A Method of Teaching the Underlying Theory of XBRL: An Application of Information Modeling and XML Programming with a Microsoft Excel Implementation

Thomas Tribunella
State University of New York at Oswego

Heidi Tribunella
University of Rochester

Author Acknowledgements
We sincerely appreciate the valuable comments we received from the participants of the 2006 AIS Educators Conference. The editors and anonymous reviewers also supplied us with useful insights. In addition, we are grateful to Joan Sanders at the Mays Business School at Texas A & M University who assigned this lab to 88 students and provided us with constructive feedback.

Published by the AIS Educator Association
http://www.aiseducators.org
A Method of Teaching the Underlying Theory of XBRL: An Application of Information Modeling and XML Programming with a Microsoft Excel Implementation

Thomas Tribunella  
State University of New York at Oswego, tribunel@oswego.edu, (315) 312-2544

Heidi Tribunella  
University of Rochester, heidi.tribunella@simon.rochester.edu, (585) 275-3757

ABSTRACT

In this paper, we describe a computer lab exercise that will help students understand the underlying theory of XBRL. The lab requires students to build the taxonomy of accounts from a trial balance, create a well structured XML compliant document, load the XML document into Microsoft Excel, and produce a set of financial statements. The exercise follows the system development life cycle, is user friendly, and is very easy to understand. Bloom’s taxonomy of thinking skills was used to structure the learning objectives. At two hours in length, the exercise has been used in an AIS class and was received very well by the students. By the end of the lab, students should understand the concept of information modeling, markup technologies, and how they can be applied to accounting.

Keywords  
XBRL, XML, Markup Language, Taxonomy, Financial Reporting

INTRODUCTION

In this study, we propose a teaching methodology that will help students understand how markup languages such as XML and XBRL fit into the financial information reporting process. We demonstrate this concept using Microsoft Excel, Notepad and Explorer. In addition, the demonstration problem will give students hands-on experience with building taxonomies and XML compliant documents in a financial reporting context.

Markup languages such as XML (eXtensible Markup Language) and XBRL (eXtensible Business Reporting Language) can create a common storage and interchange format for accounting information. This can reduce redundancies and discrepancies in financial reporting as well as create repeatable processes that make the financial reporting value chain more efficient. In short, using XML and XBRL as accounting information tools is good management practice. The platform can speed up financial reporting and may even lead to continuous reporting in the future.

An important development has occurred within the Microsoft Office package: Microsoft Office Professional 2003 is now XML compliant. This gives accountants the power to tag information and download XML compliant data and documents directly to Microsoft Excel, without special XML software tools. The only software required is Microsoft Office 2003, Notepad, and a Web browser of the user’s choice. Once the information is downloaded, the data can be manipulated for analysis.
LITERATURE REVIEW, OBJECTIVES AND CONTRIBUTIONS OF THE STUDY

Literature Review

Although much has been written about text markup languages being applied to business, conspicuously absent from the literature is pedagogical research related to teaching markup languages. Gangolly and Tribunella have posited that semantic knowledge bases for FAS (Financial Accounting Standards) could be developed (Gangolly, Hedley, and Wong 1991; Tribunella 1998, 2003). The framework for the Gangolly system was based on hierarchical and-or trees that are very similar to taxonomies. In the Gangolly paper, the prototype system was written in Prolog code (an artificial intelligence language). Tribunella addressed the same problem using SGML (Standard Generalized Markup Language). In the Tribunella paper, SGML, the parent language of XML, was used to reduce the information overload problem associated with FAS.

Other researchers have suggested that XML-based accounting tags would facilitate Continuous Online Auditing (COA) (Kogan 1999). Under this research, a standard format would be designed for enterprise data so that it could be captured by COA systems. The format would be defined by XML. For example, tagging financial data for checking accuracy and tagging elements for automatic transaction confirmations could be achieved with XML (Vasarhelyi, Alles and Kogan 2004). In addition, data that conforms to a standard set of definitions could be audited across supply chains and value chains (Vasarhelyi 2002, 266).

It has also been posited that XML may provide a solution to information management problems at CPA (Certified Public Accounting) firms and in corporate accounting departments (Cohen and Hannon 2000; Garthwaite 2000; Hoffman, Kurt, and Koreto 1999; McCausland 2000; O’Rourke 2001; Zarowin and Hardin 2000). Under this theory, financial statements, work-papers, memos, and a wide variety of financial information could be encoded with XML tags and retrieved by conceptual category. However, before this can occur, accounting professors must begin teaching XML so their graduates can employ this technology to assist with financial applications.

