Federal Energy Regulatory Commission

DRAFT

XBRL FORMS REFRESH FERC FORMS TAXONOMY GUIDE 2020 v0.1 RELEASE

Office of the Executive Director

Chief Information Officer

December 2019

Table of Contents

Tables

Figures

Version History

Document Control

This is a controlled document produced by the Federal Energy Regulatory Commission (FERC). The control and release of this document is the responsibility of the document owner. This includes any amendment that may be required.

1 Introduction

The purpose of this document is to detail the architecture of the FERC Taxonomy. The document explains the design rationale and how the taxonomy architecture satisfies stakeholder requirements. The intended audience of this document is for a technical user familiar with XBRL, other specifications and modules of XBRL. It is not intended as a tutorial or user guide. Business users may be interested in this document and it is written such that a business user familiar with the XBRL standards will be comfortable with this document. The goal of this document is to define logical models, physical models and design rules that satisfy the FERC Reporting Requirements.

1.1 Basis for the Taxonomy

The taxonomy is based on the FERC Uniform System of Accounts and the data requirements specified in the FERC data collection forms; specifically Forms 1, 2, 6, 60 and 714.

The FERC Taxonomy Architecture follows rules defined by XBRL International specifications and best practices established by XBRL US domain steering committee. Adherence to these recommendations ensures that taxonomies using the Architecture are fully XBRL 2.1 compliant. These specifications include XBRL 2.1, XBRL Dimensions 1.0. The Financial Reporting Taxonomy Architecture Recommendations were also largely followed insofar as they did not conflict with the requirements of XBRL Dimensions 1.0. The architecture follows guidance defined in the XBRL US Taxonomy development handbook.

1.2 Physical Taxonomy Structure

The physical taxonomy structure details the file structure of the taxonomy. The taxonomy is structured to allow reporting by individual form and by individual schedule.

This structure is used to support:

1. Expansion of the taxonomy over time
2. Creation of new forms
3. Reuse of schedules within different forms
4. Reduction of duplication across forms
5. Simplification of taxonomy publication

The physical structure separates those components that are used in every form from those that are form specific. For example, the taxonomy contains a concept for “assets” that is shared among multiple forms (i.e. Form 1 and Form 2) but has separate form location (GPS locations) references for each form in which it appears. Only the references associated with a particular form, such as Form 1, will be included in the Form 1 entry point. The definition of the “assets” concept is global, but the form locations references are only in the appropriate form entry point.

1.3 Logical Model

The logical model details the data and meta-data in the taxonomy. The logical model focuses on detailing the groups of concepts and relationships needed to capture the data collection requirements determined by the Commission. Key principles implemented at the level of the logical model include:

- **Single Taxonomy with multiple Entry Points**
  The FERC taxonomy is a single taxonomy that allows users to select different entry points depending on the form that is being filed.

- **Extensions**
  The taxonomy is designed to eliminate the need for extension taxonomies. The FERC implementation will not permit filers to define extension elements or relationships. However, the FERC forms do allow filers flexibility to report company specific data. To enable the capture of this data, the taxonomy uses typed dimensions for repeating forms and footnotes for companies to provide additional information. The information in footnotes is not intended to be structured.

- **Rendering Templates**
  Because the data collected by FERC has been historically forms based, the taxonomy structure reflects the structure of the Forms. The taxonomy includes html templates for each schedule that can be processed with an XBRL instance file and taxonomy entry point to render a representation of the form. The templates will render an XHTML version of the instance in an inline XBRL format. The rendering templates are defined as labels in the taxonomy and are defined as labels of the Schedule abstract element that is at the presentation root of every schedule.

- **Validations**
  The taxonomy includes validations to ensure the quality of data submitted. The validations are grouped by form and schedule, with sets of validations that are applicable to a given form. Validations defined for one schedule will also run on other schedules where the data is the same. All validations are expressed using the XULE syntax.

1.4 Taxonomy Profile

The taxonomy and associated instances use the following components of the XBRL specification:

- Typed Dimensions
- Explicit Dimensions (Segment)
- Extensible Enumeration 1.0
- Data Type Registry 1.x
- Units Registry 1.0
- Inline XBRL1.1

- Generic Links 1.0

The following components of the XBRL specification are not included as part of the taxonomy architecture:

- Tuples (part of the core XBRL specification)
- XBRL Table Linkbase
- XBRL Formula Linkbase
- XBRL Versioning

2 Taxonomy Physical File Structure

The taxonomy is a single taxonomy that comprises all the FERC Forms in a single taxonomy. The physical files comprising the taxonomy are broken into three components that are represented in folders. These folders are Elements (Elts), Schedules (Schds) and Forms. This structure is used to support:

1. Expansion of the taxonomy over time
2. Creation of new forms
3. Reuse of schedules within different forms
4. Reduction of duplication across forms
5. Simplification of taxonomy publication.

2.1 Forms Directory

The forms directory contains entry points for each form and the associated form specific meta-data. The forms entry point defines how different schedules should be collated to define the form. A form entry point imports all the schedules that comprise the form. This entry point contains the presentation linkbase that defines the components of the Form. Any information that is form specific will be included in this directory. This includes the labels that are specific to the form such as the page numbers, instructions and Form (GPS) locations used by the form.

2.2 Schedules (Schds) Directory

The schedules directory has an entry point for every distinct schedule. This allows a schedule to be individually referenced as its own entry point. A schedule could be shared across multiple forms and can stand alone as its own taxonomy. This makes it possible to use the same schedule across forms without the need to duplicate the schedule for a different form. The schedules contain the presentations, calculations and definition linkbases associated with the form. The schedule also contains the rendering template associated with the schedule. The schedule is not associated with a Form at this level, and thus can be pulled in by any form.

The schedule also contains a link to the template that renders the schedule.

Each schedule includes a presentation, calculation and definition linkbase. In those cases where a typed dimension is used the default definition linkbase is also used. (See later) In the schedules folder there are no reference or label linkbases used as these will differ depending on which form the schedule is used.

2.3 Elements (ELTS) Directory

This folder contains the core elements used in the taxonomy. It contains the standard labels including standard definitions and the account references to the uniform system of accounts. It also includes the definition of specific label roles, the definition of reference parts and the definition of FERC-specific data types.

2.3.1 Data Types

FERC specific data types are defined in the taxonomy. All FERC-specific data types are defined in the file:

```
ferc-core-types_{date of release}.xsd
```

This file is in the element's directory of the taxonomy. In addition to the FERC-defined data types the taxonomy also uses data type defined in the XBRL data types registry and the XBRL specification. The following is a list of all the possible data types supported by the taxonomy.

2.3.1.1 Data types used in the Taxonomy

Every element defined in the FERC taxonomy has a specific data type. The data types used and supported in the taxonomy include the following.

Table 2.1 Data Types

Data Type	Description
ferc-types:formItemType	This is a FERC-specific data type. This type is used to define an abstract element that represents the actual FERC form. This element is used to hold the label for the Form 1, 2 etc.
ferc-types:scheduleItemType	This is a FERC-specific data type. This type is used to define an abstract element that is the root node of a schedule. Only elements with this type can have links to the form instructions, templates and page number of the schedule.
ferc-types:statesOfUnitedStatesItemType	An enumerated data type that can contain a 2 character state abbreviation.
ferc-types:fileItemType	This is a FERC-specific data type. This type is used to record the name of files that are included with the form submission. Examples would include the name of a system map for Form 2.
nonnum:domainItemType	This is a standard XBRL data type that is used to define domains.
nonnum:textBlockItemType	This is a standard XBRL data type that is used to define text block items. These items can contain any text including xhtml.
num-us:electricCurrentItemType	This is a standard XBRL US data type that is used to define

Data Type	Description
	measures of electric current.
num-us:frequencyItemType	This is a standard XBRL US data type that is used to define measures of frequency such as hertz.
num-us:insolationItemType	This is a standard XBRL US data type that is used to define measures of insolation.
num-us:irradianceItemType	This is a standard XBRL US data type that is used to define measures of irradiance.
num-us:planeAngleItemType	This is a standard XBRL US data type that is used to define measures of angles such as degrees or radians.
num-us:pressureItemType	This is a standard XBRL US data type that is used to define measures of pressure such as psi or bar.
num-us:speedItemType	This is a standard XBRL US data type that is used to define measures of speed such as mph.
num-us:temperatureItemType	This is a standard XBRL US data type that is used to define measures of temperature such as kelvin, Fahrenheit or Celsius.
num-us:voltageItemType	This is a standard XBRL US data type that is used to define measures of voltage.
num:areaItemType	This is a standard XBRL data type that is used to define measures of area such as acres, hectares, square feet etc.
num:energyItemType	This is a standard XBRL data type that is used to define measures of energy such as kilowatts per hour.
num:lengthItemType	This is a standard XBRL data type that is used to define measures of length such as miles, meters or inches etc.
num:massItemType	This is a standard XBRL data type that is used to define measures of mass such as kilos.
num:percentItemType	This is a standard XBRL data type that is used to define percentage items.
num:powerItemType	This is a standard XBRL data type that is used to define measures of power such as kilowatts.
num:volumeItemType	This is a standard XBRL data type that is used to define measures of power such as cubic feet.
num:perShareItemType	This is a standard XBRL data type that is used to define per share item types such as price per share.

