Introduction

As the University of Washington (UW) expands its efforts to track environmentally sustainable efforts across its campuses, it becomes more and more urgent for this data to be stored and managed centrally to ensure consistency and open access to the information. As part of an effort to make central management possible, this project creates a "first draft" effort at defining the structure of the data that the University has tracked and must continue tracking in order to ensure its sustainability. Incorporating reviews of documents available to various UW stakeholders, interviews, and various information management and user centered design methodologies, this XML structure is being proposed for active use within the UW.

This documentation provides an overview of XML, then proceeds to describe the main elements of the structure itself.

A Quick XML Primer

XML is a data format intended for data portability and the structured storage of information. While it bears similarities to HTML (the markup language used to create web sites on the Web), it allows data designers much more flexibility in defining their own terminology, constraints, and vocabularies.

XML code looks like this:

```
<element>
      <subElement attribute="value">
            Some text here.. this is likely data.
      </subElement>
</element>
```

First, the main building blocks of XML are called *elements*. You can think of elements as containers that can either hold other elements or actual data. Element names are noted by the "<" and ">" characters; spaces are not allowed within element names.

Second, each element must be opened and closed (they will almost always appear in matching pairs). An open tag does not have the "/" prior to the tag name.

Third, each element may or may not have an attribute. Attributes describe an element in some way. They are denoted by `attribute="value"` within the element, and are separated from the actual element name by a space. An element may have multiple attributes or no attributes at all.

Metadata Structure Overview

The table below describes the actual structure of the metadata schema. Asterisks (*) indicate unlimited values. Attributes are omitted in favor of being documented on an element-by-element level. Both input and output elements use sub-elements prefixed with "C".

		Level				
	Root	**Two**	**Three**	**Four**	**Cardinality**	**Optional?**
1	university				1	
1.1		input			*	
C.1			materialRef		1	
C.2			Location		1	
C.2.1				name	1	
C.2.2				address	0-1	Yes
C.2.3				administrativeUnit	0-5	Yes
C.3			amount		*	
C.4			note		0-*	Yes
1.2		output			*	
C.1			materialRef		1	
C.2			location		1	
C.2.1				name	1	
C.2.2				address	0-1	Yes
C.2.3				administrativeUnit	0-5	Yes
C.3			amount		*	
C.4			note		0-*	Yes
1.3		material			*	
1.3.1			name		1-5	
1.3.2			chemicalFormula		0-1	Yes
1.3.3			utilityGenerated		0-1	Yes
1.3.3.1				traditional	0-1	Yes
1.3.3.2				renewable	0-1	Yes
1.3.4			note		0-*	Yes

Metadata Schema Usage

When creating schema elements, create definitions for the materials being described first. Inputs and outputs should then be entered using the material ID generated in the original definition. If new materials are needed, create definitions for those materials prior to defining input and output data.

All attributes with a numeric identifier beginning with "C" indicate elements common to both inputs and outputs and are grouped together towards the end of this documentation section.

1	university		
	Required:	Yes	
	Cardinality:	1	
	Type:	None (container element)	
	Value Restrictions:	None	
	Attributes:	None	
	Definition:	The root element for the schema. This only needs to be provided once; subsequent data entry should take place within the opening and closing tags for the element.	

1.1	input		
	Required:	Yes	
	Cardinality:	Unlimited (1 minimum)	
	Type:	None (wrapper element)	
	Value Restrictions:	None	
	Attributes:	id	A unique identification string that differentiates this record from others within the file. For inputs, the string should begin with "in_" (without the quotes), followed by an identification number.
		references	Optional. Establishes relationships between different elements within the file – if one input references another element (for whatever reason), this value should contain the identification string of the element being referenced. *This is not used to indicate materials being measured.*
		day	Optional. Contains the day in two-digit (DD) format. **Value Restrictions:** any number between 1 and 31.

		month	Optional. Contains the month in two-digit (MM) format. **Value Restrictions:** any number between 1 and 12.
		year	Contains the year in four-digit (YYYY) format. Indicates the year that the input occurred. **Value Restrictions:** Any number between 1800 and 9999.
		rangeStart	Optional. Indicates the start of a date range in YYYY-MM-DD, YYYY-MM, or YYYY formats.
		rangeEnd	Optional. Indicates the end of a date range in YYYY-MM-DD, YYYY-MM, or YYYY formats.
		timeStart	Optional. A free-text entry that specifies a specific start time for the data range.
		timeEnd	Optional. A free-text entry that specifies a specific end time for the data range.
		type	Optional. Indicates whether the input is a direct or indirect input. **Value Restrictions:** direct, indirect
		benchmark	Optional. A true/false value indicating whether the input element is being used as a benchmark to compare against other data. **Value Restrictions:** true, false
	Definition:	An input describes an instance of a material entering into the University's ecosystem. Any material defined within the file structure may serve as an input to the system.	

