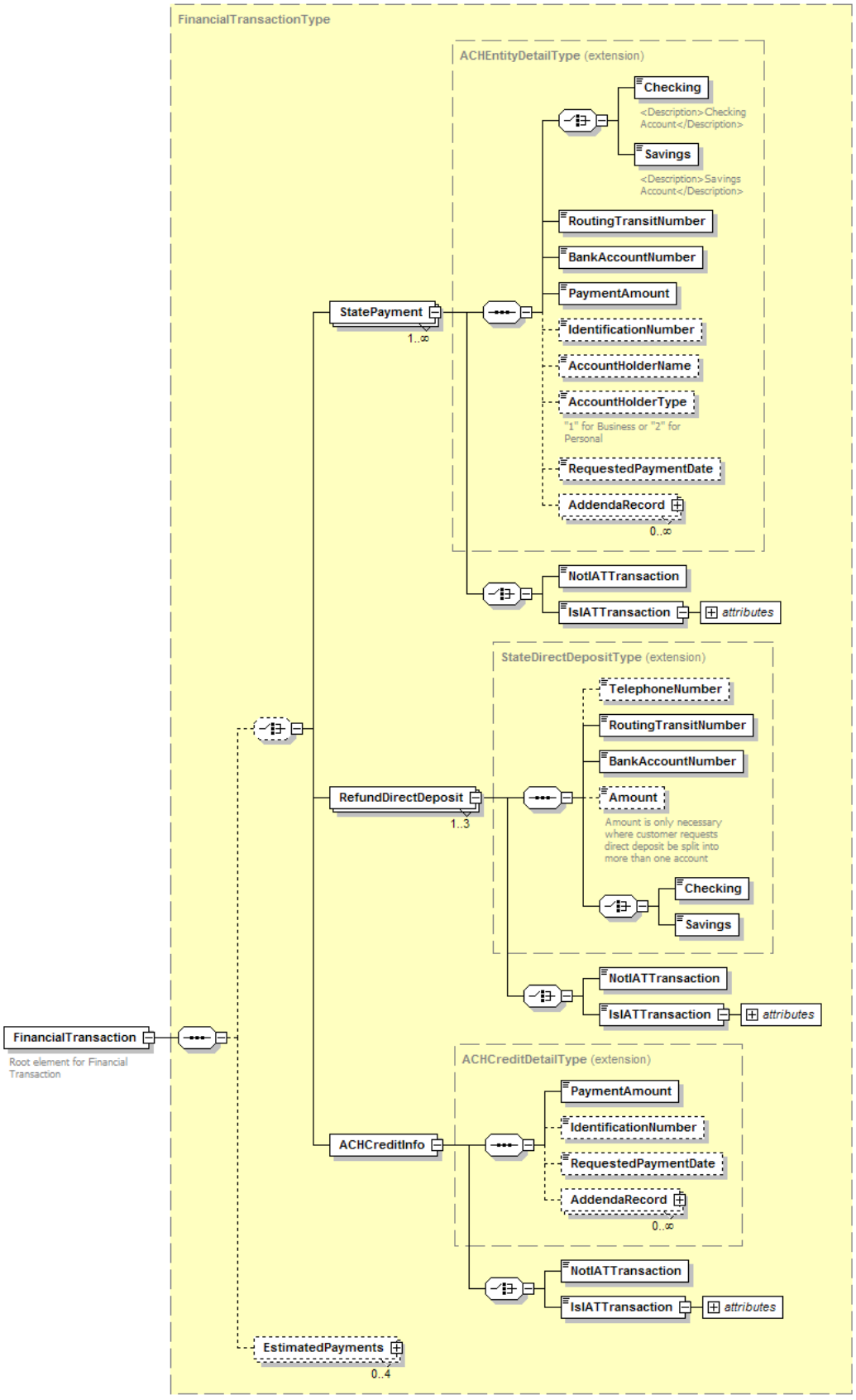
e. The RefundDirectDeposit complex occurs up to three times, for consistency with current IRS practice allowing an individual income tax refund to be split between up to three depository accounts.

f. The ACHCreditInfo complex is used to hold data to allow the state to match the return to an ACH credit transaction initiated by the taxpayer. This is an informational complex type only; the state cannot use it to actually initiate a transaction. ACHCreditInfo also contains the AddendaRecord complex.

g. Note that for the current return, there can be a choice of either an ACHDebit or ACHCreditInfo or a balance due or a RefundDirectDeposit, but not more than one. This is because any given return can either have a balance due, net zero, or a refund, but not more than one. At the same time, there can optionally be

from one to four estimated or declarations payments, regardless of the payment or refund status of the current return.

h. The original complex types – ACHEntityDetail, RefundDirectDeposit, and ACHCreditInfo, have all been extended by a choice of two new checkbox elements, NotIATTransaction and IsIATTransaction. This allows the state to determine whether or not an incoming financial transaction would require an IAT ACH transaction and handle the situation accordingly. For those states who plan to support the IAT, generally the only additional information needed from the taxpayer is the name of the taxpayer's bank. That information is provided in an attribute of IsIATTransaction.


FinancialTransactionType
ACHEntityDetailType (extension)
Checking
<Description>Checking Account</Description>
Savings
<Description>Savings Account</Description>
RoutingTransitNumber
BankAccountNumber
PaymentAmount
IdentificationNumber
AccountHolderName
AccountHolderType
"1" for Business or "2" for Personal
RequestedPaymentDate
AddendaRecord
0..∞
StatePayment
1..∞
NotIATTransaction
IsIATTransaction
attributes
StateDirectDepositType (extension)
TelephoneNumber
RoutingTransitNumber
BankAccountNumber
Amount
Amount is only necessary where customer requests direct deposit be split into more than one account
Checking
Savings
RefundDirectDeposit
1..3
NotIATTransaction
IsIATTransaction
attributes
FinancialTransaction
Root element for Financial Transaction
ACHCreditDetailType (extension)
PaymentAmount
IdentificationNumber
RequestedPaymentDate
AddendaRecord
0..∞
ACHCreditInfo
NotIATTransaction
IsIATTransaction
attributes
EstimatedPayments
0..4

2.4 EfileTypes

The name “efileTypes” (the one exception to upper camel case!) was coined by the IRS to refer to a collection of XML simple types and complex types used in electronic filing. These simple and complex types provide “building blocks” for schema development. There are two standard schemas containing efileTypes that are part of the TIGERS Common schema set. Additionally, there is a standard way for a state to provide additional efileTypes specifically for use by that state. Note that in order to use simple or complex types from any of the three efileTypes schemas in a state developed schema, the schema file(s) must be referenced by an include statement in the state developed schema.

Note that diagrams are not shown in this document for the two TIGERS standard efileTypes schemas, as each contains a large number of separate structures.

2.4.1 efileTypes.xsd

The TIGERS standard Common schema simply named **efileTypes.xsd** is supplied by IRS. It contains simple and complex types used in TIGERS schemas adapted from IRS schemas. TIGERS standard schema package includes a snapshot in time of the IRS schema. TIGERS will not necessarily issue a new release of the TIGERS standard if the IRS updates the eFileTypes schema. TIGERS will not make any additions, deletions, or changes to this schema, nor may the states alter the schema in any way. The IRS will have one eFileTypes schema for both business and individual efile programs. Simple and complex types should be used from this schema wherever feasible.

2.4.2 StateeFileTypes.xsd

The TIGERS standard Common **StateeFileTypes.xsd** schema is created and maintained by TIGERS. It contains the simple types and complex types used across the TIGERS standard schemas, plus simple types and complex types that are common to many state filings and are useful to be maintained centrally. Use of these StateeFileTypes is encouraged wherever basic “building blocks” such as AmountType or AddressType are needed, if the state cannot find usable components in efileTypes.xsd. This helps to obtain some measure of consistency across multiple states. States may not add to, delete, or change the simple or complex types in this schema.

Note that StateeFileTypes.xsd does not contain complex types designed to contain data found in tables within tax returns.

2.4.3 State Specific efileTypes

Finally, each state will have simple types and complex types unique to that state, that the state will need to use across multiple schemas, or which they wish to maintain in a centralized location. Each state will create a schema named **XXeFileTypes.xsd** where

“XX” refers to the standard state abbreviation, such as MDeFileTypes.xsd or SCeFileTypes.xsd. (Three characters are allowed for New York State and New York City.) This schema is completely under state control, and the state is responsible for creating the simple and complex types that comprise this schema as appropriate.

As an alternative, a state may choose to have separate individual and business efileTypes schemas, if differences between simple and complex types used in their individual and business schema sets justify the separation. In this case the two schemas must be named **XXIndividualeFileTypes.xsd** and **XXBusinesseFileTypes.xsd**, where again “XX” refers to the standard state abbreviation.

Note that a state will generally have some simple and complex types which are unique to a specific form or schedule. This is particularly true of complex types created to provide data in tables contained within the form or schedule. These simple and complex types may be coded inside the schema for the specific form or schedule, and do not have to reside in the state specific eFileTypes schema.

Note also that the state specific efileTypes schema is NOT part of the TIGERS Common schema set. Its packaging will be discussed later in this document.

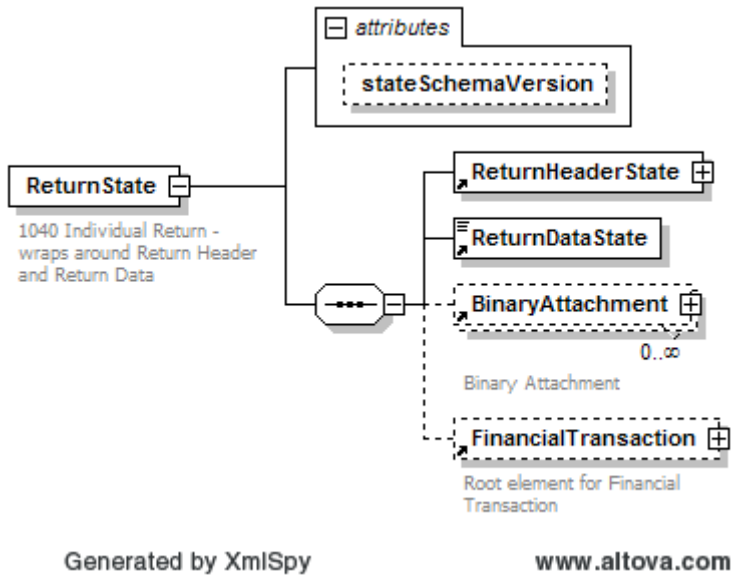
3.0 Schemas for Individual Income Tax Filing

The complete set of TIGERS standard schemas used for individual income tax filing is contained in the StateIndividualPackage schema set posted on www.statemef.com. This package is version controlled, and may only be changed through the TIGERS change control process. Note that the package contains a complete copy of the TIGERS Common schema set at the correct version and release level for the StateIndividualPackage. The TIGERS standard schemas specifically for individual income tax are contained in the sub-folder StateIndividual. **StateIndividual contains one schema – IndividualReturnHeaderState - that is controlled by TIGERS and may not be altered in any way by the states. It also contains two schemas – IndividualReturnState and ReturnDataState – that are to be used as patterns for states to use to create state-specific filings. How that is to be done is discussed later in this document. Another schema, IndividualStateEnumerations, may be added to by the state within the standards to be discussed later in this document.**

3.1 Individual Return

The overall structure for a state individual income tax return is shown below. Note that each element of the individual return schema is maintained as a separate schema, for ease of maintenance and for flexibility in use. These sub-schemas are then referenced in the overall return schema by use of include statements. The text version of the schema, which is really quite short, is shown below the structure diagram, to illustrate the use of “include” and “ref” functions.