Data Type	Description
srt-types:perUnitItemType	This is a standard XBRL data type that is used to define any item type over another unit such as USD. For example, the monthly rental per square foot in USD.
xbrli:sharesItemType	This is a standard XBRL data type that is used to define share item types such as ordinary shares issued by the company.
xbrli:anyURIItemType	This is a standard XBRL data type that is used to define uri item types such as website addresses.
xbrli:booleanItemType	This is a standard XBRL data type that is used to define Boolean item types with a value of true or false.
xbrli:dateItemType	This is a standard XBRL data type that is used to define date item types such as the date of the report.
xbrli:decimalItemType	This is a standard XBRL data type that is used to define decimal item types.
xbrli:durationItemType	This is a standard XBRL data type that is used to define duration item types. Duration item types express a period of time in the XML format P1Y1M.
xbrli:integerItemType	This is a standard XBRL data type that is used to define integer item types.
xbrli:monetaryItemType	This is a standard XBRL data type that is used to define monetary item types, such as an amount measured in USD.
xbrli:normalizedStringItemType	This is a standard XBRL data type that is used to define normalized string item types.
xbrli:pureItemType	This is a standard XBRL data type that is used to define pure item types. These types have no units, such as an exchange rate.
xbrli:stringItemType	This is a standard XBRL data type that is used to define string item types.

2.3.2 Footnotes

FERC filers will need to include footnotes as part of the filing program. These FERC filings will use standard XBRL footnotes roles as defined in the XBRL specification to allow filers to footnote any XBRL fact.

In addition to the standard footnote arc role, a special footnote arc role is used to indicate which facts are identified as confidential. This allows marking of confidential data on a fact by fact basis. The special footnote arc role is:

```
http://www.ferc.gov/arcrole/Confidential
```

This special footnote arc role is defined in file named:

```
ferc-core-footnote-roles_{Date of release}.xsd
```

Further details on using this special footnote arc role to identify confidential data is discussed in section 3.6 Confidential Data.

2.3.3 References

All elements in the taxonomy include references. References are primarily used to indicate the location of an element on a historical form. This provides a guide to filers and users of the taxonomy to link historical data to data filed in an XBRL format. It also helps in the transition of data and helping users understand what element represents an historical row and column number.

The XBRL specification comes with several standard reference types. The FERC taxonomy adds another reference role type called “Form Location”. This has an identifier of:

```
http://ferc.gov/form/2020-01-01/roles/reference/formLocation
```

References of this type create a relationship between the XBRL element and where the data is located relative to the location of the form, the schedule it appeared in and the row and column in which it appears. This is often referred to as the GPS location of the data.

In addition, references are included to identify the elements relationship to the uniform system of accounts used by the FERC. The FERC taxonomy includes a reference role type called “Account”. This has an identifier of:

```
http://ferc.gov/form/2020-01-01/roles/reference/account
```

Each of these reference roles are included in a file called:

```
ferc-core-ref-roles_2020-01-01.xsd
```

The name of this file will **not** generally change with each release of the taxonomy as this file is not expected to change.

Each reference contains a number of parts. All the parts for the FERC taxonomy are defined in the file called:

```
ferc-core-ref-parts_{Date of release if changed}.xsd
```

2.3.3.1 Form Parts and Location

The reference parts define the details of the form location and the uniform system of accounts. The form location role can contain the following parts:

Table 2.2 Form Parts

Part Name	Description
Form	Name of the form for the applicable reference.
Schedule	Name of the schedule for the applicable reference
Row	Identifier of the row associated with the element of the schedule.
Row Start	The starting row for elements that repeat on a schedule.
Row End	The ending row for elements that repeat on a schedule.
Column	The column identifier associated with an element. A single element can have multiple columns associated with it.
Period	The period part reference records the periodicity of the data that has been used for the element in FERC forms. Possible values include the following: Current, Prior, Prior2, Current 3M, Prior 3M, Month1, Month2, Month3, Quarter1, Quarter2, Quarter3, 3D

Dimension	The dimension part reference records the dimensions associated with the data that has been used for this element in the FERC forms. The format of the value for this part can vary if the dimension is explicit or typed. For an explicit dimension the format is as follows:
	Axis qname = Member qname
	An example of this is:
	`ferc:UtilityTypeAxis=ferc:ElectricityUtilityMember`
	
	For a typed dimension only the qname of the axis is required. An example of this is:
	
	`ferc:OfficerAxis`

The following is a label reference for Accretion Expense:

Form:	Form2
Schedule:	114 - Schedule - Statement of Income
Column:	c
Row:	24
Period:	Current

This reference indicates that the current value of accretion expense can be found on Row 24 in column C of Schedule 114 on the Form 2.

The value for this same item for the electricity utility is referenced as follows:

Form:	Form2
Schedule:	114 - Schedule - Statement of Income
Column:	g
Row:	24
Period:	Current
Dimension:	ferc:UtilityTypeAxis=ferc:ElectricityUtilityMember

Note that the dimension attribute is provided to distinguish that column g references data reported for Electric Utilities whereas column c references data for all utility types.

2.3.3.2 Account

Account references contain the following parts:

Table 2.3 Accounts Parts

Part Name	Description

USoA	Name of the Uniform System of Accounts. Current names are: Electric, Gas, Oil, Centralized Service
Account	Account number

For the element accretion expense, the account reference is:

USoA:	Gas
Account:	411.10

2.3.4 Label Roles

The label roles define the different types of labels that can be used in the taxonomy. These label roles can be associated with different presentation links so that different labels can appear on an element depending on what schedule it appears. Additionally, label roles are used to differentiate labels used for different location within a schedule, such as opening balance and closing balance labels. The list of all the possible label role types that are specific to the FERC are defined in the following file:

```
ferc-core-lab-roles_{Date of release}.xsd
```

2.4 Physical Taxonomy Directory

The following diagram shows the high-level physical structure of the taxonomy:

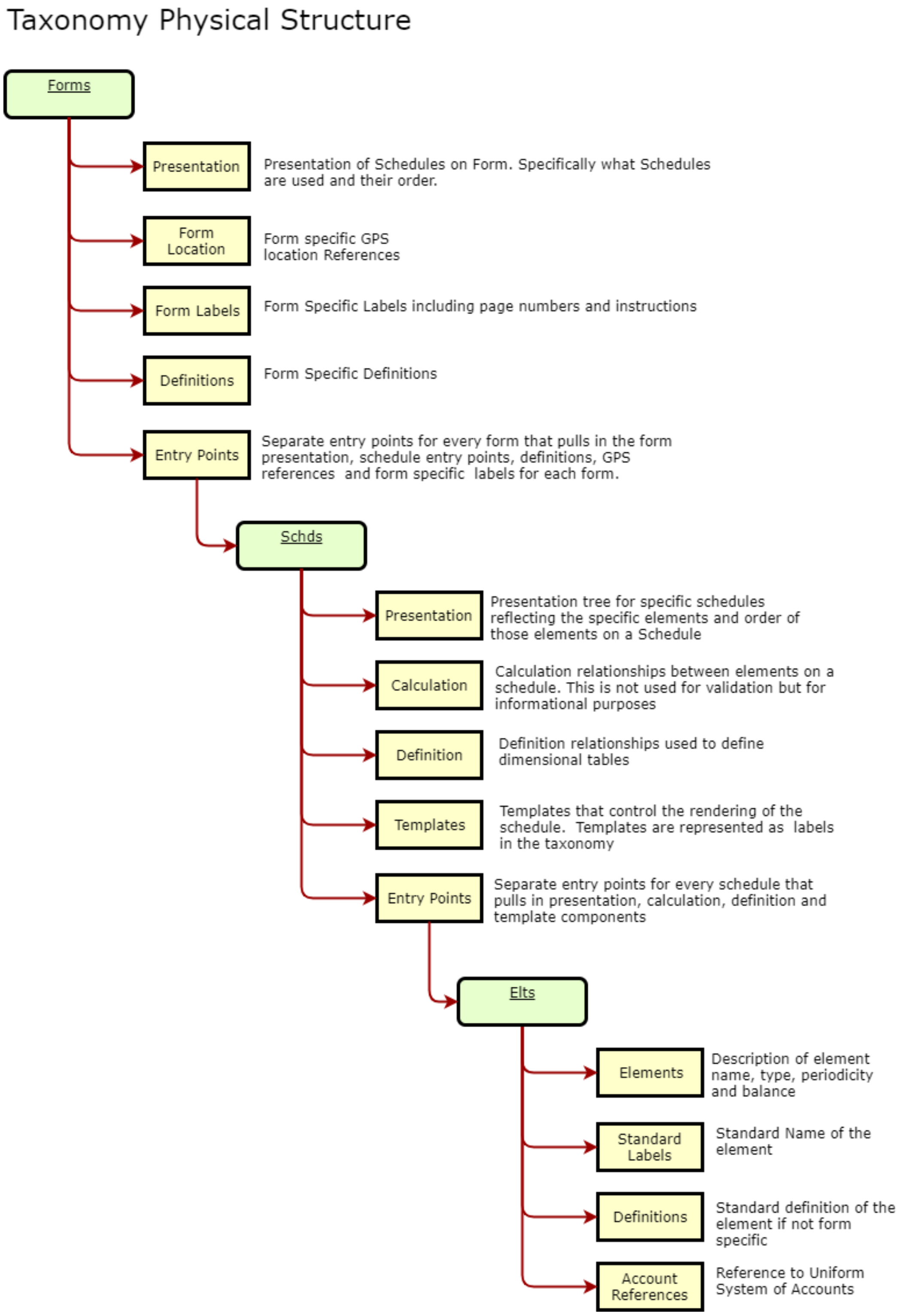

Taxonomy Physical Structure
Forms
Presentation
Presentation of Schedules on Form. Specifically what Schedules are used and their order.
Form Location
Form specific GPS location References
Form Labels
Form Specific Labels including page numbers and instructions
Definitions
Form Specific Definitions
Entry Points
Separate entry points for every form that pulls in the form presentation, schedule entry points, definitions, GPS references and form specific labels for each form.
Schds
Presentation
Presentation tree for specific schedules reflecting the specific elements and order of those elements on a Schedule
Calculation
Calculation relationships between elements on a schedule. This is not used for validation but for informational purposes
Definition
Definition relationships used to define dimensional tables
Templates
Templates that control the rendering of the schedule. Templates are represented as labels in the taxonomy
Entry Points
Separate entry points for every schedule that pulls in presentation, calculation, definition and template components
Elts
Elements
Description of element name, type, periodicity and balance
Standard Labels
Standard Name of the element
Definitions
Standard definition of the element if not form specific
Account References
Reference to Uniform System of Accounts

Figure 2-1 Taxonomy Physical Structure

The forms directory is comprised of sub directories for each form used by FERC. The form sub directories contain all the meta data for each form as well as the validation and template rulesets applicable for each form.