	XML Example:
	
	`<input id="in_00001" year="2008" month="05" day="15" type="direct">`
	`  <materialRef>m_00001</materialRef>`
	`  <location>`
	`  <name>Seattle</name>`
	`  </location>`
	`  <amount unitOfMeasurement="count">3000</amount>`
	`  <note>This is an estimated value.</note>`
	`</input>`		

1.2	output

	Required:	Yes	
	Cardinality:	Unlimited (1 minimum)	
	Type:	None (wrapper element)	
	Value Restrictions:	None	
	Attributes:	id	A unique identification string that differentiates this record from others within the file. For outputs, the string should begin with "out_" (without the quotes), followed by an identification number.
		references	Optional. Establishes relationships between different elements within the file – if one input references another element (for whatever reason), this value should contain the identification string of the element being referenced. *This is not used to indicate materials being measured.*
		day	Optional. Contains the day in two-digit (DD) format. **Value Restrictions:** any number between 1 and 31.
		month	Optional. Contains the month in two-digit (MM) format. **Value Restrictions:** any number between 1 and 12.
		year	Contains the year in four-digit (YYYY) format. Indicates the year that the input occurred. **Value Restrictions:** Any number between 1800 and 9999.
		rangeStart	Optional. Indicates the start of a date range in YYYY-MM-DD, YYYY-MM, or YYYY formats.
		rangeEnd	Optional. Indicates the end of a date range in YYYY-MM-DD, YYYY-MM, or YYYY formats.
		timeStart	Optional. A free-text entry that specifies a specific start time for the data range.
		timeEnd	Optional. A free-text entry that specifies a specific end time for the data range. timeEnd may be omitted if there is no end time, in which case, timeStart should be interpreted as a fixed value.
		type	Optional. Indicates whether the output is a

			direct or indirect output. **Value Restrictions:** direct, indirect
		benchmark	Optional. A true/false value indicating whether the output element is being used as a benchmark to compare against other data. **Value Restrictions:** true, false
	Definition:	An output describes an instance of a material leaving the University's ecosystem. Any material defined within the file structure may serve as an output to the system (within reason).	

	XML Example:
	
	`<output id="out_00001" year="2008">`
	`  <materialRef>m_00002</materialRef>`
	`  <location>`
	`  <name>Bothell</name>`
	`  </location>`
	`  <amount unitOfMeasurement="lbs">23492320</amount>`
	`</output>`		

1.3	Material		
	Required:	Yes	
	Cardinality:	Unlimited (1 minimum)	
	Type:	None (container element)	
	Value Restrictions:	None	
	Attributes:	id	A unique identification string that differentiates this record from others within the file. For outputs, the string should begin with "out_" (without the quotes), followed by an identification number.
		references	Optional. Establishes relationships between different elements within the file – if one input references another element (for whatever reason), this value should contain the identification string of the element being referenced. *This is not used to indicate materials being measured.*
		type	Describes, in broad terms, the type of material being dealt with. **Value Restrictions:** recycled, virgin, ferrous, hazardous, other

		isTypeOf	Indicates relationships with other material types. This is more specific than the references element, which can describe relationships between more general items. This attribute explicitly constructs material hierarchies.
		chemical	A true/false value indicating whether the material described is a chemical. **Value Restrictions:** true, false
		greenhouseGas	A true/false value indicating whether the material described is a greenhouse gas. **Value Restrictions:** true, false
		recyclable	A true/false value indicating whether the material described is recyclable. **Value Restrictions:** true, false
	Definition:	Describes a material that is imported or exported from the UW's ecosystem. Can be a physical material or something intangible.	

	XML Example:
	
	`<material type="other" id="m_00002">`
	`  <name type="formal">electrical generation</name>`
	`  <utilityGenerated>`
	`  <renewable>Wind</renewable>`
	`  </utilityGenerated>`
	`</material>`		

1.3.1	name		
	Required:	Yes	
	Cardinality:	1-5	
	Type:	String	
	Value Restrictions:	None	
	Attributes:	type	Describes the type of name being entered. **Value Restrictions:** formal, informal
	Definition:	The name of the material (either formal, informal, or both).	