The BinaryAttachments and FinancialTransaction schemas have been addressed as part of the Common schema set. The IndividualReturnHeaderState and ReturnDataState schemas will be discussed in the next sections.



Text of IndividualReturnState schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 sp1 U (http://www.altova.com) by SCDOR (SC DEPT OF
REVENUE & TAXATION) -->
<xsd:schema xmlns="http://www.irs.gov/efile" xmlns:efile="http://www.irs.gov/efile"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.irs.gov/efile" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0">
        <xsd:annotation>
                <xsd:documentation>
                        <Description>State e-file Individual Income Tax Schema</Description>
                </xsd:documentation>
        </xsd:annotation>
        <xsd:include schemaLocation="IndividualReturnHeaderState.xsd"/>
        <xsd:include schemaLocation="ReturnDataState.xsd"/>
        <xsd:include schemaLocation="../Common/BinaryAttachment.xsd"/>
        <xsd:include schemaLocation="../Common/FinancialTransaction.xsd"/>
        <!-- Individual Return - 1040 -->
        <xsd:element name="ReturnState">
                <xsd:annotation>
                        <xsd:documentation>1040 Individual Return - wraps around Return
Header and Return Data</xsd:documentation>
                </xsd:annotation>
                <xsd:complexType>
                        <xsd:sequence>
                                <xsd:element ref="ReturnHeaderState"/>
                                <!--ReturnDataState to be built by each state -->
                                <xsd:element ref="ReturnDataState"/>
                                <xsd:element ref="BinaryAttachment" minOccurs="0"
maxOccurs="unbounded"/>
                                <xsd:element ref="FinancialTransaction" minOccurs="0"/>
                        </xsd:sequence>
                        <xsd:attribute name="stateSchemaVersion" type="String20Type"/>
                        <!-- Version of the state schema set used for the filing-->
                </xsd:complexType>
        </xsd:element>
</xsd:schema>
```

Note the attribute stateSchemaVersion for the return. This allows the state to verify the value input by the return originator's software to the versions of the state schema package supported by the state.

Important concept: Although the IndividualReturnState schema cannot be modified by the state, each state will create versions of the schema differentiated by the contents of versions of the ReturnDataState schema, which is discussed in a later section below.

These versions of IndividualReturnState all have the same core elements, but are distinguished by their **file names**, such as “IndividualReturnSC1040.xsd” for a South Carolina long form and “IndividualReturnSC1040EZ.xsd” for a South Carolina short form. This allows the state to package separate schemas for different types of filings.

3.2 Individual Header

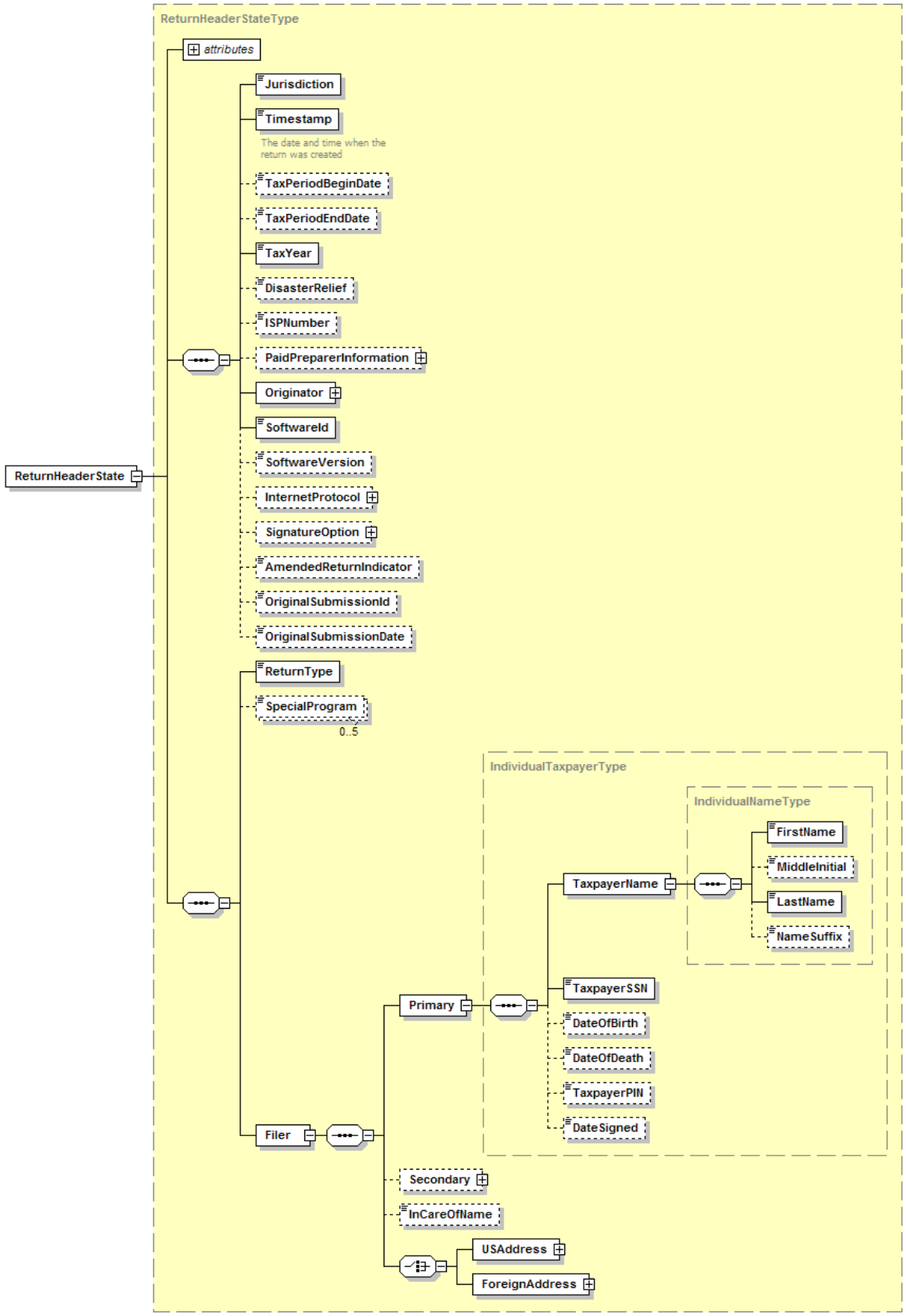
The header for an individual income tax return is shown below. This schema is an extension of the common core header schema. Note that at this time, the IRS has published two releases of its header for the individual income tax return; changes have been made in response to review by states and industry. The state IndividualReturnHeaderState schema has also been modified, to align more closely with the IRS schema. The chief differences from the TIGERS Common core header are the addition of the primary and secondary individual filers, including their SSNs, PINs, and optionally their dates of birth and death, the address for the return, and the ReturnType element.

The ReturnType element is used to provide the type of state filing, generally by use of a state form number to indicate, for example, “SC1040” for a long form filing, “SC1040EZ” for a short form filing, and “SC1040NR” for a non-resident filing. If a state chooses to specify the valid values for this element in their schema set, an enumerated list is to be used to contain the permissible values. This enumerated list is contained in a simple type ReturnTypeType, within the schema IndividualStateEnumerations. In this way, the IndividualReturnHeaderState schema is kept standard, while still allowing the state to customize the ReturnType element values. The IndividualStateEnumerations schema will be addressed further in a later section.

The optional SpecialProgram element is used to indicate a program, such as Free File Alliance or MultiLingual Initiative, in which the preparer/originator is participating. The programs themselves are enumerated in the StateEnumerations schema discussed in section 3.3.

The optional secondary filer element has the same structure as the primary filer. It is not shown on the diagram below simply for reasons of space on the page.

Note that a choice is given between a US-formatted address and a foreign address.


ReturnHeaderStateType
attributes
Jurisdiction
Timestamp
The date and time when the return was created
TaxPeriodBeginDate
TaxPeriodEndDate
TaxYear
DisasterRelief
ISPNumber
PaidPreparerInformation
Originator
SoftwareId
SoftwareVersion
InternetProtocol
SignatureOption
AmendedReturnIndicator
OriginalSubmissionId
OriginalSubmissionDate
ReturnHeaderState
ReturnType
SpecialProgram
0..5
IndividualTaxpayerType
IndividualNameType
FirstName
MiddleInitial
LastName
NameSuffix
TaxpayerName
TaxpayerSSN
DateOfBirth
DateOfDeath
TaxpayerPIN
DateSigned
Primary
Filer
Secondary
InCareOfName
USAddress
ForeignAddress

3.3 State Enumerations Lists

The IndividualStateEnumeration schema is shown below. This is the only TIGERS supplied schema within the StateIndividual package that may be modified by the states to customize it to their business requirements. The decision was made to provide separate enumeration list schemas in the StateIndividual and StateBusiness packages, rather than a single Common schema, because a given element may have different permissible values in individual vs. business filings,

As published by TIGERS, the IndividualStateEnumerations schema contains only two simple types, the ReturnTypeType and SpecialProgramType. ReturnTypeType is used in the IndividualReturnHeaderState schema. As noted previously, this usage allows the state to provide an enumerated list of valid values for the ReturnType element, without modifying the TIGERS standard IndividualReturnHeaderState schema. As shown below, the ReturnTypeType defaults to a TIGERS efileType called String20Type, which is a character string of length 20. Additionally, an example is given, commented out, of how to add an enumerated list to this simple type. A short list of two sample values "Form1" and "Form1A" is shown. States may choose to leave ReturnTypeType as a 20-character variable, using coded business rule edits outside the XML schema parser to check for valid values, or the state may replace the commented out section with an enumerated list of the state's return types. If the state replaces the commented out section with its own permissible values for ReturnType, then the schema parser will enforce the use of those values.

The second simple type provided in IndividualStateEnumerations is SpecialProgramType. This is an indicator used in the IndividualReturnHeader to hold any special program in which the return has participated, such as Free File. Again, the default is String20Type, but this can be overridden as shown in the commented out lines giving sample values of "Freefile" and "Spanish."

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.irs.gov/efile"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.irs.gov/efile" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0">
        <xsd:include schemaLocation="../Common/StateeFileTypes.xsd"/>
        <xsd:annotation>
                <xsd:documentation>This is a place for states to store enumerated lists of values
</xsd:documentation>
        </xsd:annotation>
        <xsd:simpleType name="ReturnTypeType">
                <xsd:annotation>
                        <xsd:documentation>Enumerate the specific Return Types that your state
accepts</xsd:documentation>
                </xsd:annotation>
```

```
            <xsd:restriction base="String20Type">
                <!--for example:
                <xsd:enumeration value="Form1"/>
                <xsd:enumeration value="Form1A"/>
                -->
            </xsd:restriction>
        </xsd:simpleType>
        <xsd:simpleType name="SpecialProgramType">
            <xsd:annotation>
                <xsd:documentation>Enumerate the specific special programs that your state supports</xsd:documentation>
            </xsd:annotation>
            <xsd:restriction base="String20Type">
                <!--for example:
                <xsd:enumeration value="Freefile"/>
                <xsd:enumeration value="Spanish"/>
                -->
            </xsd:restriction>
        </xsd:simpleType>
</xsd:schema>
```

The IndividualStateEnumerations schema is **not** limited just to the two simple types ReturnTypeType and SpecialProgramType. States may add other enumerated list variables to this schema according to business needs – enumerated lists utilized in state forms and schedules included in the ReturnDataState schema for the state. In this way the state can have one convenient place to maintain all enumerated lists, especially those that may be utilized in more than one of the state filing types.