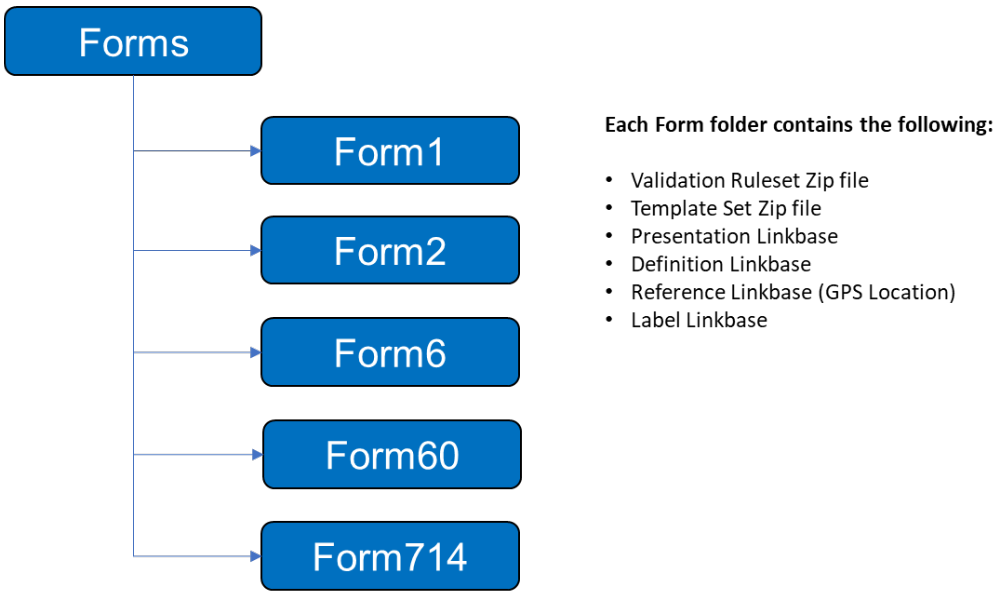


Figure 2-2 Forms Directory

2.4.1 Validation Rulesets

Each form when filed is validated using a validation ruleset that is applicable for that specific form. The validation ruleset is a compiled version of the rules associated with each form and is stored as a zip file. The validation ruleset is used to validate the form. Alternatively, the source XULE text defined in the label linkbase can also be run against the form. The validation ruleset is provided as a convenience that allows any data file to be validated.

2.4.2 Template Sets

In addition to the validation ruleset is a template set that compiles together all the template schedules for a specific form and can be used to render that specific form. The template set renders each schedule in the order defined in the template set. The template set is stored as a zip file in the specific form folder.

3 Taxonomy Logical Structure

The goal of the FERC taxonomy is to separate the presentational aspects of the form from the underlying data model. The taxonomy design incorporates the following principles:

1. The data from one form to the next must be consistent for the same disclosure.
2. The data across different forms for the same company should be consistent.
3. The same data should **not** be duplicated across different schedules in the same submission.
4. An element is only defined once even though it may appear in multiple schedules and forms.
5. An element will always have the same value wherever it appears in a schedule.
6. Filers will not be able to define extension taxonomies and can only use the base taxonomy.
7. The filed instance must be able to capture all the data currently reported in FERC forms.
8. The taxonomy does not require additional disclosures beyond those defined in the forms.
9. Rendered forms should be consistent with the format of published forms, or improve upon the historical form render.
10. Filers can validate their filings before they file.
11. Filers can render the XBRL instance as a form before they file.
12. Filers can control the order of disclosures on free form tables.
13. The FERC taxonomy is a single taxonomy with multiple entry points.
14. All elements have a standard label and definition.

FERC forms have several free form tables that allow filers to report data in a semi structured format. This allows filers to report company specific definitions and data in sequence tables. Additional data could also be provided as a footnote. The taxonomy is structured to support these free form tables without the need for extensions by using tightly defined typed dimensions. In this document these are referred to as sequence tables.

This approach means that the filer has significant flexibility on how they report the data on the XBRL instance. To restrict this flexibility and to ensure consistency and data quality across filings, the taxonomy includes a large number of validations to constrain this flexibility. These validations incorporate those already used but add additional validations to support filing in an XBRL format.

3.1 Concept Namespaces

The FERC taxonomy uses one namespace for all concepts. The format of the namespace is as follows:

```
http://ferc.gov/form/{release-date}/ferc
```

With each release of the taxonomy the release date is changed so element namespaces will be updated with a new namespace.

The namespace prefix used in the taxonomy is “ferc”. This is constantly applied throughout the entire taxonomy.

3.2 Extended Link Roles

The FERC forms taxonomy has different roles for every schedule defined in the forms. Each of these roles has the following format:

```
http://ferc.gov/form/{release-date}/roles/schedule/{schedule name (no space)}
```

The schedule name defined is unique within the taxonomy.

All link roles have a standard definition that follows the convention:

```
{schedule number} - Schedule - {Schedule Name}
```

For example, the schedule on page 106 for formula rates is defined as:

106 - Schedule - Information on Formula Rates

The schedule number used generally corresponds to the page number of the schedule in the form. If a schedule is used across multiple forms the page number may be different. In these cases, a different definition is used using a generic link to redefine the name for that form.

In some cases, a single schedule will contain multiple extended link roles. In these cases, the multiple extended link roles will be packaged in a one set of files associated with the schedule , but the extended link role description will be different for each role, by adding a suffix to distinguish the role. For example:

- 120 – Schedule – Statement of Cash Flows
- 120 – Schedule – Statement of Cash Flows – Cash Provided by Outside Sources
- 120 – Schedule – Statement of Cash Flows – Other Cash Outflows for Plant

3.3 Concepts

A large number of concepts exist within the FERC taxonomy. Each taxonomy concept is uniquely identifiable with labels, definitions, a data type and/or references to the financial reporting literature issued by the FERC or other standards setters and regulators.

Rules governing the concept, its label, the element name representing the concept, and its supporting references are documented in detail in the XBRL US Taxonomy implementation Style Guide.

3.3.1 Labels

Each financial reporting concept has a standard label and a definition label. In addition, the concept includes schedule specific labels that corresponds to the name of the concept as it appeared on a FERC form. All labels are provided in US values.

3.3.2 References

Each concept is referenced to the uniform system of accounts if applicable and referenced to its location on a schedule. A single concept can have multiple form (GPS) locations associated with it especially if that element appears in multiple schedules and multiple forms. The structure of the references is discussed in section 2.3.3 References.

3.3.3 Data Types & Units

The taxonomy uses built in XBRL data types as well as additional FERC-defined data types. These data types are expected to be used with the units defined in the XBRL International unit's registry, or any additional units permitted by the FERC. The concept definitions will indicate what is the unit that should be used to report a value.

3.3.4 Nillable

All concepts in the taxonomy allow nillable values to be entered. Nillable values are allowed for those cases when a filer provides confidential data. In these cases, the value in the redacted instance will be nil. In addition, a filer can report a nillable value, where appropriate.

3.3.5 Schedule Data type

The taxonomy defines a data type called scheduleItemType. This is the root node of each schedule. Only one element in a schedule link role can use this data type. The data type is an anchor for the details of the schedule. Only elements with this data type can have the following label types:

- Schedule Page Number
- Instructions
- Rendering Template

This element is always an abstract type and should always be the root node of a schedule.

The rendering template for each schedule is documented as a label of the Schedule Abstract element with a type of ferc-types:scheduleItemType. The label type used is:

```
http://ferc.gov/form/2020-01-01/roles/label/Template
```

Templates describe how the XBRL data can be rendered in an xhtml format using the instance document, taxonomy and template. The structure of templates is described in section 5.3 The Form Template.

3.3.6 Form Data type

The taxonomy defines a data type called formItemtype. Elements with this data type represent the root of a form. The elements using this data type indicate the details of the form and should only appear in the List of Schedules presentation tree. This element can only have descendants that are scheduleItemType elements. The listing of schedules presentation tree is also used to determine the order in which schedules are rendered.

The form data type has the instructions of the schedule listing but also includes all the cover page information of the form in an html format. This means that that the majority of a forms content is defined in the taxonomy rather than in a form template.