	XML Example:
	
	`<name type="formal">headcount</name>`
	`<name type="informal">student enrollment</name>`		

1.3.2	chemicalFormula	
	Required:	No
	Cardinality:	0-1
	Type:	String
	Value Restrictions:	None
	Attributes:	None
	Definition:	Notes the material's chemical formula, if applicable. Used only if the material is a chemical (as set in the material's attributes).
	XML Example: `<chemicalFormula>H2O</chemicalFormula>`	

1.3.3	utilityGenerated		
	Required:	No	
	Cardinality:	0-1	
	Type:	None (container element)	
	Value Restrictions:	None	
	Attributes:	Class	Indicates the type of utility being described. **Value Restrictions:** water, sewage, garbage, electricity
	Definition:	Indicates the type of utility used to generate or create the material (only useful in certain situations).	

	XML Example:
	
	`<utilityGenerated class="water"/>`
	`<utilityGenerated>`
	`  <renewable>wind</renewable>`
	`</utilityGenerated>`		

1.3.3.1	traditional	
	Required:	No
	Cardinality:	0-1
	Type:	String
	Value Restrictions:	steam, gas, coal, nuclear
	Attributes:	None
	Definition:	Lists traditional utility types (mainly useful for electricity generation).

	XML Example:
	
	`<utilityGenerated>`
	`  <traditional>steam</traditional>`
	`</utilityGenerated>`

1.3.3.2	renewable	
	Required:	No
	Cardinality:	0-1
	Type:	String
	Value Restrictions:	wind, hydro, solar, water
	Attributes:	None
	Definition:	Lists renewable utility types (mainly useful for electricity generation).

	XML Example:
	
	`<utilityGenerated>`
	`  <renewable>hydro</renewable>`
	`</utilityGenerated>`	

1.3.4	note	
	Required:	No
	Cardinality:	0-1
	Type:	String
	Value Restrictions:	None
	Attributes:	None
	Definition:	Allows a note to be made about a material.
	XML Example: `<note>This is an estimated value.</note>`	

C.1	materialRef	
	Required:	Yes
	Cardinality:	1
	Type:	IDREF
	Value Restrictions:	Material identification numbers only.
	Attributes:	None
	Definition:	A reference to a single material ID number as defined in a

		material tag. Identifies the material being used as an input/output.
	XML Example: `<materialRef>m_00001</materialRef>`	

C.2	location	
	Required:	Yes
	Cardinality:	1
	Type:	None (container element)
	Value Restrictions:	None
	Attributes:	None
	Definition:	The location in which the input or output occurred.

	XML Example:
	
	`<location>`
	`  <name>Seattle</name>`
	`  <administrativeUnit type="College">`
	`  Information School`
	`  </administrativeUnit>`
	`</location>`	

C.2.1	name	
	Required:	Yes
	Cardinality:	1
	Type:	String
	Value Restrictions:	Seattle, Bothell, Tacoma, Pack Forest, Friday Harbor Laboratories, Olympic Natural Resources Center, Big Beef Creek, Minor Outlying
	Attributes:	None
	Definition:	The general name of a location within the UW's sphere of influence.

	XML Example:
	
	`<location>`
	`  <name>Pack Forest</name>`
	`</location>`	

C.2.2	address

	Required:	No	
	Cardinality:	0-1	
	Type:	String	
	Value Restrictions:	None	
	Attributes:	None	
	Definition:	Stores the physical address of a location. Address is typically only used if the "name" element is set to "Minor Outlying". It may be used to indicate a higher level of granularity for other name values however.	

	XML Example:
	
	`<location>`
	`<name>Seattle</name>`
	`<address>1959 NE Pacific, Seattle, WA 98195</address>`
	`</location>`		

C.2.3	administrativeUnit		
	Required:	No	
	Cardinality:	0-5	
	Type:	String	
	Value Restrictions:	None	
	Attributes:	Type	Indicates the type of administrative unit being described. **Value Restrictions:** Central Administrative Group, College, Building, Department, Unit, Research Center
	Definition:	Indicates the administrative unit that is specifically being referenced as a part of this location. If this is not included, data is explicitly referring to the item listed in the name element.	
	XML Example: `<administrativeUnit type="Unit">Information Technology</administrativeUnit>`		