3.4 ReturnDataState

The ReturnDataState schema included in the TIGERS StateIndividualPackage is only a stub – a schema with a root element only. This stub ReturnDataState is identical for the StateIndividual and StateBusiness packages. The state must retain that root element, and may not change its name, but will use that root element to create state-specific filing structures by including schemas for state forms and schedules for the respective individual filing program(s).

Key features of the ReturnDataState structure include the following.

a. The root element of the schemas for all forms are to begin with “Form” followed by the name of the form; root elements of schedules begin with “Sch” followed by the name of the schedule, and root elements of worksheets begin with “Wks” followed by the name of the worksheet. If any other common prefixes are desired, they can be proposed to TIGERS for a vote. In general the schema file names should equal the root elements.

b. All forms are to be maintained as separate schemas. All schedules and worksheets containing more than just a few lines are also to be maintained as separate schemas, and all schedules and worksheets, regardless of size, may be maintained as separate schemas. This allows forms, schedules, and worksheets to be assembled as needed for various filing types, without having to be maintained multiple times. It also allows the state to modify the schema for a form or schedule as business requirements dictate, without having to modify the schemas of forms or schedules that have not changed.

c. Schedules pertaining to a specific form are to be "attached" to the form through the use of an include statement and a reference element. Worksheets are to be attached to forms or schedules in the same manner. Extremely short schedules or worksheets may be included as complex types inline in the appropriate schemas; however, the use of the include statement and reference element is preferred where feasible. Schedules that apply to multiple forms may occur at the same level as the forms in the overall schema. In general, states are advised to keep the schema structure as "flat" as possible, by including forms and schedules for the filing at the same level.

d. Line item elements are to be named by the state, based on the actual line items on the forms, schedules, and worksheets. **TIGERS has published lists of most often used names for elements and complex types, based on the filings of a number of actual states. States are highly encouraged to utilize these names where they make sense; however their use is not mandatory.** Elements created by the state **must** follow the rules given in section 5.0.

e. Complex types are to be used both for structured data within a return, and to represent tables. **TIGERS has published a library of commonly used non-tabular complex types, based on the filings of a number of states. These complex types are available for use, and states are encouraged to use them as "building blocks" for the state schemas wherever feasible; however, their use is not mandatory.**

f. States are to create as many different versions of the ReturnDataState schema as the state needs to specify different types of filings. To continue the example used above, the schema IndividualReturnSC1040.xsd might include the schema ReturnDataSC1040.xsd, while the schema IndividualReturnSC1040EZ.xsd would contain the schema ReturnDataSC1040EZ.xsd. This allows the state to control which forms and schedules can be used for each type of filing. For example, the ReturnDataSC1040.xsd schema for a South Carolina long form may include a number of schedules that are not allowed to be filed with the South Carolina SC1040EZ form, and are therefore not included in the ReturnDataSC1040EZ.xsd schema.

4.0 Schemas for Business Income Tax Filing

The complete set of TIGERS standard schemas used for business income tax filing, including all filings in the Fed/State Corporate and Fed/State Partnership efile programs, is contained in the StateBusinessPackage schema set posted on www.statemef.com. This package is version controlled, and may only be changed through the TIGERS change control process. Note that the package contains a complete copy of the TIGERS Common schema set at the correct version and release level for the StateBusinessPackage. The TIGERS standard schemas specifically for business income tax are contained in the sub-folder StateBusiness. **StateBusiness contains two schemas – BusinessReturnHeaderState and BusinessReturnOtherHeaderState – that are controlled by TIGERS and may not be altered in any way by the states. StateBusiness also contains six schemas - BusinessReturnStateCombined, BusinessReturnStateSingle, ReturnDataState, ParentReturnDataState, ParentReturnState, SubsidiaryReturnDataState, and SubsidiaryReturnState – that are patterns for states to follow in creating state-specific schemas for business return filing. Instructions for how to do this are discussed later in this document. StateBusiness also contains one schema – BusinessStateEnumerations – that is under state control and may be modified within the standards to be discussed later in this document.**

4.1 Business Return

There are two structures for state business returns, discussed in the next two sections. One provides a complex structure for combining parent and subsidiary returns with an overall business return, which may in some states represent a consolidated return. The other return is a much simpler structure, without the parent or subsidiary returns. States may use these two structures as appropriate.

4.1.1 Combined Business Return

The overall structure for a “combined” state business tax return is shown below. Note that each element of the business return schema is maintained as a separate schema, for ease of maintenance and for flexibility in use. These sub-schemas are then referenced in the overall return schema by use of include statements and reference elements. The text version of the schema, which is really quite short in spite of its complexity, is shown below the structure diagram, to illustrate the use of “include” and “ref” functions.

The BinaryAttachments and FinancialTransaction schemas have been addressed as part of the Common schema set. The BusinessReturnHeaderState and ReturnDataState schemas will be discussed in the next sections.

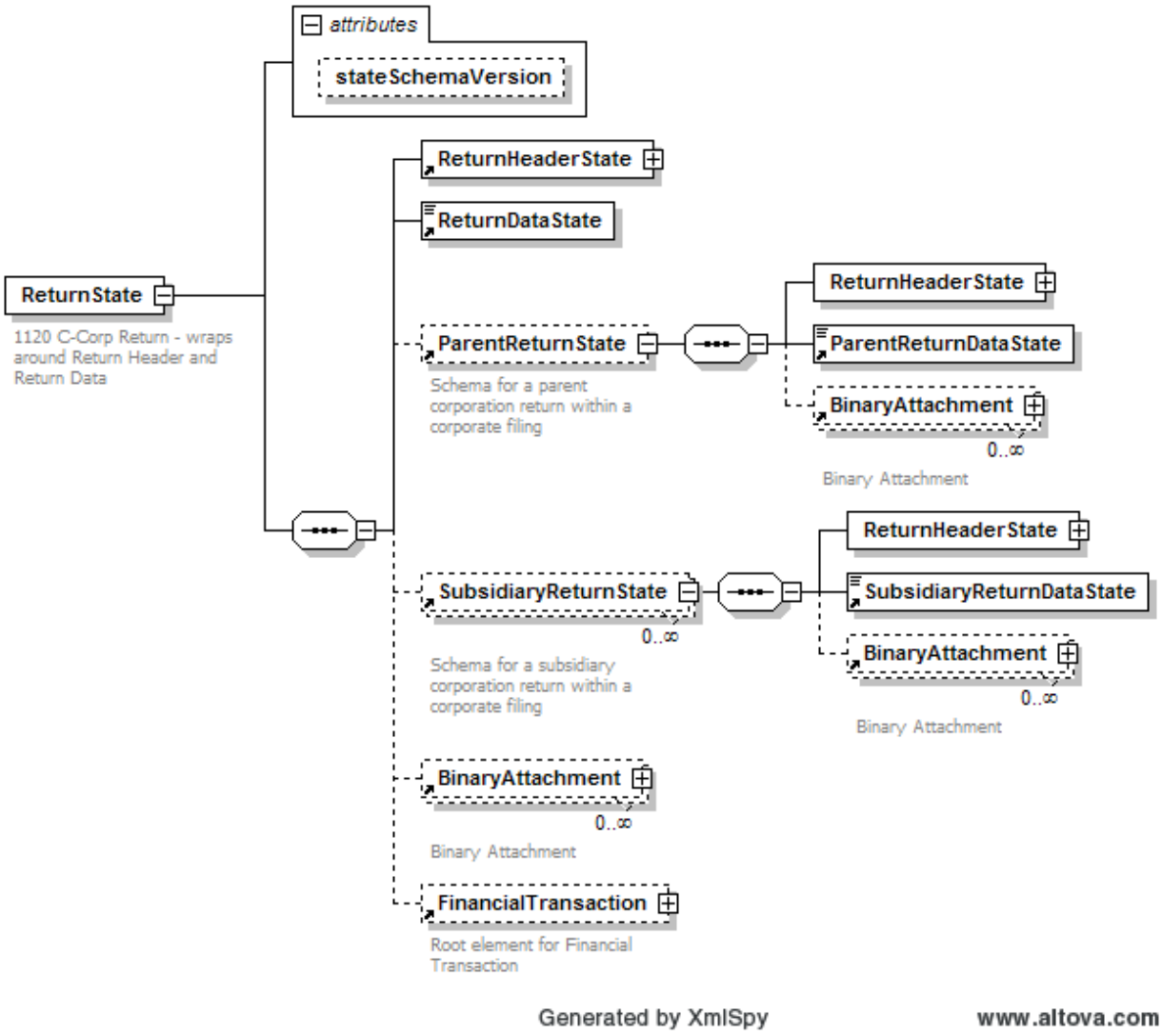
The following are key features of this schema.

a. The initial ReturnHeaderState and ReturnDataState contain the data for the overall or primary return filer corporation. This instance of ReturnHeaderState contains the full business return header.

b. The single parent and multiple subsidiary structures are optional as needed. Note that although they each contain the element ReturnHeaderState, these elements are **different** both from the full header above, and from each other. Their structures are taken from the IRS headers for parent and subsidiary. Note also that the parent and each subsidiary may have its own binary attachments.

c. The ReturnDataState elements shown in the schema are placeholders for the actual ReturnDataState elements that will be developed by the states for each of their filings. They are comprised of state forms, schedules, and worksheets as appropriate.

d. There is only one financial transaction for the entire filing.



Text of the combined business return schema follows, showing the use of “include” and “ref” statements:

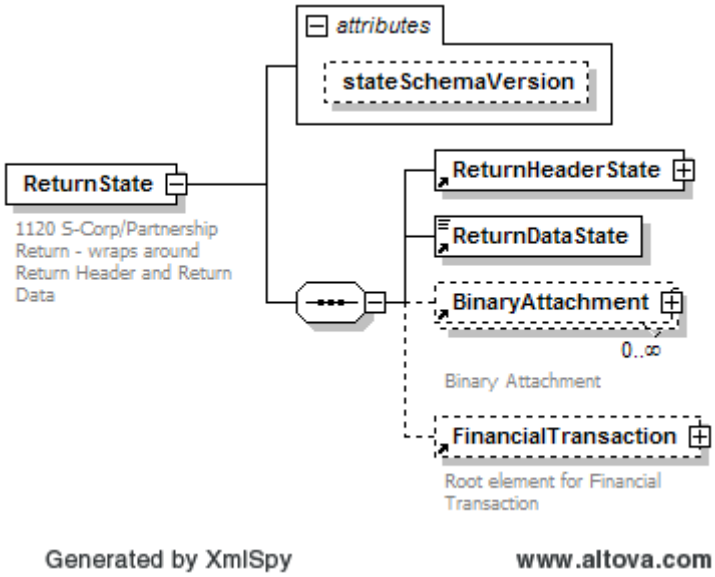
```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 sp1 U (http://www.altova.com) by SCDOR (SC DEPT OF REVENUE & TAXATION) -->
<xsd:schema xmlns="http://www.irs.gov/efile" xmlns:efile="http://www.irs.gov/efile"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.irs.gov/efile" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0">
        <xsd:annotation>
                <xsd:documentation>
                        <Description>State e-file Corporate Income Tax Schema</Description>
                </xsd:documentation>
        </xsd:annotation>
```

```
        <xsd:include schemaLocation="BusinessReturnHeaderState.xsd"/>
        <xsd:include schemaLocation="BusinessReturnOtherHeaderState.xsd"/>
        <xsd:include schemaLocation="ReturnDataState.xsd"/>
        <xsd:include schemaLocation="../Common/BinaryAttachment.xsd"/>
        <xsd:include schemaLocation="../Common/FinancialTransaction.xsd"/>
        <xsd:include schemaLocation="ParentReturnState.xsd"/>
        <xsd:include schemaLocation="SubsidiaryReturnState.xsd"/>
        <!-- Corporate Return - 1120 -->
        <xsd:element name="ReturnState">
                <xsd:annotation>
                        <xsd:documentation>1120 C-Corp Return - wraps around Return Header
and Return Data</xsd:documentation>
                </xsd:annotation>
                <xsd:complexType>
                        <xsd:sequence>
                                <xsd:element ref="ReturnHeaderState"/>
                                <!--ReturnDataState to be built by each state -->
                                <xsd:element ref="ReturnDataState"/>
                                <!-- Parent Return -->
                                <xsd:element ref="ParentReturnState" minOccurs="0"/>
                                <!-- Subsidiary 1120 Return -->
                                <xsd:element ref="SubsidiaryReturnState" minOccurs="0"
maxOccurs="unbounded"/>
                                <xsd:element ref="BinaryAttachment" minOccurs="0"
maxOccurs="unbounded"/>
                                <xsd:element ref="FinancialTransaction" minOccurs="0"/>
                        </xsd:sequence>
                        <xsd:attribute name="stateSchemaVersion" type="String20Type"/>
                        <!-- Version of the state schema set used for the filing-->
                </xsd:complexType>
        </xsd:element>
</xsd:schema>
```