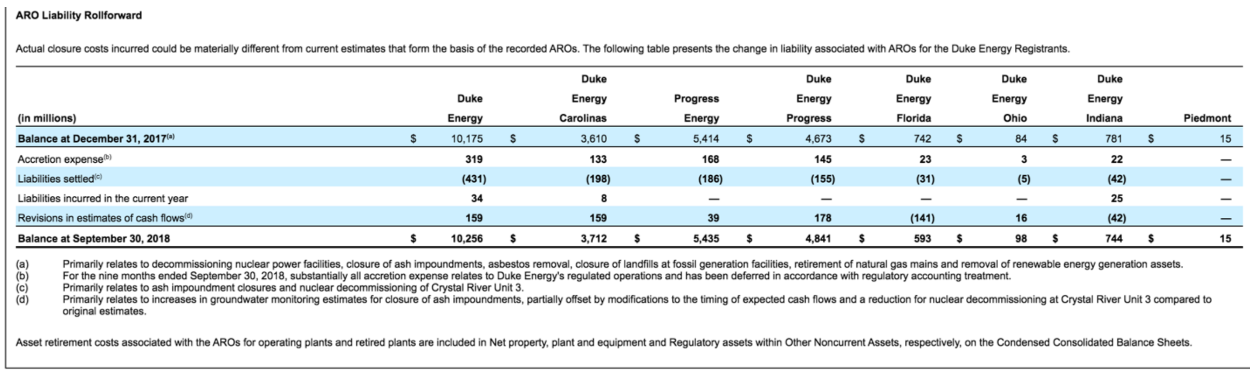
3.3.7 File Data type

In some schedules the filer is required to file supporting documentation. Rather than incorporating a binary document into the XBRL filing, the filer can submit this information as a separate file. To associate the file with the XBRL filing, the filing name is provided. The file name is entered as a text string using the fileItemType element. When a filing is submitted to the FERC submission portal the files associated with these element types will need to be uploaded.

3.3.8 Text Block Data type

The taxonomy includes the textBlockItemType. This data type is used so that the filer can copy large text blocks in an xhtml format into a template. It is expected that these disclosures will include formatting in the data reported. The rendering templates have been structured so that this formatting will not be impacted by the templates cascading style sheets.

The textBlockItemType is defined in the non numeric data types schema maintained by XBRL International. This data type is registered in the Data Type Registry also maintained by XBRL International. The diagram below shows xhtml data formatted in a text block item type.



ARO Liability Rollforward

Actual closure costs incurred could be materially different from current estimates that form the basis of the recorded AROs. The following table presents the change in liability associated with AROs for the Duke Energy Registrants.

(in millions)	Duke Energy	Duke Energy Carolinas	Progress Energy	Duke Energy Progress	Duke Energy Florida	Duke Energy Ohio	Duke Energy Indiana	Piedmont
Balance at December 31, 2017[a]	$ 10,175	$ 3,610	$ 5,414	$ 4,673	$ 742	$ 84	$ 781	$ 15
Accretion expense[b]	**319**	**133**	**168**	**145**	**23**	**3**	**22**	—
Liabilities settled[c]	**(431)**	**(198)**	**(186)**	**(155)**	**(31)**	**(5)**	**(42)**	—
Liabilities incurred in the current year	**34**	**8**	—	—	—	—	**25**	—
Revisions in estimates of cash flows[d]	**159**	**159**	**39**	**178**	**(141)**	**16**	**(42)**	—
Balance at September 30, 2018	**$ 10,256**	**$ 3,712**	**$ 5,435**	**$ 4,841**	**$ 593**	**$ 98**	**$ 744**	**$ 15**

(a) Primarily relates to decommissioning nuclear power facilities, closure of ash impoundments, asbestos removal, closure of landfills at fossil generation facilities, retirement of natural gas mains and removal of renewable energy generation assets.
(b) For the nine months ended September 30, 2018, substantially all accretion expense relates to Duke Energy's regulated operations and has been deferred in accordance with regulatory accounting treatment.
(c) Primarily relates to ash impoundment closures and nuclear decommissioning of Crystal River Unit 3.
(d) Primarily relates to increases in groundwater monitoring estimates for closure of ash impoundments, partially offset by modifications to the timing of expected cash flows and a reduction for nuclear decommissioning at Crystal River Unit 3 compared to original estimates.

Asset retirement costs associated with the AROs for operating plants and retired plants are included in Net property, plant and equipment and Regulatory assets within Other Noncurrent Assets, respectively, on the Condensed Consolidated Balance Sheets.

FERC FORM NO. 2 (12-96)

Page 108.1

3.3.9 Domain Data type

The taxonomy uses the domainItemType. This data type is used to identify dimension the dimension domain and member concepts. The domainItemType is defined in the non numeric data types schema and registered in the XBRL Data Type Registry (maintained by XBRL international).

3.4 Tables and Dimensions

The FERC taxonomy uses taxonomy defined dimensions including typed dimensions and explicit dimensions to represent the FERC data. All data defined in the FERC taxonomy is included in a table. In addition to regular tables a default table is defined to allow default values to be reported when that same element is used without a typed dimension.

3.4.1 Sequenced Schedules

Many FERC schedules allow the filer to define data in blank tables. The form instructions detail the disclosures required, but because the disclosure is filer specific, the filer defines the descriptions and associated values in an order defined by the filer. Because the FERC filing program does not allow extension taxonomies, the order of items in FERC forms has to be controlled by data submitted in the instance document. A simple example of such a disclosure may be a listing of the directors of the company. The filer historically listed them in the order they deemed appropriate. The XBRL implementation of the directors table allows the same data to be represented in the same order.

The order could be dictated by providing a sequence for the type dimension and ordering the items based on the typed dimension value. This approach however violates one of the principles of the logical model: The presentation requirements of a fact should not define the aspects of that fact. By using the typed dimension to dictate presentation order means that the same director may be defined with a different typed dimension member from filing to filing. The details of a given director such as their name is the same fact and the typed dimension of the fact should remain constant through time.

The presentational data used for sorting needs to be defined in a manner that continues to ensure the underlying data is consistent between filings of a single filer.

When a company uses a typed dimension, it is used with a typed member that has an alphanumeric key[1]. This key should be unique and consistent between filings. The taxonomy has been defined so that each repeating schedule has a unique typed dimension. The values of the alphanumeric key used for the typed dimension must have a specific meaning. These typed dimensions should avoid using meaningless integers unless the filing company has steps in place to ensure they are consistent from period to period.

3.4.1.1 Ordering Items

To render sequenced data in the correct order on the form the filer can control the ordering of items on the form by associating the alphanumeric key of the typed dimension with a specific order number.

To achieve this a line item called "OrderNumber" is used to record an order number. An OrderNumber is defined for every sequenced table in the taxonomy. Each typed member is associated with a specific order number that controls the order by which specific rows will appear on the form when rendered. For example, in the directors schedule the table would be as follows:

DirectorNameAxis = DirectorNameMember	**Order Number**	**Director Name**	**Salary**
JoeSmith	1	Joe Smith	150,000
AnneBoulder	2	Anne Boulder	200,000
LeeKing	3	Lee King	300,000

There is a presentation table defined in the taxonomy for each schedule that allows the filer to define their own row numbers. The presentation table is always comprised of the following:

1. Typed Axis applicable to disaggregate the schedule
2. Line item called OrderNumber.

To report the salary of the directors the filer would report the following (each row represents a separate reported fact):

DirectorNameAxis = DirectorNameMember	**Line Item**	**Value**
JoeSmith	DirectorName	Joe Smith
JoeSmith	Salary	150,000
JoeSmith	OrderNumber	1

[1] The alphanumeric key is a typed dimension value and is defined by the filer in the instance document. These values must be unique and be consistent between filings if representing the same data.

AnneBoulder	DirectorName	Anne Boulder
AnneBoulder	Salary	200,000
AnneBoulder	OrderNumber	2
LeeKing	DirectorName	Lee King
LeeKing	Salary	300,000
LeeKing	OrderNumber	3

When rendered the details of Joe Smith's name and salary would appear first on the director schedule before Anne Boulder or Lee Kink.

If an order number is not specified, the row number used will be ordered alphabetically based on the member name. On the example above if an order number is not provided then then Anne Boulder will appear first.

Order numbers are defined as a decimalItemType and can also have negative values. The use of decimals allows new rows to be inserted between existing rows. The order number can be the same as the row number, but they are not required or expected to be the same. Data reported without an order number will be ordered as it is processed by the rendering software, which is not predictable. Order numbers defined in an instance must be unique for a given schedule. This is validated as part of the form validation checks.

3.4.1.2 Row Numbers

The row numbers on sequence tables are automatically generated as an incremental sequence starting at 1. The row numbers are not recorded in an XBRL instance and are not defined in the FERC taxonomies. This is discussed in detail in section 5 Form Rendering.

3.4.2 Typed Dimensions

The FERC taxonomy uses typed dimensions throughout to capture repeating data. Typed dimensions do not allow the reporting of default values within a table with a typed dimension. This is generally fine if the reported elements are only used within that specific table. If a table with repeating rows has a total at the bottom, the totals will be reported without a typed dimension value. For those elements used as a total, the element must be included in a table that allows the reporting of default values. The FERC taxonomy includes a specific role and definition linkbase called the default table. This table includes elements included in typed dimension tables so that default values can be reported. This table is included in all published Form entry points and schedule entry points that used typed dimensions that report default values.

Typed dimensions use the segment portion of the context in the instance document.

3.4.3 Explicit Dimensions

Explicit dimensions are defined in the taxonomy. All explicit dimensions have a default domain defined and must be closed. Explicit dimensions use the segment portion of the context in the instance document.