C.3	amount		
	Required:	Yes	
	Cardinality:	Unlimited (1 minimum)	
	Type:	Integer	
	Value Restrictions:	None	
	Attributes:	unitOfMeasurement	Indicates the unit of measurement being

			used to describe the amount of material. **Value Restrictions:** square foot, KwH, gallons, lbs, percentage, hours, kilowatts, megawatts, dollars, metric tons, megagram, terajoule, liter, megaliter, miles, count
	Definition:	The amount of the material being described as an input.	
	XML Example: `<amount unitOfMeasurement="count">300</amount>`		

C.4	note	
	Required:	No
	Cardinality:	Unlimited (no minimum)
	Type:	String
	Value Restrictions:	None
	Attributes:	None
	Definition:	Allows notes to be made about an input or output.
	XML Example: `<note>This is an estimated value.</note>`	

Design Notes

- When defining materials, the schema maintainer will occasionally end up defining the material as the unit of measurement for that material (that is, they are one and the same); thus, the unitOfMeasurement attribute for the amount element used in input and output notations is optional. However, providing both is still recommended (note that this will require changes to the controlled vocabulary list for this attribute).
- Note that if the UW decides to contain all data in XML, it will need to rearchitect the schema files somewhat to allow for smarter version control. My recommendation here is to rename any file beginning with "uwsustainability_" so that it begins with "uwsustainability_0.0.1_". The "uwsustainability_" files can then act as pointers to the most recent version while maintaining separate versions of each file as needed.

File Structure

The project XML files are structured as follows, with the master file at the top and included files lower in the hierarchy:

uwsustainability.xsd
This is the main schema definition file that ties the schema together. It contains all of the main attributes used by the schema itself.

uwsustainability_attributes.xsd
This file contains any attributes that are defined within the schema. These attributes are isolated from the main schema for maintenance purposes.

uwsustainability_complextypes.xsd
This file contains any complex types that are defined within the schema. These complex types are isolated from the main schema for maintenance purposes.

uwsustainability_vocabulary.xsd
This file contains any simple types that define vocabulary terms used within the schema. These vocabulary terms are isolated from the main schema for maintenance purposes.

Next Steps

1. Using the provided XML schema, enter a backlog of information (from the last year or so of available records). Ensure that the schema allows for sufficient robustness to represent all of the data available. The data sample should be fairly wide. Tweak as necessary.
   A needs assessment should be run to determine what departments need to be able to access and how they need to access it. Needs assessments are typically structured or unstructured interviews that attempt to gather information about how a user intends to utilize a particular resource. Interviews are not the only tool available – tools such as card sorts, participatory

observation (where the person doing the research watches someone as they move through everyday tasks), or focus groups can be extremely useful here. There are a number of resources available on conducting needs assessments available through UW Libraries – the following title is suggested, though I do not claim to have reviewed it thoroughly:

- *Needs Asssessement: A User's Guide*. Roger A Kaufman; Alicia Mabel Rojas; Hanna Mayer. Englewood Cliffs, N.J.: Educational Technology Publications, 1993. ISBN: 087778258X

This needs assessment determines who needs the data and how the data should be accessed for the greatest level of flexibility. This needs assessment greatly influences later steps.

2. Establish a set of guidelines that determines what data is in scope versus out of scope for this schema. Consider existing data sources across campus and minimize data duplication wherever possible. Note that the purpose of this schema is to unify disparate data sources into a single source, but that some data is best maintained by other departments across campus.
3. If the schema properly represents the data and all changes are completed, create an interface for entering the data. At this point, it might also be useful to change the XML into a database schema to address scalability issues. The needs assessment may be used here to assess the range of information that will be stored in the data schema; it may also be used here to begin preliminary designs for a data interface.
4. Begin preliminary designs for interfaces to enter and interact with the data stored by the schema.

Further Recommendations

- Since the University of Washington is an institution of higher learning, all effort should be made to keep students involved in the project on a cross-disciplinary basis, including, at minimum, students from the Information School to help address issues of data and knowledge management. Offering internships and capstone projects to Information School students is a wonderful way to continue this work.
- Consider creating a hire (likely at least 1FTE) to assist in data management and assimilation tasks. A good chunk of the responsibility for this position is likely to have to do with developing the frontend and testing the data entry abilities of the schema, as well as potentially converting the provided XML schema into a database format if the need for this is identified. The candidate should have knowledge of XML, XSLT, some web-based programming language (ASP.NET, PHP, others), experience with MSSQL Servers, SQL, needs assessments, usability testing, and user-centered design principles, as a start.
- Involve academic units in determining how this data can be translated into information that can then decrease costs and increase productivity.
- Consider refinements in data collection across the UW based on findings from the data entry testing done on the schema and reviews of currently existing data.