4.1.2 Single Business Return

The single business return structure is a much simpler structure, as it does not allow parent or subsidiary returns. The structure is, in fact, essentially identical to the structure for the individual income tax return. This schema may be used for a simple corporation, without parent or subsidiaries. It may also be used for an S-corporation filing or a partnership filing, where parent and subsidiary returns do not apply and are not to be permitted by the schema.

Again, the BinaryAttachments and FinancialTransaction schemas have been addressed as part of the Common schema set. The BusinessReturnHeaderState and ReturnDataState schemas will be discussed in the next sections.



Text of the single business return is shown below, showing the use of “include” and “ref” statements.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 sp1 U (http://www.altova.com) by SCDOR (SC DEPT OF REVENUE & TAXATION) -->
<xsd:schema xmlns="http://www.irs.gov/efile" xmlns:efile="http://www.irs.gov/efile"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.irs.gov/efile" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0">
        <xsd:annotation>
                <xsd:documentation>
                        <Description>State e-file Corporate Income Tax Schema</Description>
                </xsd:documentation>
        </xsd:annotation>
        <xsd:include schemaLocation="BusinessReturnHeaderState.xsd"/>
        <xsd:include schemaLocation="BusinessReturnOtherHeaderState.xsd"/>
        <xsd:include schemaLocation="ReturnDataState.xsd"/>
        <xsd:include schemaLocation="../Common/BinaryAttachment.xsd"/>
        <xsd:include schemaLocation="../Common/FinancialTransaction.xsd"/>
        <!-- Corporate Return - 1120 -->
        <xsd:element name="ReturnState">
                <xsd:annotation>
                        <xsd:documentation>1120 S-Corp/Partnership Return - wraps around
Return Header and Return Data</xsd:documentation>
                </xsd:annotation>
                <xsd:complexType>
                        <xsd:sequence>
```