3.5 Labels

The taxonomy defines different label types for every schedule. This design ensures that there is no potential conflict between label roles. The form name is included as part of the label role. For example, the instructions associated with a schedule has a specific label role for instructions that are linked to the root node of the schedule. The label role for instructions for form 1 is:

```
F1Instructions
```

The label role for form2 instructions is:

```
F2Instructions
```

This approach is used because the same schedule may be used in two different forms but may have slightly different instructions.

The labels for specific schedules are also unique. For example, the labels on the Form1 Income Statement have their own label role:

```
F1IncomeStatement
```

This label is then used to render Form1. Each element in the income statement also has a standard label. The difference is that the standard label is used to reference the element whereas F1IncomeStatement is the label that is used when rendering the Form 1 Income Statement. For example, the standard label of Accretion Expense is Accretion Expense but the label for the Form1 income statement role “F1IncomeStatement” is:

```
Accretion Expense (411.10)
```

3.5.1 HTML Labels

In some cases, labels are defined as HTML. HTML labels allow the inclusion of formatting in the label. HTML labels are always used for instructions associated with the schedules. The instructions at the top of the schedule are usually formatted as ordered lists. To ensure that schedule instructions are appropriately rendered they are included in the taxonomy in an HTML format. When they are rendered, they will appear appropriately on the form. HTML labels are discussed in section 5.3 The Form Template.

3.6 Confidential Data

A filer has the ability to request that data submitted as part of the filing be treated as confidential. To indicate that a fact is confidential the filer uses the confidential footnote arc role. This is a FERC-defined arc role defined in the taxonomy that indicates that the fact is confidential. The footnote arc role is a self-referencing arc role that goes from the fact back to the fact. (See ***Figure 3-1 Confidential Footnote arc*** This allows the FERC processor to identify these facts and redact them for public consumption.

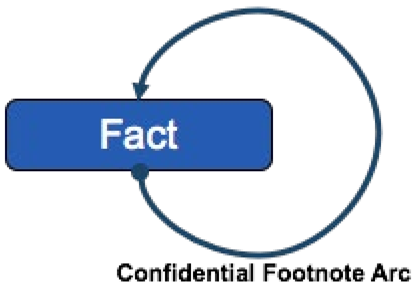


Figure 3-1 Confidential Footnote arc

3.7 Presentation Linkbase

Presentation linkbases are used in the FERC taxonomy for the following reasons:

1. To indicate the labels that are appropriate for a given schedule
2. To allow for easy review of the taxonomy in a taxonomy viewer tool
3. To define the order of schedules
4. To aid in the validation of a filing.

All schedules include a presentation linkbase.

All elements used by the taxonomy are defined in a core taxonomy file of elements irrespective of the form being reported. It is possible for a filer to report values for elements that appear in other forms. To control for these types of errors, the taxonomy contains validation rules that check that fact values have only been reported for elements that are defined in the presentation linkbase for the form.

3.8 Calculation Linkbase

Calculation relationships between elements are defined in the taxonomy. These calculations are included for information purposes so that users can understand the calculation relationships between elements. These calculations are not always complete because of reversal items contained in many schedules. These cannot always be accurately represented using the calculation linkbase. For this reason, the calculation linkbase should be used to assist in understanding the meaning of the disclosures but should not be used for validation purposes.

Validations are performed using a separate validation module defined using the XULE syntax.

3.9 Validations

The taxonomy includes validations represented as rules. A filing should validate against these rules to be successfully filed. Each form has a set of validation rules applicable to it. In addition, there are validation rules that apply to all forms. These are organized so that validation rules do not need to be duplicated by the taxonomy manager.

The validation rules are expressed as XULE rules and are maintained by the taxonomy author.

The validation rules can be classified into the following groups:

1. Required disclosures
2. Calculation Checks
3. Rollforwards
4. Negative Values
5. Filing Restrictions

3.9.1 Rule Categories

3.9.1.1 Required Disclosures

These rules check that required disclosures within the form have been made. The rule checks for the existence of a value for the element in the current reporting period. If a value is not reported the rule evaluates that the value is not reported and logs an error.

3.9.1.2 Calculation Checks

The calculation checks recalculate the totals and subtotals defined in the Form. The calculation checks can support various tolerances due to rounding errors. The rules currently allow differences due to rounding. The rounding is specific to the decimals associated with the components of the calculation. In addition, calculation checks can add values broken down by a specific dimension to ensure that subtotals of dimensions are accurately reported.

3.9.1.3 Roll Forwards

The validations include roll forward calculation checks to ensure that the differences between opening and closing balances of an account are calculated accurately.

3.9.1.4 Negative Values

The validations include negative value checks to determine that values in a schedule cannot be entered with negative values where negative items are not permitted.

3.9.1.5 Filing Restrictions

Filing restrictions are designed to prevent deviation from the FERC filer manual. These rules include the following:

- Duplicate values are not reported in the filing.
- Additional disclosures are excluded from the filing. Because the filing is based off a core set of shared elements, it is possible for filers for example to report values for elements included in other forms or to report data for historical periods that are not required by the form. When rendered as a form these values would not appear. These rules warn the filer that they have made disclosures that do not appear on the form, even though they are XBRL valid.
- OrderNumbers are not duplicated
- Fact ids are included for every fact.

3.9.2 Rule Format

Validations are expressed as discrete rules. Each rule has an identifier that is associated to the form and the schedule that applies to the rule.

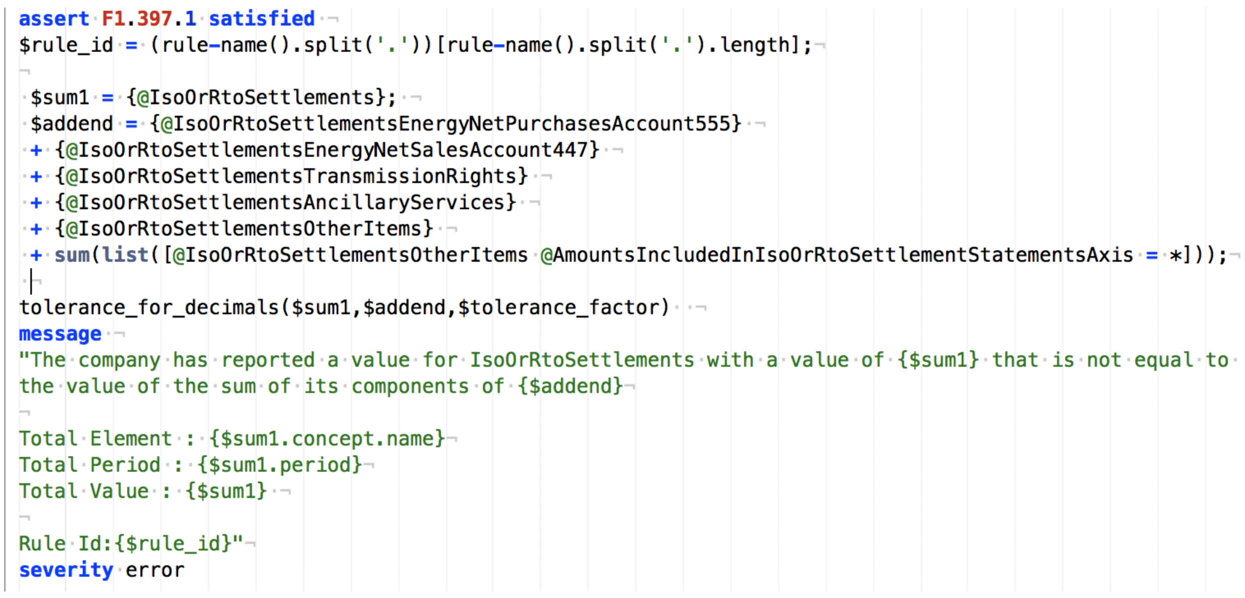
Each rule includes a description of the error and the fact that relates to the error.

Validations have varying degrees of severity. Severities currently defined are as follows:

- Error: The condition defined is an error and must be fixed prior to filing.
- Warning: The condition defined is incorrect but will not prevent the filing from being submitted.
- Information: The condition could be a problem but is reported for the filer to check.

In the example below is a rule for Form 1, schedule 397. The name of the rule is FERC.F1.397.1. The naming convention defines this information. The last 1 represents that this is the first rule for this schedule and differentiates it from other rules for the schedule. This rule has an error severity which means it must be resolved. The rule defines a calculation relationship and checks that the two calculated values should be the same.

All calculation rules include a function that allows the tolerance between two numbers comprising an equivalency test to be adjusted globally. Each rule also includes a message that details the facts and values comprising the calculation that failed the validation.



```
assert F1.397.1 satisfied
$rule_id = (rule-name().split('.'))[rule-name().split('.').length];

 $sum1 = {@IsoOrRtoSettlements};
 $addend = {@IsoOrRtoSettlementsEnergyNetPurchasesAccount555}
 + {@IsoOrRtoSettlementsEnergyNetSalesAccount447}
 + {@IsoOrRtoSettlementsTransmissionRights}
 + {@IsoOrRtoSettlementsAncillaryServices}
 + {@IsoOrRtoSettlementsOtherItems}
 + sum(list([@IsoOrRtoSettlementsOtherItems @AmountsIncludedInIsoOrRtoSettlementStatementsAxis = *]));

tolerance_for_decimals($sum1,$addend,$tolerance_factor)
message
"The company has reported a value for IsoOrRtoSettlements with a value of {$sum1} that is not equal to
the value of the sum of its components of {$addend}

Total Element : {$sum1.concept.name}
Total Period : {$sum1.period}
Total Value : {$sum1}

Rule Id:{$rule_id}"
severity error
```

4 Taxonomy Versioning

The key principles followed for versioning the taxonomy are as follows:

1. Only one version of the taxonomy will be maintained at a given point in time.
2. All prior versions of the taxonomy will be available to support historical submissions
3. All form entry points are static. The content of a Discoverable Taxonomy Set (DTS) will never change over time.
4. Changes to a form will result in a new entry point being published.
5. Namespaces are tied to the release version of the taxonomy. Each release of the taxonomy has a separate namespace.
6. Once a taxonomy is published it cannot be subsequently amended.

Updated releases of a specific form will not require a re-release of all other forms. For example, updates to Form 1 may be made on one date, and updates to Form 2 on another date, even though both forms use the same base taxonomy elements.

When an updated form is published the FERC will only publish an updated entry point for that form. Because the FERC Forms taxonomy is a single taxonomy, an update to one form could potentially change another. For example, if an updated release of Form 1 corrected a standard label in an element used in both Form 1 and Form 2 what would happen to the filers filing Form 2? The FERC would make the correction and release the entry point for Form1 as well as all the updated taxonomy files. Any users of Form 1 would switch to the new entry point on a designated date. The FERC would not publish an updated entry point for Form 2. Form 2 filers would continue to use the prior taxonomy. When an updated Form 2 is released the earlier change made in Form 1 would already be incorporated into the updated Form 2. At no point would the FERC staff be maintaining two versions of the taxonomy. There will always be multiple versions of the taxonomy in use by the filers, but as updated forms need to be released the old form entry point would always be superseded by the Form in the current version of the taxonomy.

Each approved entry point is unique and will have an associated hash to verify that the collection of taxonomy files used is consistent over time for that entry point.

When a filer needs to resubmit a prior version of a filing the entry point appropriate to that filing period is used. Every historical version release of the taxonomy is available so filers can resubmit prior versions of the filings with the taxonomy that was appropriate at the time.

The diagram below shows the entry points used for a taxonomy release.

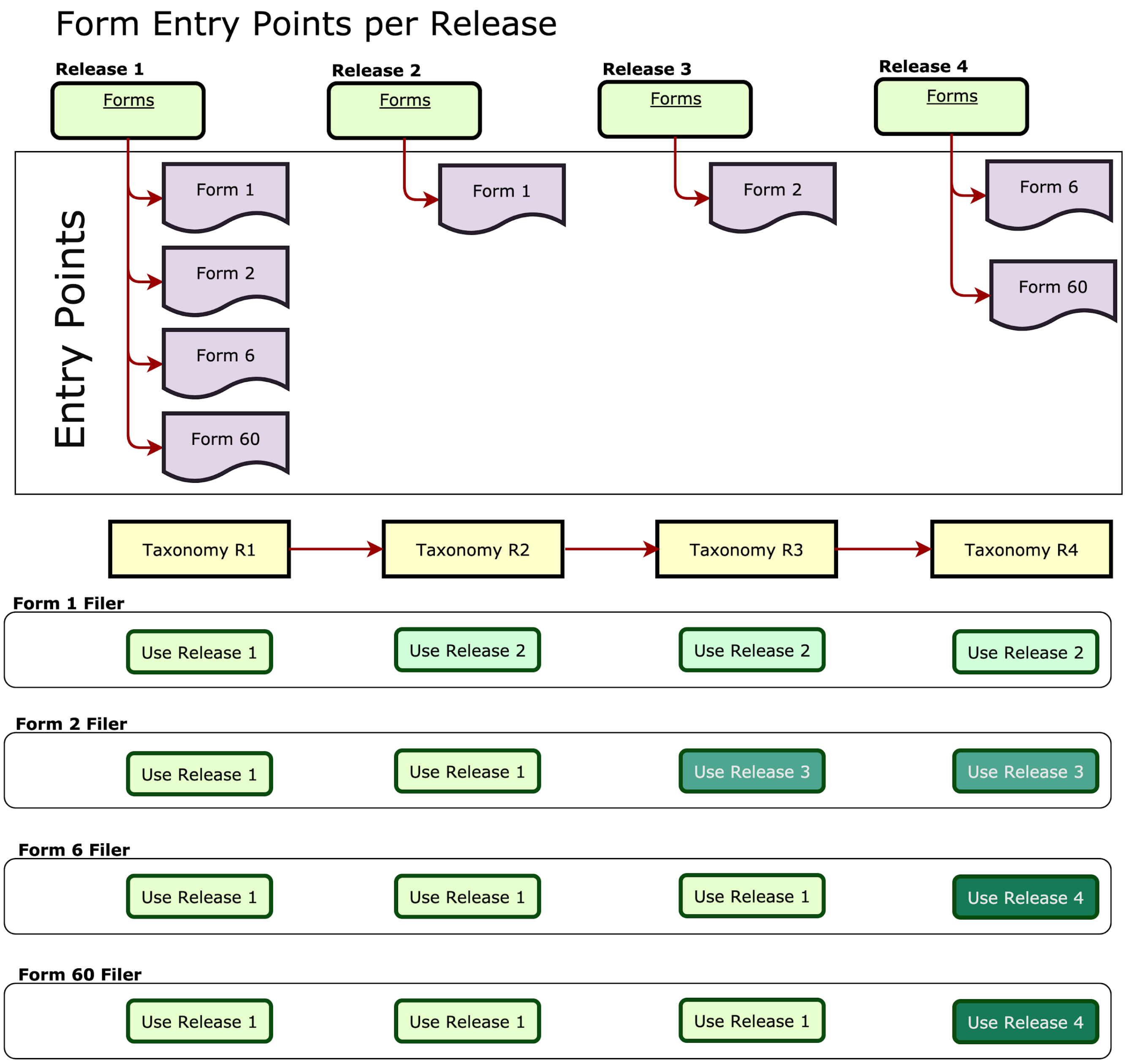


Figure 4-1 Form Entry Points Per Release

The submission portal will check the appropriateness of the taxonomy release version based on the year and period of the filing. The release version of the taxonomy will include the period and year end date that should be used with the filing. {Should the filings that a release is applicable to be included in the taxonomy or should this be recorded somewhere else.}

In no cases will the element names, data types, periodicity or balance attributes be changed. The taxonomy is monolithic, and elements must remain constant over time to ensure continuity of data over time. If existing elements need to be amended because of an error in the original taxonomy or a rule change then a new element with a different element name will be created.

5 Form Rendering

5.1 Overview

When an XBRL filing is made to the FERC one of the functions that needs to be performed is to create a human readable version of the XBRL instance that looks like a traditional form. The following figure shows the process flow of a filing and where form rendering occurs:

Form Rendering in Submission Process

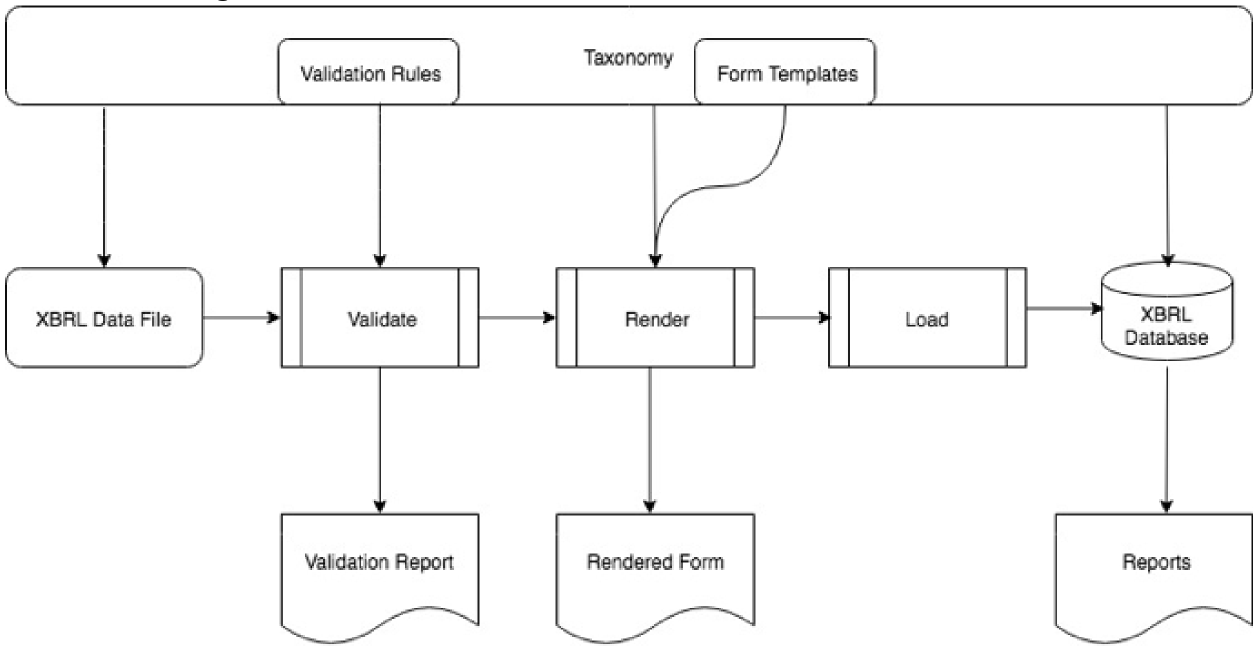


Figure 5-1 Form Rendering

The rendered forms shown above as a result of the filing process are rendered using 3 components

1. Instance document (Contains the filers data)
2. The FERC taxonomy
3. A rendering template for each form

The form template defines the format of the form such as the font style, borders, placement on the page etc. Most text on the form should come from the taxonomy.