```
                        <xsd:element ref="ReturnHeaderState"/>
                        <!--ReturnDataState to be built by each state -->
                        <xsd:element ref="ReturnDataState"/>
                        <xsd:element ref="BinaryAttachment" minOccurs="0"
maxOccurs="unbounded"/>
                        <xsd:element ref="FinancialTransaction" minOccurs="0"/>
                    </xsd:sequence>
                    <xsd:attribute name="stateSchemaVersion" type="String20Type"/>
                    <!-- Version of the state schema set used for the filing-->
                </xsd:complexType>
        </xsd:element>
</xsd:schema>
```

4.2 Business Headers

The StateBusiness package contains two header schemas. The BusinessReturnHeaderState schema contains the mandatory header for the entire filing, whether combined or single. The BusinessReturnOtherHeaderState schema contains the parent and subsidiary headers. It is a direct copy of a schema provided by IRS for parent and subsidiary headers plus others, and it is used unchanged by TIGERS.

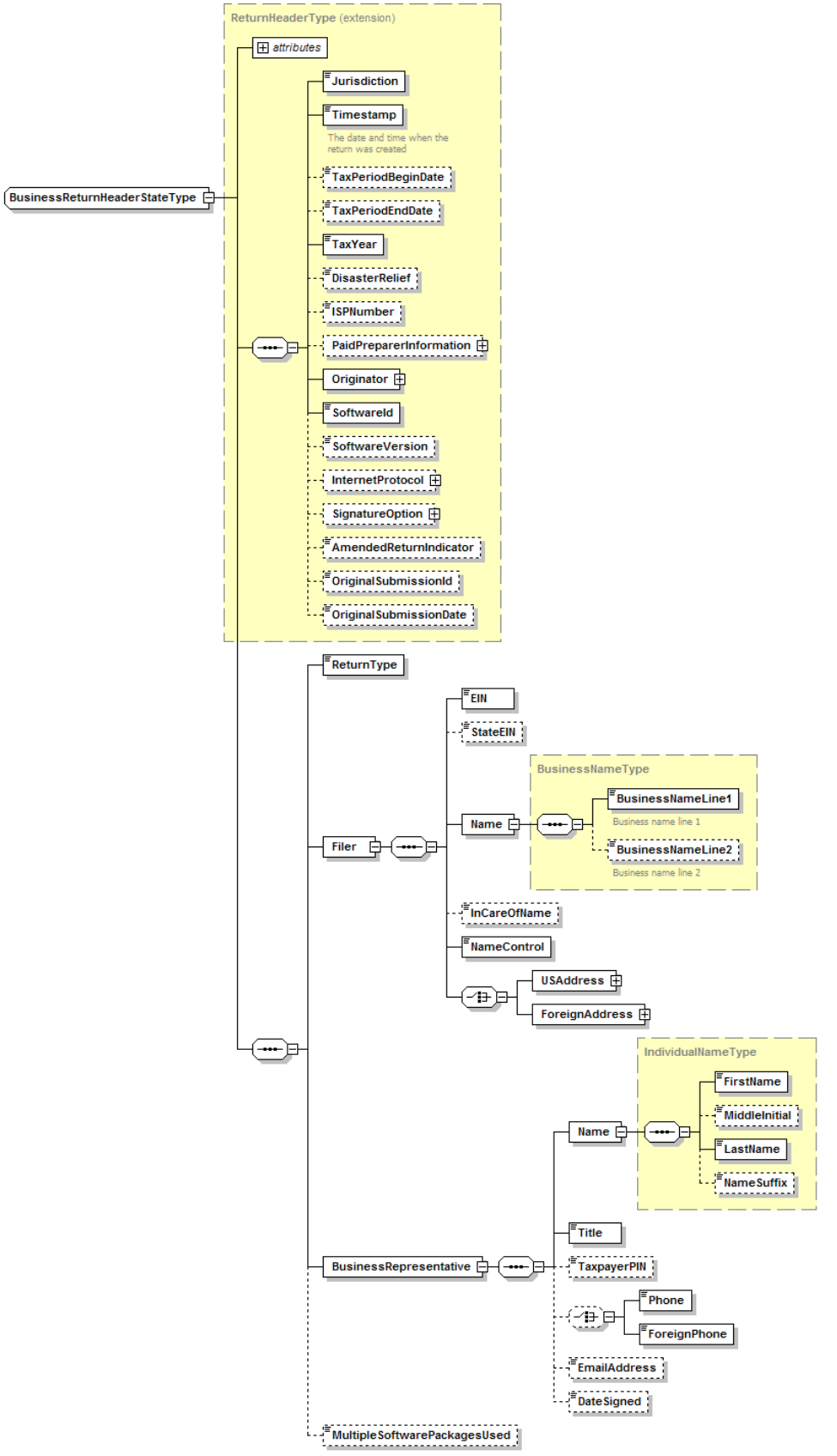
4.2.1 Full Business Header

The header for a business return filing, whether combined or single, is shown below. This schema is an extension of the TIGERS Common core header schema. The chief differences from the TIGERS Common core header are the addition of the Filer and BusinessRepresentative complex types and the ReturnType element, plus an indicator as used by IRS to indicate whether multiple software packages were used to create the electronic return.

The ReturnType element is used to provide the type of state filing, generally by use of a state form number to indicate, for example, “SC1120” for a C-corporation filing, or “SC1120S” for an S-corporation filing. If a state chooses to specify the valid values for this element in their schema set, an enumerated list is to be used to contain the permissible values. This enumerated list is contained in a simple type ReturnTypeType, within the schema BusinessStateEnumerations. In this way, the BusinessReturnHeaderState schema is kept standard, while still allowing the state to customize the ReturnType element values. The BusinessStateEnumerations schema will be addressed further in a later section.

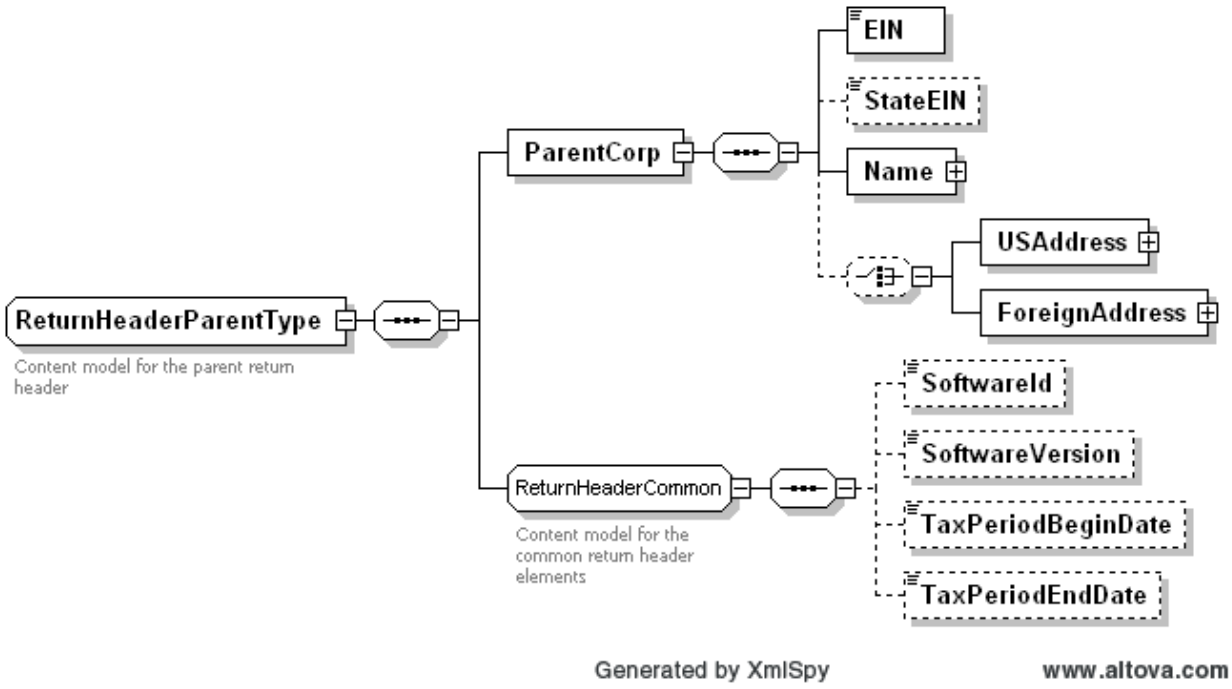
The Filer and BusinessRepresentative structures adhere closely to the equivalent IRS structures; the main modification is the inclusion of an optional state assigned identification number, called StateEIN, for the filer. The BusinessRepresentative complex type is equivalent to the Officer complex type in the IRS 1120 schema package

and the Partner complex type in the IRS 1065 schema package. It is used for an officer or partner as appropriate. Note that the name of the Filer is formatted as the name of the business, while the BusinessRepresentative name is formatted for an individual. Note that a choice is given between a US-formatted address and a foreign address for the filer.


BusinessReturnHeaderStateType
ReturnHeaderType (extension)
attributes
Jurisdiction
Timestamp
The date and time when the return was created
TaxPeriodBeginDate
TaxPeriodEndDate
TaxYear
DisasterRelief
ISPNumber
PaidPreparerInformation
Originator
SoftwareId
SoftwareVersion
InternetProtocol
SignatureOption
AmendedReturnIndicator
OriginalSubmissionId
OriginalSubmissionDate
ReturnType
Filer
EIN
StateEIN
Name
BusinessNameType
BusinessNameLine1
Business name line 1
BusinessNameLine2
Business name line 2
InCareOfName
NameControl
USAddress
ForeignAddress
BusinessRepresentative
Name
IndividualNameType
FirstName
MiddleInitial
LastName
NameSuffix
Title
TaxpayerPIN
Phone
ForeignPhone
EmailAddress
DateSigned
MultipleSoftwarePackagesUsed
Generated by XmlSpy
www.altova.com

4.2.2 Parent Header

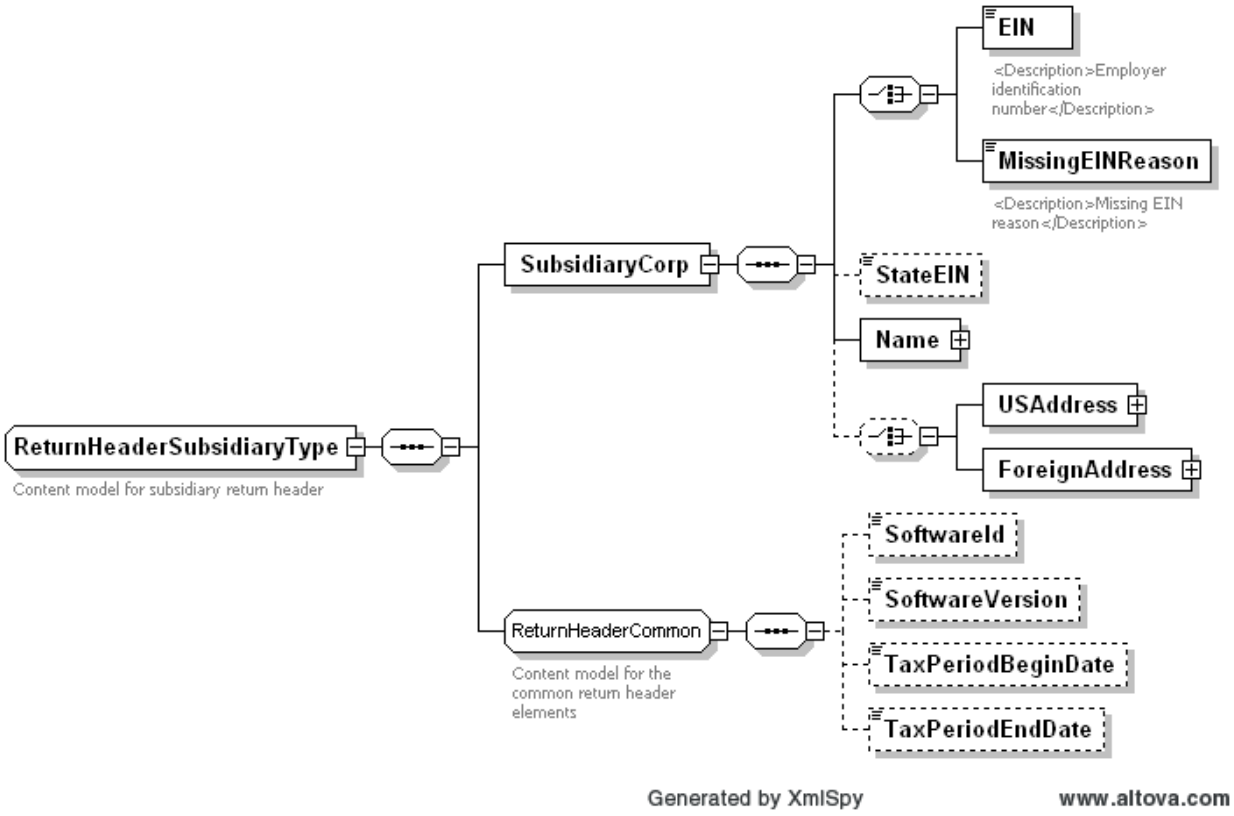
The header for a parent corporation sub-schema ParentReturnState is shown below. Note that this header is much simpler than the BusinessReturnHeaderState, as it does not contain data elements needed for the overall filing. Note that this parent header is actually a complex type, and not itself a sub-schema. This and the subsidiary header are the two exceptions to the TIGERS use of reference elements. This was done in order to copy the BusinessReturnOtherHeaderState schema unchanged from the IRS equivalent schema.



4.2.3 Subsidiary Header

The header for a subsidiary corporation sub-schema SubsidiaryReturnState is shown below. Note that this header is also much simpler than the BusinessReturnHeaderState, as it also does not contain data elements needed for the overall filing. Note that the subsidiary header is actually a complex type, and not itself a sub-schema. This, like the parent header above, are the two exceptions to the TIGERS use of reference elements. This was done in order to copy the BusinessReturnOtherHeaderState schema unchanged from the IRS equivalent schema.

Note that in accordance with IRS practice, the overall corporation as given in the BusinessReturnHeaderState and the parent corporation must both provide federal Employer Identification Numbers (EIN), while the subsidiary may not have an EIN.



4.3 State Enumeration Lists

The BusinesStateEnumeration schema is show below. This is the only TIGERS supplied schema within the StateBusiness package that may be modified by the states to customize it to their business requirements. The decision was made to provide separate enumeration list schemas in the StateIndividual and StateBusiness packages, rather than a single Common schema, because a given element may have different permissible values in individual vs. business filings,

As published by TIGERS, the BusinessStateEnumerations schema contains only one simple type, the ReturnTypeType, which is used in the BusinessReturnHeaderState schema. As noted previously, this usage allows the state to provide an enumerated list of valid values for the ReturnType element, without modifying the TIGERS standard BusinessReturnHeaderState schema. As shown below, the ReturnTypeType defaults to a TIGERS efileType called String20Type, which is a character string of length 20. Additionally, an example is given, commented out, of how to add an enumerated list to this simple type. A short list of two sample values “Form1” and “Form1A” is shown. States may choose to leave ReturnTypeType as a 20-character variable, using coded edits outside the XML schema parser to check for valid values, or the state may replace the commented out section with an enumerated list of the state’s return types.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns="http://www.irs.gov/efile"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.irs.gov/efile" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0">
        <xsd:include schemaLocation="../Common/StateefileTypes.xsd"/>
        <xsd:annotation>
                <xsd:documentation>This is a place for states to store enumerated lists of values
</xsd:documentation>
        </xsd:annotation>
        <xsd:simpleType name="ReturnTypeType">
                <xsd:annotation>
                        <xsd:documentation>Enumerate the specific Return Types that your state
accepts</xsd:documentation>
                </xsd:annotation>
                <xsd:restriction base="String20Type">
                        <!--for example:
                        <xsd:enumeration value="Form1"/>
                        <xsd:enumeration value="Form1A"/>
                        -->
                </xsd:restriction>
        </xsd:simpleType>
</xsd:schema>
```

The BusinessStateEnumerations schema is **not** limited just to the one simple type ReturnTypeType. States may add other enumerated list variables to this schema according to business needs – enumerated lists utilized in state forms and schedules included in the ReturnDataState schema for the state. In this way the state can have one convenient place to maintain all enumerated lists, especially those that may be utilized in more than one of the state filing types.

4.4 ReturnDataState

The ReturnDataState schema included in the TIGERS StateBusinessPackage is only a stub – a schema with a root element only. This stub ReturnDataState is identical for the StateIndividual and StateBusiness packages. The state must retain that root element, and may not change its name, but will use that root element to create state-specific filing structures by including schemas for state forms and schedules for the respective business filing program(s).

Key features of the ReturnDataState structure include the following.

a. The root element of the schemas for all forms are to begin with “Form” followed by the name of the form; root elements of schedules begin with “Sch” followed by the name of the schedule, and root elements of worksheets begin with “Wks” followed by the name of the worksheet. If any other common prefixes are

desired, they can be proposed to TIGERS for a vote. In general the schema file names should equal the root elements.

b. All forms are to be maintained as separate schemas. All schedules and worksheets containing more than just a few lines are also to be maintained as separate schemas, and all schedules and worksheets, regardless of size, may be maintained as separate schemas. This allows forms, schedules, and worksheets to be assembled as needed for various filing types, without having to be maintained multiple times. It also allows the state to modify the schema for a form or schedule as business requirements dictate, without having to modify the schemas of forms or schedules that have not changed.

c. Schedules pertaining to a specific form are to be "attached" to the form through the use of an include statement and a reference element. Worksheets are to be attached to forms or schedules in the same manner. Extremely short schedules or worksheets may be included as complex types inline in the appropriate schemas; however, the use of the include statement and reference element is preferred where feasible. Schedules that apply to multiple forms may occur at the same level as the forms in the overall schema. In general, states are advised to keep the schema structure as "flat" as possible, by including forms and schedules for the filing at the same level.

d. Line item elements are to be named by the state, based on the actual line items on the forms, schedules, and worksheets. **TIGERS has published lists of most often used names for elements and complex types, based on the filings of a number of actual states. States are highly encouraged to utilize these names where they make sense; however their use is not mandatory.** Elements created by the state **must** follow the rules given in section 5.0.

e. Complex types are to be used both for structured data within a return, and to represent tables. **TIGERS has published a library of commonly used non-tabular complex types, based on the filings of a number of states. These complex types are available for use, and states are encouraged to use them as "building blocks" for the state schemas wherever feasible; however, their use is not mandatory.**

f. States are to create as many different versions of the ReturnDataState schema as the state needs to specify different types of filings. To continue the example used above, the schema BusinessReturnSC1120.xsd might include the schema ReturnDataSC1120.xsd, while the schema BusinesReturnSC1120S.xsd would contain the schema ReturnDataSC1120S.xsd. This allows the state to control which forms and schedules can be used for each type of filing. For example, the ReturnDataSC1120.xsd schema for a South Carolina C-corporation may include a number of schedules that are not allowed to be filed with the South Carolina SC1120S form, and are therefore not included in the ReturnDataSC1120S.xsd schema.