The advantages of using a form template are as follows:

- Data only needs to be entered once
- Forms can be updated without the need to change any software
- Historical data can be rendered in an updated form
- Forms can be rendered as inline XBRL
- Rendering code is open source so utilities can render before filing

The rendering also allows the rendering of blank forms. The following diagram shows the components required to render populated and blank forms.

Rendering Components

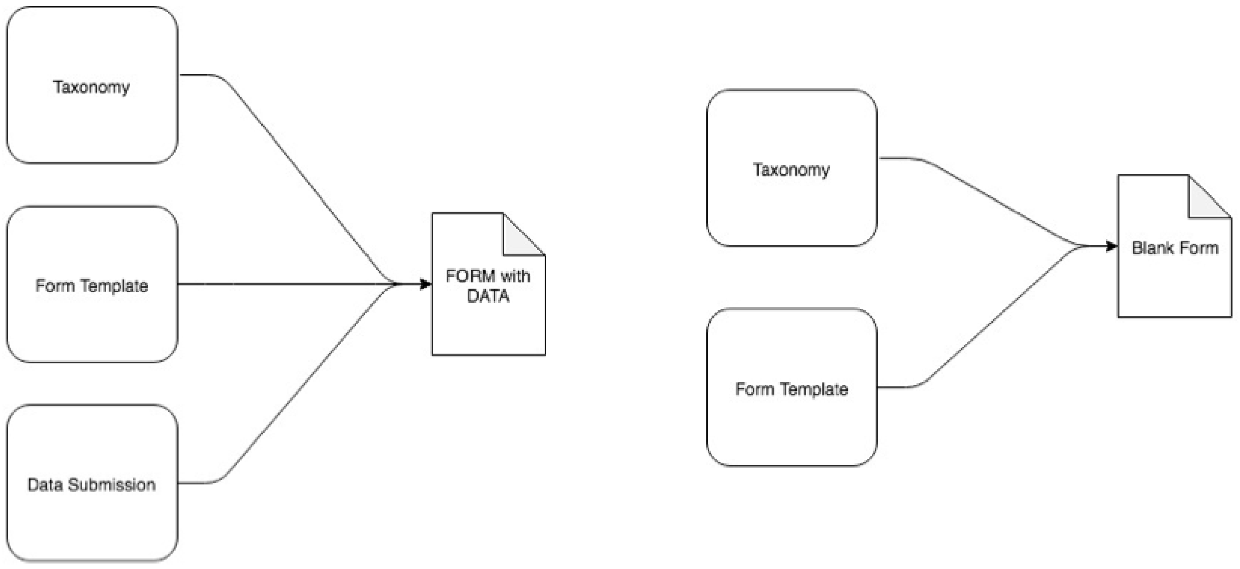


Figure 5-2 Rendering Components

The technology used to render the forms relies on the XULE language to express how data from the taxonomy and the instance should be meshed together to create a rendering of the form.

The resulting form is rendered in an xhtml format as an inline XBRL document. This makes the FERC filing program similar to programs adopted by the US Securities and Exchange Commission (SEC) and other global regulators. It also has the advantage that many of the FERC filers will be familiar with this approach as they already file in inline XBRL to the SEC.

Because the document is in an xhtml format, formatted text can be included in the filing in an xhtml and will be rendered consistently.

5.2 The Taxonomy

The taxonomy contains data that is used to render the form. Changes made to the taxonomy will automatically update the rendered form. Generally speaking, any change to the form is going to require a new taxonomy release. The information in the taxonomy used to render the form is as follows:

1. Form Instructions
2. Labels of the elements
3. Page number of the form
4. Form column headings
5. Legacy row numbers.

6. Number of rows in the form

In some forms the taxonomy can also be used to control the number of columns that appear in the form.

The taxonomy contains the static data about the form which is encapsulated in FERC regulations. Any changes in FERC regulations are implemented by making changes to the taxonomy, eliminating the need to make changes to the code that processes and renders the data. In addition, blank forms will be able to be generated from the taxonomy directly.

The taxonomy does not contain any information about the actual formatting of the form itself. For example, what is the width of a column? what color is shading? what is the format of the text? etc. The taxonomy does not define how the data is laid out on the form. All formatting is controlled by the form template using a single CSS stylesheet.

5.3 The Form Template

An important requirement of the XBRL implementation is to make the availability of the data compatible with historical form formats. The creation of rendered forms helps to facilitate the change to XBRL data by presenting the data in a familiar format. To render XBRL data as a form, formatting information needs to be defined somewhere. In this case, formatting data is defined in the form template.

The form template is defined as an html document template that can be viewed in a browser. There is a single template for each schedule of the form. The template contains special tags or elements that defines how XBRL data is inserted into the template. If a form is updated it may or may not be necessary to update the template associated with a schedule. This depends on the nature of the change. Changes to names or additional rows or changes to instructions will generally only require a change to the taxonomy.

5.3.1 HTML Structure of the form template

All FERC forms are defined using html tables. A table is comprised of rows and columns in a specific table. The form templates usually contain a number of tables. The first table is used to represent the column header. This is generally the same across all schedules and looks like the following:

Name of Respondent	This Report Is: An Original A Resubmission	Date of Report	Year/Period or Report End of:

This row is expressed in a table using an html tag that looks like the following:
<table class ="xbrl ident-table">

This table has a single row. Each row is described using the <tr> tag. The table row <tr> includes 4 cells that are represented using the element <td>. This table is generally used for almost all schedules in the form. This table will be identical across all schedules.

Secondly the next table below the identification header includes schedule title and instruction of each schedule as shown below.

Schedule Name
Instruction 1. …. 2. ….

These two rows are expressed in

```
<table>
        <tr class=”sub-title xbrl”>
                <td>Schedule Name</td>
        </tr>
        <tr>
                <td class = “xbrl instruction”>Instruction</td>
        </tr>
        …….
</table>
```

Instructions in the taxonomy can be defined in a valid xhtml format, so they can be rendered cleanly in the template. If not expressed in xhtml then the instructions will appear as unformatted text. This may be appropriate for instructions that are a single line of text.

After the row with the instructions, the schedules are defined using html and values inserted from the filing instance document. In many forms data is represented as sequences of items. An example of a sequence table is Form 1, Schedule 104 Officers, which repeats rows for each director.

The example below shows a table that repeats a row for each director. Rendering this table requires that rows of data can be repeated based on the number of officers in the instance document. The template uses XULE expression and special html tags to insert values from the taxonomy and the instance to generate the data.

The example below shows the officer schedule with example data to show the headings and the table rows. When the template is rendered, the headings and data in the rows will be replaced with the data from the instance document when it is rendered.

Line No.	Heading 1	Heading 2	Heading 3
1	aaaa	$10,000,000	$10,000,000
2	bbbb	$10,000,000	$10,000,000
3	cccc	$10,000,000	$10,000,000

5.3.2 Components of the form template

The template uses css to control the format of a form. The same css is used for all forms and schedules. This means that forms have a consistent look and feel across all schedules and forms. It also means that a change to all forms can be done quickly. For example, text could be changed from sans-serif to serif if need be.

The template contains special tags to control where inserted data is placed. Any tag with the prefix “xule” is a special tag. For example:

```
<xule:expression>
```

All tags with the prefix `xule` dictate how data from either the instance or taxonomy will be inserted into the form. If you look at the template in a browser these xule tags are hidden.

5.3.2.1 Types of XULE tags

5.3.2.1.1 XULE Expression

The first XULE tag we will cover is xule expression. `<xule:expression>`

This tag will contain a xule expression that evaluates to some kind of value, that can be pulled either form the taxonomy or from the instance document.

Attributes
This tag has a number of allowable attributes. The first is **html**. If the expression has this attribute the values returned will be rendered as html. If this is missing, then html tags returned will be output as string values. The expression tag with the html attribute is defined with a boolean value as follows:

```
<xule:expression html="true">
```

If the html attribute is left off it is the same as:

```
<xule:expression html="false">
```

In addition to the html attribute the expression can also contain a **class** attribute. This is used to control formatting of the template in a browser. In the example below the value of the class is hide. This allows the user to toggle if they want to see the xule expression or not in the template.

```
<xule:expression html="true" class="hide">
```

The third attribute of the expression tag is the **name** attribute. This attribute allows a series of expressions to be grouped together as one across the template. This is discussed in more detail later in the document.

```
<xule:expression html="true" class="hide" name="ISHeadings" >
```

The fourth attribute is the **fact** attribute. This attribute controls if the value of the expression is returned as a value from the instance that is rendered as an inline XBRL fact. If this attribute is set to true, then the value in the html version of the form will appear as an inline XBRL fact. This means that in an inline viewer you will be able to drill down into the fact and see all the details about it, such as the element name, its unit of measure, its description in the taxonomy etc.

If this attribute is left off the value will be returned as a string. The fact attribute is expressed as the following:

```
<xule:expression html="true" class="hide" name="ISHeadings" fact="true" >
```

The fifth attribute of tag is called **part**. The part attribute allows the renderer to layout lists of results that are returned from the xule expression. For example, a xule expression may return a list of 4 results. The part attribute tells the renderer that we are expecting n results and dictates that only a part of that result is being rendered in a particular location.