5.0 Data Element Rules

All data elements created by the state as part of the state form schemas must follow these rules.

a. Names can contain letters, numbers, and other characters that are not reserved for use in XML syntax. However, use of special characters is discouraged because some parsers may not support a specific character.

b. Names cannot start with a number, punctuation, or special character.

c. Names must not start with the letters xml (or XML or Xml).

d. Names cannot contain spaces or underscores. Hyphens may be used if part of a word contained in the name, such as a form name, but should not be used to concatenate words within the data element name.

e. Element names are “Upper Camel Case” – words are directly concatenated and the first letter of each word is capitalized. For example: school district code would SchoolDistrictCode – no spaces or underscores.

f. Attribute names are “LowerCamelCase” - words are concatenated and the first letter of the first lower case and the first letter of al remaining words are capitalized. For example: school district code would schoolDistrictCode. Attributes are to be used for metadata – that is, data about elements. Primary line item data is to be conveyed in elements, not attributes.

g. Names **cannot** be longer than 30 characters, due to restrictions imposed by some database products. However, tag name should be as short and simple as possible to still be clear and descriptive.

h. Element names must be meaningful. Do not use the line numbers in the names of elements. For example, do not create element names like ‘Line5’ or ‘Line5MinusLine3’. Line numbers are subject to change annually (if not more frequently) as data fields are added to or removed from forms. Tying the element names to line numbers leads to unnecessary schema code maintenance as elements are added or removed mid-form and other elements shift line number. It also defeats the intent of descriptive XML tags in making the incoming return (instance document) easy to read and debug as necessary.

6.0 Namespaces

Namespaces are used within XML to establish XML “environments” in which data elements, complex types, and schemas form a consistent whole. Namespace usage is one of the more sophisticated and least understood features of XML. All schemas within the TIGERS schema set utilize the IRS modernized efile namespace, including the specification ‘targetNamespace=http://www.irs.gov/efile’ in each schema. This means that each state must utilize the same namespace specifications in all of the schemas in the state packages.

The use of the IRS namespace is due to the XML syntax rules for including one schema within another schema. Both source and target schemas must be of the same namespace. This means that in order for TIGERS to directly utilize IRS schemas such as efileTypes.xsd or the schemas for the forms W-2 or 1099, the TIGERS schemas must be of the same namespace. Likewise, in order for the state schemas to include TIGERS efileTypes schemas header structures, and included IRS schemas in the state return data and return filing structures, those state schemas must also be of the same namespace.

Although states may question their need to utilize the IRS namespace, this is a TIGERS standard, and must be followed in order for the TIGERS provided schemas to be standard and parse identically across multiple states.

7.0 Schema Packaging

The term “packaging” is used here to refer to the structure of folders and sub-folders used to contain the various schemas when posted by the states for industry use. It is desirable to maintain as much uniformity as possible in schema packaging, so that a software developer can use a single approach to accessing the schemas in the packages that the software developer downloads from as many states as possible. At the same time, there are acknowledged differences between a state that is only supporting one MeF program, a state that is supporting two programs separately, and a state that is implementing an integrated program with both individual and business filings. The standards below are intended to allow that flexibility while maximizing uniformity.

7.1 TIGERS Schema Packages

The TIGERS supplied schema packages are available at www.statemef.com. As noted previously, the Common package is available as standalone; however, the appropriate version/release of Common is included in the StateIndividual and StateBusiness packages.

Each package has a version/release number indicated by “Vxx.yy” where xx is the version number and yy is the release number. A change in release number only (a “minor release”) means that the release is backwards compatible; that is, any instance document generated from the previous release will still validate against the new release. In general, a change in release number only represents the addition of optional new components. A change in version number (a “major release”) means that the release is **not** backwards compatible; that is, an instance document generated from the previous release may not be valid against one or more schemas in the new release.

7.2 State Schema Packages

State schema packages are to be named xxAAAYYYYVn.n where

- xx is the standard state abbreviation (xxx for NYS or NYC)
- AAA is :”Business” for a state business package, “Individual” for a state individual package, or “MeF” for those states wishing to publish an integrated package
- YYYY is the tax year
- Vn.n is the version/release number, such as V1.2 for version 1, release 2.

7.2.1 State Individual Packages

A sample folder structure for a state individual package is shown below.

- xxIndividualYYYYVn.n (package version)
  - Common\ (TIGERS Common from StateIndividualPackage)
    - ACHDebit.xsd
    - BinaryAttachment.xsd
    - efileTypes.xsd
    - FinancialTransaction.xsd
    - ReturnHeader.xsd
    - StateDirectDeposit.xsd
    - StateeFileTypes.xsd
  - StateIndividual
    - IRSForms
      - IRSW2
      - IRS1099R
      - StateW2G
      - State1099G
      - State1099MISC
      - State1099INT
    - IndividualReturnHeaderState.xsd
    - IndividualStateEnumerations.xsd
  - xxCommon\ (common across xx business and individual schemas)
    - xxeFileTypes.xsd
    - FormXXX.xsd
    - FormZZZ.xsd
    - ScheduleB.
  - xxIndividual\
    - IRSForms
      - IRS111
      - IRS222
      - IRS333
    - xxForms\
      - FormNNN.xsd
      - FormYYY.xsd
      - ScheduleA.xsd
      - WksIII.xsd
    - ReturnDataxxNNN.xsd
    - ReturnDataxxYYY.xsd
    - IndividualReturnxxNNN.xsd
    - IndividualReturnxxYYY.xsd

Note that the names of the schema files for all state forms consist of “Form” followed by the state form name; schemas for schedules are named “Sch” followed by the state schedule name; and schemas for worksheets are named “Wks” followed by the name of the state worksheet.

The TIGERS standard ReturnDataState schema is deleted from StateIndividual, and is replaced by ReturnDataxx schemas in the xxIndividual folder for each type of state filing, for example ReturnDataSC1040 and ReturnDataSC1040A. This allows the state to tailor the forms, schedules and worksheets included in the ReturnDataState structure for each type of filing.

Similarly, the TIGERS standard IndividualReturnState schema is deleted from StateIndividual and is replaced by IndividualReturnxx schemas in the xxIndividual folder for each type of state filing. Each IndividualReturnxx schema exactly duplicates the TIGERS structure, except for including the appropriate ReturnDataxx schema for the specific filing type. For example, ReturnSC1040 would include ReturnDataSC1040 as a sub-schema.

7.2.2 State Business Packages

Below is a sample folder structure for a state business schema package. Note that it is essentially the same structure as the individual package. The StateBusiness folder from the TIGERS StateBusinessPackage replaces the StateIndividual folder. As discussed previously, the TIGERS issued StateBusiness folder contains structures for both a simple filing, BusinessReturnStateSingle, which is essentially the same structure as IndividualReturnState, and BusinessReturnStateCombined, which is a more complex structure to accommodate filings containing returns for parent and subsidiary corporations. States must determine whether they need to utilize one or both of these structures in building their own BusinessReturnxx schemas. In doing so, the TIGERS standard BusinessReturnStateCombined and BusinessReturnStateSingle, ParentReturnState, and SubsidiaryReturnState schemas are deleted from the StateBusiness folder, since they are only patterns, and are replaced by the BusinessReturnxx schemas in the xxBusiness folder. Similarly, the ReturnDataState, ParentReturnDataState, and SubsidiaryReturnDataState schemas, all of which are just stubs, are deleted from StateBusiness and replaced by state specific equivalents in xxBusiness.

Most states at time of this writing intend to release separate Business and Individual packages, so that a change to one does not require a change in the release of the other. A state releasing an integrated package would incorporate schemas from both StateBusiness and StateIndividual into the above structure.

Note that the names of the sub-folders are version independent. This is done so that it is not necessary to modify every schema in a folder each time there is a new release. Otherwise, if the folder names changed, the include statements would also have to change to refer to the new folders.

- xxBusinessYYYYVn.n (package version)
  - Common\ (TIGERS Common from StateBusinessPackage)
    - ACHDebit.xsd
    - BinaryAttachment.xsd
    - efileTypes.xsd
    - FinancialTransaction.xsd
    - ReturnHeader.xsd
    - StateDirectDeposit.xsd
    - StateeFileTypes.xsd
  - StateBusiness\ (TIGERS StateBusiness from StateBusinessPackage)
    - BusinessReturnHeaderState.xsd
    - BusinessReturnOtherHeaderState.xsd
    - BusinessStateEnumerations.xsd
  - xxCommon\ (common across xx business and individual schemas)
    - xxeFileTypes.xsd
    - FormXXX.xsd
    - FormZZZ.xsd
    - ScheduleB.xsd
  - xxBusiness\
    - xxForms\
      - FormNNN.xsd
      - FormYYY.xsd
      - ScheduleA.xsd
      - WksIII.xsd
    - ReturnDataxxNNN.xsd
    - ReturnDataxxYYY.xsd
    - ParentReturnDataxxNNN.xsd
    - SubsidiaryReturnDataxxNNN.xsd
    - BusinessReturnxxNNN.xsd
    - BusinessReturnxxYYY.xsd
    - ParentReturnxxNNN.xsd
    - SubsidiaryReturnxxNNN.xsd

7.3 How to Build a State Individual Package

7.3.1 Build State Form, Schedule, and Worksheet Schemas

The first step for the state is to build the schemas for the various forms, schedules, and worksheets that make up the state's individual income tax filing(s). A separate schema must be used for every form, and for every significant schedule or worksheet. As stated previously in this document, the root element of the schemas for all forms are to begin with "Form" followed by the name of the form; root elements of schedules begin with "Sch" followed by the name of the schedule, and root elements of worksheets begin with "Wks" followed by the name of the worksheet. If any other common prefixes are desired, they can be proposed to TIGERS for a vote. In general the schema file names (.xsd files) should equal the root elements. Naming conventions for elements are given in section 5.0 of this document.

As the state builds these form, schedule, and worksheet schemas, the state will probably encounter the need to create complex types that are used in more than one form, schedule, or worksheet. These shared complex types must be added to the state schema xxeFileTypes, where xx is the state abbreviation. The complex types can then be maintained in one place, and used wherever they are needed. This xxeFileTypes schema must then be included in each of the state form, schedule, or worksheet schemas needing any of these shared complex types.

The xxeFileTypes schema is stored within a folder named xxCommon, indicating components that are common to more than one usage. The schemas for any forms, schedules, or worksheets that are also shared across more than one type of filing, such as a schedule that is part of both the state's resident filing and its non-resident filing, should also be stored in the xxCommon folder. The remaining schemas for forms, schedules, and worksheets should be stored within a folder named xxForms, within a higher level folder called xxIndividual. The xxCommon and xxIndividual folders must be added to the TIGERS standard schema package as shown in section 7.2.1 above.

7.3.2 Build ReturnDataXX Schema(s)

The next step for the state is to determine how many separate types of Individual income tax filings the state will support through MeF efile. These programs are distinguished by different main forms, with potentially different requirements for additional forms, schedules, and/or worksheets. For example, a state may have a "long" form, a simpler "A" form, and a non-resident form. The "A" form may not allow schedules to be filed with it that can be filed with the "long" form.