The part attribute should always be an integer value. The part attribute is defined as follows:

```
<xule:expression class="hide"  name="formula" fact="true" part="3">
```

The sixth attribute of the expression tag is **format**. This allows the data returned to be formatted using inline XBRL format translations. For example, to convert a date from the XBRL format of “2018-12-31” to a slash date format such as “12/31/2018” the format is used with an inline formatting transform.

```
<xule:expression class="hide"  name="formula" fact="true" part="3" format="ixt1:dateslashus"
>
```

The inline transformation of ixt1:dateslashus will convert the date when rendered to use a slash format.

The seventh attribute is the **scale** attribute. This allows the data shown in the form to be scaled when rendered. This follows the format in inline XBRL. For if a value of 4,500,000 was shown in the form as 4,500 because it was scaled to 1,000's then a scale of 3 would be added to the fact as shown below.

```
<xule:expression class="hide"  name="formula" fact="true" part="3" scale="3" >
```

The eighth attribute is the **sign** attribute. This is used if the value in the formatted version should be shown as a negative item in the form but is shown as a positive item or is shown as a positive item in the form and a negative value in the XBRL instance. The sign attribute uses a value of "-" to indicate a sign flip.

```
<xule:expression class="hide"  name="formula" fact="true" part="3" sign="-">
```

In summary the xule:expresion element has the following attributes:

Attribute	**Value**	**Example**
html	Can only be set to true	`html="true"`
class	Can have any valid css class	`class="hide"`
name	Can have any value, but the name should be the same across expressions	`name="ISHeadings"`
fact	Can only be set to true	`fact="true"`
part	Must have a value of an integer	`part="3"`
format	Must be a valid inline transformation	`format="ixt1:dateslashus"`
scale	Can be a positive or negative integer and must be used with a numerical value.	`scale="3"`
sign	Can only have a value of "-" and must be used with a numerical value.	`sign="-"`

XULE Expressions

To use the xule expression element it must contain a valid xule expression. A xule expression allows the user to pull data from either the FERC taxonomy and the filed instance being rendered. For example, to put a label defined in the taxonomy on a form for a column heading can be performed using a xule expression. As an alternative the label could also be hard coded on the form template. However, to reduce maintenance in future it is always preferable that form content that changes should be defined in the taxonomy and not in the template.

To get the label for the legal name of the respondent in a form the following can be defined:

```
<xule:expression class="hide">
taxonomy().concept(ferc:RespondentLegalName).label("http://ferc.gov/form/20
20-01-01/roles/label/F1Heading").text </xule:expression>
```

This only has one attribute and will return the value “Name of Respondent:”. The xule expression instructs the renderer to go to the taxonomy and get the concept called `RespondentLegalName` and return the label type “F1Heading” associated with the element and return it as text. This may seem unnecessarily long, but an element can have multiple labels associated with it, which also can be expressed in multiple languages. These qualifiers are very powerful because in other schedules of the form this same element is rendered with a different label i.e. “01 Exact Legal Name of Respondent”. It also means that if the label is updated in the taxonomy it will be updated across every schedule of the form.

The name of the actual legal entity is reported in the filer’s XBRL instance. The following XULE expression retrieves the respondent legal entity name.

```
<xule:expression class="hide"
fact="true">[@ferc:RespondentLegalName]</xule:expression>
```

The expression gets the value of the respondent legal name with no dimensions.

The resulting rendered values for the label and the data from the instance is:

Name of Respondent: FERC Test Company

5.3.2.2 XULE Replace

The second XULE tag is the xule replace tag: `<xule:replace>.` The xule replace element tells the rendering engine to take the content inside it and evaluate it and replace with the evaluated values. This element must always contain either an expression element or line number element (covered in a section below) and an optional template display element.

The following example shows how the replace element is used:

```
<xule:replace>
      <xule:expression class="hide">
taxonomy().concept(ferc:UsesFormulaRates).label("http://ferc.gov/form/2020-01-
01/roles/label/F1FormulaRates").text
      </xule:expression>
      <xule:template-display>
        Does the respondent have formula rates?
      </xule:template-display>
</xule:replace>
```

In this case everything in the replace will be replaced with the value returned by the expression. In this case the string “`Does the respondent have formula rates?`”.

Note also the template-display tag. This value is also replaced. However, this is included to show an example of what the rendering could look like when viewing the template in a browser. This is used when viewing the template prior to rendering, to show examples of the data that could be returned by the expression.

The xule replace tag has no attributes. All xule:expression tags should be included within a xule:replace tag.

5.3.2.2.1 XULE Repeat

Xule repeat is used as an attribute of a table row <tr> element. The xule repeat tag instructs the renderer to repeat the row for each line in a sequence of data returned by an expression. The following shows that the table row will be repeated for the expression that is called ConstructionWorkInProgres.

```
<tr class="schedule-row" xule:repeat="ConstructionWorkInProgres">
```

5.3.2.2.2 XULE Class

The xule class element is used to assign a css class to a value inserted into the template. If a value returned from a xule expression meets certain criteria it can be rendered in different ways using css classes. For example, if the value is negative it could be rendered in red, or a table cell could be shaded if the value is for an abstract element that will never have a value.

The xule class element can have two attributes. The first attribute is the location attribute which is optional. The location attribute can have a value of “parent”. This will apply the class defined in the expression and apply it to the parent node rather than the current node. In the example above the expression checks if the element in an abstract element. If the element is an abstract element, then the rendered version will gray out the parent node which is the table cell. The resulting rendering of the <td> element will have a class added to it called “gray-out”

```
<td class="gray-out"></td>
```

The second attribute is hide. This can have a value of true which will hide this element when viewing the template in a browser. This is the same as the hide attribute used on the xule expression element.

The xule:class acts to insert a new class in the element that is rendered in the final xhtml version. In the example below the xule class is used to shade a box in the balance sheet that is an abstract item. It looks at the value and determines if the value is an abstract. If it is true, then a class called grayed out is added to the parent element. In this case, this is the td element where the value appears.

```
<xule:replace>
      <xule:expression class="hide" name="BSLineItems" part="4"> $rowl[5]
</xule:expression>
       <xule:class location="parent" class="hide" >if $rowl[1].is-abstract "gray-
out" else ""</xule:class>
</xule:replace>
```

5.3.2.2.3 XULE Line Number

The xule line number creates a sequential list of numbers that are added to a column in a form. This is used on those schedules which are repeating rows. The line number has the name attribute associated with it. This name must match the name used for the rule expression that generates the sequence.

The line number also can include child nodes that indicate the start number of the sequence. The child node controls where the line number starts. The tag startNumber allows the template designer to start a sequence number from a number other than 1. To start a sequence number at 16 this is expressed as follows:

```
<xule:lineNumber name="InvestmentInSubsidiary">
<xule:startNumber>16</xule:startNumber>
</xule:lineNumber>
```

5.4 Handling Periods

FERC forms contained a number of reporting periods. Most values are reported in the current reporting period with values also reported for prior periods for comparative analysis and to record opening balance of accounts.

To render the data in the template the rendering engine uses variables for each of the various time periods reported in a FERC form. The values of these variables represent an instant date or a duration of time. Each of these dates are determined from the Report year and the report period. The report period defines the duration and the report year defines the year.

The rendering templates use the following variables:

```
$currentDuration = $current-start to $current-end
$currentInstant = $current-end
$priorDuration = $prior-start to $prior-end
$priorInstant = $prior-end
$threeMonthsCurrent =
$threeMonthsPrior =
$current-start = reportYear + $reportPeriodLookup[0]
$current-end = reportYear + $reportPeriodLookup[1]
$prior-start = reportYear -1 + $reportPeriodLookup[0]
$prior-end = reportYear + $reportPeriodLookup[1]

$reportPeriodLookup = {'Q4': ('01-01', '12-31'),
                 'Q3': ('07-01', '12-31'),
                 'Q2': ('04-01', '06-30'),
                 'Q1': ('01-01', '03-31')}
```

When extracting values from an instant document to add to a template, the values can then be referenced by the variables such as $currentDuration.

For example, to get the value of NetIncomeLoss for the current reporting period the following xule expression could be used.

```
<xule:expression class="hide" name="ISLineItem">
[@concept = NetIncomeLoss @period = $currentDuration]
</xule:expression>
```

5.5 Standard Classes Used

The following defines the standard classes defined in a css stylesheet to control formatting of the forms.

Table 5.1 Standard Classes

Class	Description
abstract	Used to indicate that the value is an abstract item. These are usually headers and this class bolds the value
account-number	Used to indicate that a value is an account number so that account number formatting can be done in a consistent manner
body.xbrl	Allows the body of the template to be formatted differently from the body of note disclosures that may be inserted into the template.
table.xbrl	Allows the tables of the template to be formatted differently from tables that are

Class	Description
	inserted into the template as a fact. For example, a footnote disclosure in an xbrl format should not inherit the table formatting of the template.
td.table	Allows the cells of a table in the template to be formatted differently from cells that are inserted into the template as a fact. For example, a footnote disclosure in an xbrl format should not inherit the cell formatting of the template.
hide	Used to hide template formatting options when viewing the template.
date_items	Used to identify items that are dates
numeric_items	Used to identify items that are numeric value
sch-title	Used on columns of a schedule to define the formatting of the schedule column.
monetary_items	Defines the format for monetary amounts. All monetary amounts should use this class.
percent_items	Used to identify items that are percent items.
total_items	Used to identify items that are total items.
gray-out	Used to indicate if a cell should be grayed out.
col-heading-row	Used to indicate the row that is a column heading