The state then builds a ReturnData schema for each type of filing, giving the forms, schedules and worksheets that make up the filing. For example, South Carolina might have ReturnDataSC1040, ReturnDataSC1040A, and ReturnDataSC1040NR. **Each of these schemas still has the root element ReturnDataState, but the file name (.xsd) of the schema indicates the type of filing.**

In our example, ReturnDataSC1040 schema is stored as file ReturnDataSC1040.xsd, but still has root element ReturnDataState. Under that root element, South Carolina would include the schema FormSC1040.xsd, plus the schemas for all of the other forms, schedules, and worksheets that can be filed along with SC1040. All of these schemas are included by reference. For example, inside schema ReturnDataSC1040, there will be a statement 'xsd:include schemaLocation="../SCForms/FormSC1040.xsd"' to include the schema for the form SC1040, which is stored as FormSC1040.xsd inside the folder SCForms, inside the higher level folder SCIndividual. Also inside the ReturnDataSC1040 schema, element ReturnDataState will be turned into a complex type, and the first element of that complex type will be 'xsd:element ref="FormSC1040"' where FormSC1040 is the root element of schema stored as FormSC1040.xsd.

To show this more clearly:

```
<xsd:schema>
        <xsd:include schemaLocation="../SCForms/FormSC1040.xsd"/>
        <xsd:element name="ReturnDataState>
                <xsd:complexType>
                        <xsd:sequence>
                                <xsd:element ref="FormSC1040/>
                                <xsd:element ref=" …."/>
                        </xsd:sequence>
                </xsd:complexType>
        </xsd:element>
</xsd:schema>
```

This is an over-simplification of the schema, but it shows the general structure that is needed. The "…" refers to another form or schedule that is filed with the SC1040, and there may be as many of these elements as the state needs. This same process is followed for the all of the other filing types, such as SC1040A and SC1040NR.

The state now deletes the original stub schema ReturnDataState.xsd from the TIGERS StateIndividual folder. This schema was simply a placeholder to indicate that state ReturnDataxx schemas needed to be built; it no longer serves any useful purpose.

7.3.3 Build IndividualReturnXX Schema(s)

Finally, for each of the ReturnDataxx schemas, the state now must build the actual return filing schema IndividualReturnxx. To do that, the state must copy the IndividualReturnState schema down to the xxIndividual folder, renaming it to match one of the ReturnDataxx schemas. For example, the filing for South Carolina form SC1040, built as schema ReturnDataSC1040, will be contained in schema IndividualReturnSC1040.

The state must next replace the include statement in the renamed schema for the stub schema ReturnDataState with an include statement for the appropriate ReturnDataxx schema. In the example above, South Carolina will have copied the IndividualReturnState schema down to the SCIndividual folder, and renamed it IndividualReturnSC1040. Now the state will replace the statement <xsd:include schemaLocation="ReturnDataState.xsd" with the statement <xsd:include schemaLocation="ReturnDataSC1040.xsd" in the schema.
The element statement referencing the schema does **not** have to be changed, because the root element of the state ReturnDataxx schema – in our example, the root element of the schema ReturnDataSC1040 – is still called ReturnDataState.

This process must be repeated for each type of state filing, represented by each ReturnDataxx schema that the state has created. The state now deletes the original schema IndividualReturnState from the TIGERS StateIndividual folder. It has served as a pattern for the standard structure that must be used for each state specific IndividualReturnxx schema, and no longer serves any useful purpose.

7.3.4 Optional – Customize FinancialTransaction

While the full FinancialTransaction schema is quite powerful, and its use is encouraged by TIGERS, some states have expressed strong concerns with functionality that their agency does not support. States can utilize business rules within their application programs to reject returns containing payment options not supported by the state. However, a number of states want to eliminate these options in the schema, so that there is no chance of their being used by the software preparing the return. For example, if a state does not accept payment by ACH Credit for a particular program, the state may not want any possibility that software will support it.

After much discussion, the TIGERS consensus vote was to allow states to customize the FinancialTransaction schema, according to the following rules. This is the ONLY change that a state may make to the TIGERS Common schemas.

a. The state must start with the full FinancialTransaction schema. The schema has four major data elements: StatePayment, RefundDirectDeposit, ACHCreditInfo, and EstimatedPayments.

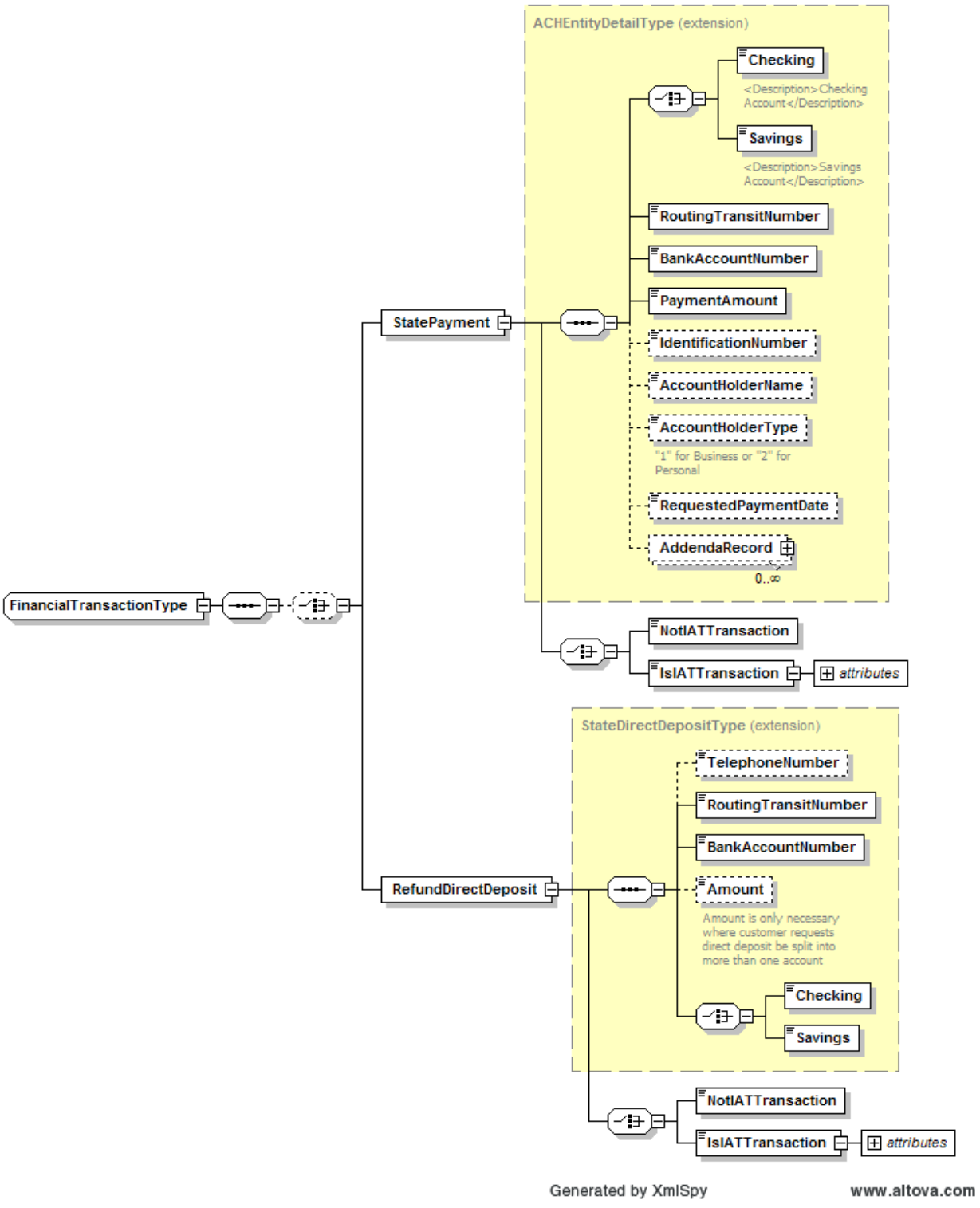
b. The state can pick and choose which of the four data elements to utilize, and which to delete. For example, a state may delete ACHCreditInfo in its entirety from the FinancialTransaction schema. However, the state may NOT alter any of the efileTypes that make up any of the four data elements.

c. Additionally, the state can determine how many occurrences to allow for each of the four elements. For example, a state can utilize RefundDirectDeposit, but limit it to a single occurrence for a single bank account.

d. If a state does not wish to incorporate either electronic payment or direct deposit of refunds into its MeF program, the state may delete the FinancialTransaction schema in its entirety from the IndividualReturnXX and/or Business ReturnXX schemas.

e. If a state does not support the IAT for financial transactions, then the state may delete the IsIATTransaction element, leaving the NotIATTransaction element to be checked as a jurat. In that case, the state must change the <xsd:choice> and </xsd:choice> operators to <xsd:sequence> and </xsd:sequence>. Additionally, the state may leave both choices, but eliminate the attribute for the taxpayer's bank, so that IsIATTransaction is put back to simple CheckboxType. Finally, the state may eliminate the entire extension, both IsIATTransaction and NotIATTransaction, if the state makes it clear that it is not accepting IATs. However, it must be noted that this will NOT prevent a taxpayer from requesting a financial transaction whose funds are coming from, or going to, a foreign account.

Below is a diagram of a customized FinancialTransaction schema, along with the schema code.



<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 rel. 3 sp2 (http://www.altova.com) by Terry Garber (SC DEPT OF REVENUE & TAXATION) -->

```
<xsd:schema xmlns="http://www.irs.gov/efile"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
TargetNamespace="http://www.irs.gov/efile" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xsd:annotation>
       <xsd:documentation>
            <Description>Data for tax payments or refund deposits</Description>
                <ReleaseDate>November 12, 2009</ReleaseDate>
      </xsd:documentation>
  </xsd:annotation>
  <xsd:include schemaLocation="efileTypes.xsd"/>
  <xsd:include schemaLocation="StateeFileTypes.xsd"/>
  <!-- Root Element -->
  <xsd:element name="FinancialTransaction" type="FinancialTransactionType">
       <xsd:annotation>
           <xsd:documentation>Root element for Financial Transaction</xsd:documentation>
      </xsd:annotation>
  </xsd:element>
  <!--Structure for Full Financial Transaction Schema-->
  <xsd:complexType name="FinancialTransactionType">
       <xsd:sequence>
            <xsd:choice minOccurs="0">
            <!-- Payment(s) -->
                 <xsd:element name="StatePayment">
                    <xsd:complexType>
                         <xsd:complexContent>
                              <xsd:extension base="ACHEntityDetailType">
                                   <xsd:choice>
                                        <xsd:element name="NotIATTransaction"
type="CheckboxType"/>
                                         <xsd:element name="IsIATTransaction">
                                             <xsd:complexType>
                                                 <xsd:simpleContent>
                                                     <xsd:extension base="CheckboxType">
                                                         <xsd:attribute name="ReceivingDFIName"
type="String50Type"/>
                                                     </xsd:extension>
                                                 </xsd:simpleContent>
                                             </xsd:complexType>
                                        </xsd:element>
                                   </xsd:choice>
                              </xsd:extension>
                         </xsd:complexContent>
                    </xsd:complexType>
                 </xsd:element>
           <!-- Refund(s) -->
```

```
                <xsd:element name="RefundDirectDeposit">
                    <xsd:complexType>
                        <xsd:complexContent>
                            <xsd:extension base="StateDirectDepositType">
                                <xsd:choice>
                                    <xsd:element name="NotIATTransaction"
type="CheckboxType"/>
                                    <xsd:element name="IsIATTransaction">
                                        <xsd:complexType>
                                            <xsd:simpleContent>
                                                <xsd:extension base="CheckboxType">
                                                    <xsd:attribute name="ReceivingDFIName"
type="String50Type"/>
                                                </xsd:extension>
                                            </xsd:simpleContent>
                                        </xsd:complexType>
                                    </xsd:element>
                                </xsd:choice>
                            </xsd:extension>
                        </xsd:complexContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:choice>
        </xsd:sequence>
    </xsd:complexType>
</xsd:schema>
```

NOTE: Generally, the above is all that is needed. However, if the state is not using the same customization of FinancialTransaction in all of its programs – for example, if the state wants to allow electronic payment with SC1040 but not with SC1040A – then all but the first customization must be given a different schema file (.xsd) name; there cannot be two schema files named FinancialTransaction.xsd. The second customized FinancialTransaction schema must be stored in XXCommon. In this case, the include statement in the IndividualReturnXX schema for the second FinancialTransaction version must be modified to point to the alternatively named FinancialTransaction schema file. An example of the modified IndividualReturnXX is shown below. Assume that the second version of FinancialTransaction is named FinTranDebitOrRefund.xsd. Note that the root element is still named “FinancialTransaction” – the include statement alone points to the second FinancialTransaction schema.

The IndividualReturnXX will use the new schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 rel. 3 sp2 (http://www.altova.com) by Terry Garber (SC DEPT OF REVENUE & TAXATION) -->
<xsd:schema xmlns="http://www.irs.gov/efile"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.irs.gov/efile" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="1.0">
	<xsd:annotation>
		<xsd:documentation>
			<Description>State e-file Individual Income Tax Schema</Description>
		</xsd:documentation>
	</xsd:annotation>
	<xsd:include schemaLocation="IndividualReturnHeaderState.xsd"/>
	<xsd:include schemaLocation="ReturnDataState.xsd"/>
	<xsd:include schemaLocation="../Common/BinaryAttachment.xsd"/>
	<xsd:include schemaLocation="../XXCommon/FinTranDebitOrRefund.xsd"/>
	<!-- Individual Return - 1040 -->
	<xsd:element name="ReturnState">
		<xsd:annotation>
			<xsd:documentation>1040 Individual Return - wraps around Return Header and Return Data</xsd:documentation>
		</xsd:annotation>
		<xsd:complexType>
			<xsd:sequence>
				<xsd:element ref="ReturnHeaderState"/>
				<!--ReturnDataState to be built by each state -->
				<xsd:element ref="ReturnDataState"/>
				<xsd:element ref="BinaryAttachment" minOccurs="0" maxOccurs="unbounded"/>
				<xsd:element ref="FinancialTransaction" minOccurs="0"/>
			</xsd:sequence>
			<xsd:attribute name="stateSchemaVersion" type="String50Type"/>
			<!-- Version of the state schema set used for the filing-->
		</xsd:complexType>
	</xsd:element>
</xsd:schema>
```

7.4 How to Build a State Business Package

7.4.1 Build State Form, Schedule, and Worksheet Schemas

The first step for the state is to build the schemas for the various forms, schedules, and worksheets that make up the state's corporate and/or partnership income tax filing(s). A separate schema must be used for every form, and for every significant schedule or worksheet. As stated previously in this document, the root element of the schemas for all forms are to begin with "Form" followed by the name of the form; root elements of schedules begin with "Sch" followed by the name of the schedule, and root elements of worksheets begin with "Wks" followed by the name of the worksheet. If any other common prefixes are desired, they can be proposed to TIGERS for a vote. In general the schema file names (.xsd files) should equal the root elements. Naming conventions for elements are given in section 5.0 of this document.

As the state builds these form, schedule, and worksheet schemas, the state will probably encounter the need to create complex types that are used in more than one form, schedule, or worksheet. These shared complex types must be added to the state schema xxeFileTypes, where xx is the state abbreviation. The complex types can then be maintained in one place, and used wherever they are needed. This xxeFileTypes schema must then be included in each of the state form, schedule, or worksheet schemas needing any of these shared complex types.

The xxeFileTypes schema is stored within a folder named xxCommon, indicating components that are common to more than one usage. The schemas for any forms, schedules, or worksheets that are also shared across more than one type of filing, such as a schedule that is part of both the state's C-corp filing and its S-corp or partnership filing, should also be stored in the xxCommon folder. The remaining schemas for forms, schedules, and worksheets should be stored within a folder named xxForms, within a higher level folder called xxBusiness. The xxCommon and xxBusiness folders must be added to the TIGERS standard schema package as shown in section 7.2.2 above.

7.4.2 Build ReturnDataXX Schema(s)

The next step for the state is to determine how many separate types of Business income tax filings the state will support through MeF efile. These programs are distinguished by different main forms, with potentially different requirements for additional forms, schedules, and/or worksheets. For example, a state may have a C-corp form, a simpler S-corp form, and a Partnership form. The S-Corp form may not allow schedules to be filed with it that can be filed with the C-corp form.

The state then builds a ReturnData schema for each type of filing, giving the forms, schedules and worksheets that make up the filing. For example, South Carolina might have ReturnDataSC1120, ReturnDataSC1120S, and ReturnDataSC1065. **Each of**

these schemas still has the root element ReturnDataState, but the file name (.xsd) of the schema indicates the type of filing.

In our example, ReturnDataSC1120 schema is stored as file ReturnDataSC1120.xsd, but still has root element ReturnDataState. Under that root element, South Carolina would include the schema FormSC1120.xsd, plus the schemas for all of the other forms, schedules, and worksheets that can be filed along with SC1120. All of these schemas are included by reference. For example, inside schema ReturnDataSC1120, there will be a statement 'xsd:include schemaLocation="../SCForms/FormSC1120.xsd"' to include the schema for the form SC1120, which is stored as FormSC1120.xsd inside the folder SCForms, inside the higher level folder SCIndividual. Also inside the ReturnDataSC1120 schema, element ReturnDataState will be turned into a complex type, and the first element of that complex type will be 'xsd:element ref="FormSC1120"' where FormSC1120 is the root element of schema stored as FormSC1120.xsd.

To show this more clearly:

```
<xsd:schema>
        <xsd:include schemaLocation="../SCForms/FormSC1120.xsd"/>
        <xsd:element name="ReturnDataState>
                <xsd:complexType>
                        <xsd:sequence>
                                <xsd:element ref="FormSC1120/>
                                <xsd:element ref=" …."/>
                        </xsd:sequence>
                </xsd:complexType>
        </xsd:element>
</xsd:schema>
```

This is an over-simplification of the schema, but it shows the general structure that is needed. The "…" refers to another form or schedule that is filed with the SC1120, and there may be as many of these elements as the state needs. This same process is followed for the all of the other filing types, such as SC1120S and SC1065.

Note that for a C-corp filing in a state with filings including a parent corporation and one or more subsidiary corporations, similar schemas must be created to replace the ParentReturnDataState and SubsidiaryReturnDataState schemas, again keeping the original root element names.

The state now deletes the original stub schema ReturnDataState.xsd from the TIGERS StateBusiness folder. This schema was simply a placeholder to indicate that state ReturnDataxx schemas needed to be built; it no longer serves any useful purpose. Similarly, the state deletes the original stub schemas ParentReturnDataState and SubsidiaryReturnDataState, whether or not the state created parent and subsidiary data schemas.

7.4.3 Build BusinessReturnXX Schema(s)

Finally, for each of the ReturnDataxx schemas, the state now must build the actual return filing schema BusinessReturnxx. To do that, the state must first determine the structure of the state filing – does the filing include sub-filings for a parent and one or more subsidiary corporations, like BusinessReturnStateCombined, or is the filing a simpler structure, line BusinessReturnStateSingle? The state must then copy the appropriate schema, BusinessReturnStateCombined or BusinessReturnStateSingle, down to the xxBusiness folder, renaming it to match one of the ReturnDataxx schemas. For example, the filing for South Carolina form SC1120, built as schema ReturnDataSC1120, will be contained in schema BusinessReturnSC1120, which will be built on a copy of TIGERS standard schema BusinessReturnStateCombined.

The state must next replace the include statement in the renamed schema for the stub schema ReturnDataState with an include statement for the appropriate ReturnDataxx schema. In the example above, South Carolina will have copied the BusinessReturnStateCombined schema down to the SCBusiness folder, and renamed it IndividualReturnSC1040. Now the state will replace the statement <xsd:include schemaLocation="ReturnDataState.xsd" with the statement <xsd:include schemaLocation="ReturnDataSC1120.xsd" in the schema.
The element statement referencing the schema does **not** have to be changed, because the root element of the state ReturnDataxx schema – in our example, the root element of the schema ReturnDataSC1120 – is still called ReturnDataState.

Similarly, if the state is using the BusinessReturnStateCombined schema as a model, the state must build ParentReturnxx and SubsidiaryReturnxx schemas. To do this, the include statements for ParentReturnDataState and SubsidiaryReturnDataState must be replaced with include statements for the appropriate state specific schemas. Then the include statements for ParentReturnState and SubsidiaryReturnState inside the BusinessReturnStateCombined schema must be replaced by include statements for the new ParentReturnxx and SubsidiaryReturnxx schemas.

This process must be repeated for each type of state filing, represented by each ReturnDataxx schema that the state has created.

The state then deletes the original BusinessReturnStateCombined, BusinessReturnStateSingle, ParentReturnState, and SubsidiaryReturnState schemas from the TIGERS standard StateBusiness folder. These schemas served as patterns to indicate the required standard structures for state specific combined, single, parent, and subsidiary schemas, and no longer serve any useful purpose.

7.4.4 Optional – Customize FinancialTransaction

Follow steps given in section 7.3.4, substituting BusinessReturnSingleXX or BusinessReturnCombinedXX instead of IndividualReturnXX.

8.0 Return Packaging

The state submission or filing under Modernized eFile is a zipped package. The following diagram shows the structure of the unzipped package

.

- Manifest\
  - Manifest.xml (mainifest formatted to IRS standards)
- XML\
  - Submission.xml ("Submission" is replaced by the unique state submission id)
- Attachment\
  - Attachment1.pdf (pdf files of binary attachments)
  - Attachment2.pdf (etc)
- OtherState\ (optional return from another state, as in part-resident filiing)
  - XML\
    - ST.xml ("ST" is replaced by the standard state abbreviation)
  - Attachment\
    - Attachment1.pdf (pdf files of binary attachments)
    - Attachment2.pdf (etc)
- IRS\
  - XML\
    - Submission.xml ("Submission" is replaced by the unique state submission id)
  - Attachment\
    - Attachment1.pdf (pdf files of binary attachments)
    - Attachment2.pdf (etc)
- IRS1\
  - XML\
    - Submission.xml ("Submission" is replaced by the unique state submission id)
  - Attachment\
    - Attachment1.pdf (pdf files of binary attachments)
    - Attachment2.pdf (etc)

The OtherState folder, which is optional, is used to contain a copy of a return filed to another state, using that state's schemas. This would be used, for example, if the state filing is a part-year return and the state requires a copy of the other state's return. If there are multiple other states, the folders can be named OtherState1, OtherState2, etc. **Note that the use of this folder cannot be mandatory, because the preparer of the state return may not have access to the other state's return, which may have been prepared by someone else in the other state.**

The IRS folder contains the state copy, provided by the taxpayer, of the taxpayer's federal filing. The IRS1 folder is optional, and is used **only** where two separate and distinct IRS filings are to be provided

- For business filings, this is generally a copy of a corporation's consolidated filing and a copy of a state pro forma, which also uses the IRS schema set.
- For individual filings, this is generally a copy of the primary taxpayer's return and a copy of the secondary taxpayer's return, if the couple filed separately with the IRS.
- In very rare cases, an IRS2 folder may be needed for a joint pro forma filing, if the couple filed separately with the IRS.