A search of the literature located many applications employing XML and XBRL. Below is a partial list of some of the organizations utilizing markup languages for business related applications:

- IRS (Internal Revenue Service) Electronic Filing: The IRS plans to create an XML library of documents to facilitate the electronic filing of tax returns (Strand, McGuire, Watson, and Hoffman 2001; Karim, Gara, and Pinsker 2005b).
- SEC (Securities and Exchange Commission): The SEC now accepts electronic reports that are compliant with XML and XBRL from publicly traded companies through the EDGAR (Electronic Data Gathering And Retrieval) system (Hannon 2000, 2001; Debreceny et. al. 2005).
- AICPA (American Institute of CPAs): The AICPA strongly supports the use of XBRL which allows users to electronically publish, exchange, and analyze business information such as financial statement and general ledger information (Tie 2000).
- UK (United Kingdom) Tax Reporting: The UK government announced that XBRL will be mandatory for company tax filings after March, 2010 (Calvert 2006).
- The International Accounting Standards Committee (IASC) has taxonomies under construction for its International Financial Reporting Standards (IFRS).
- Software vendors are developing tools to make taxonomy construction easier. Companies such as Altova, Wattle Software, Digital ML, Xmetal, Oxygen XML Editor, XML Maker, EditX, Rivet Software, UBmatrix, and a host of others (including freeware) all provide excellent support for XML applications.

In spite of the rapid growth in markup technologies and applications, there are still significant challenges ahead. There is a lack of training at the college level for accounting and business students. According to Karim, Deshmukh, and Romine (2003), perceived obstacles related to integrating XML and XBRL into the accounting curriculum are displayed below.

- Lack of faculty interest, average = 3.77, SD (Standard Deviation) = 0.99
- Lack of student interest, average = 2.45, SD = 1.12
- Lack of instructional materials including textbooks, average = 3.61, SD = 0.98
- Lack of job opportunities, average = 2.12, SD = 0.99
- Lack of faculty preparedness, average = 4.33, SD = 0.94

The data is based on a survey of professors using a five point Likert scale in which one equals no obstacle and five equals a severe obstacle. As we can see from the above statistics, lack of instructional materials and lack of faculty preparedness are significant problems. This paper focuses on solving those two problems by providing a simple example of how XML can be used in conjunction with Microsoft Excel.
Learning Objectives and Contributions of this Study

In this study, we have started with a trial balance, tagged it in XML, and generated a set of financial statements in Microsoft Excel. This is a very basic application of information modeling and XML that will allow an instructor to teach the theory behind XBRL. This lab exercise does not use XBRL to tag financial statement elements. Instead, it requires students to practice the process of developing tags for financial statement elements using XML. Accordingly, the students learn the underlying theory of how XBRL tags were developed.

This simple lab exercise allows the students to learn the process and tools without being confused by excessively detailed technology and information. The learning objectives of this lab, which are listed below, are organized around Bloom’s Taxonomy of Thinking Skills:

1. Knowledge (lower order thinking skill): Recall facts and definitions related to markup languages such as XML and XBRL.
2. Comprehension (lower order thinking skill): Understand the principles of markup languages.
3. Application (higher order thinking skill): Apply XML to a financial information problem.
4. Analysis (higher order thinking skill): Identify the parts of the information domain under analysis and determine the taxonomic links between the parts.
5. Synthesis (higher order thinking skill): Reorganize the elements from the general ledger into financial statements through the use of XML.
6. Evaluation (higher order thinking skill): Judge the effectiveness of the process for compiling financial information from general ledger data.

The benefits of teaching XML are substantial. First, XBRL is an extension of XML and is used by the SEC for the EDGAR financial reporting system. It is an absolute necessity that accounting students become aware of the technical standards they will face once they are employed. Second, once a student tags a small sample of financial information and structures a spreadsheet, downloading additional information related to more complex problems is simplified because the system development procedures are understood. This paper guides the reader through a simple yet powerful example that will demonstrate the usefulness of information modeling combined with XML technology.

MODELING AND BACKGROUND INFORMATION ON MARKUP LANGUAGES

Semantic Modeling

Before an application can employ XML, the domain information must be structured and classified into a clearly organized taxonomy. Professionals as well as students must learn how to form well-structured documents. This is an information modeling problem, not a technology problem. The following quote from a landmark article in Scientific American explains the importance of information modeling and organization:

“Exceptional progress has been made in providing high-speed access to carefully defined and limited stores of records. With progress, however, has come the realization that the ultimate goal is far more difficult to achieve than had been thought earlier. It involves not only the development of techniques for storing and manipulating records but also the improvement of our understanding and simulation of the ways in which people make associations and value judgments, together with the development of more reliable methods of predicting human information needs. The problem is largely an intellectual one, not simply one of developing faster and less expensive machinery.” (Liptez 1966)

Human information needs must be systematically modeled and organized. Without the model, information is disorganized and difficult to retrieve. Hence the information, even though it is stored in a high-speed computer, is rendered useless. As Liptez (1966) suggests, the problem is not solved by machine speed, but it will be solved by understanding the information user. Therefore, semantic models of the information domain must be developed before the instance documents are coded with XML. If modeling skills are not developed by professors and students, this lack of knowledge will create a significant obstacle to using markup languages.

Taxonomies and Markup Languages

Text markup languages and tags such as XML and XBRL are organized in taxonomies. XML-based systems extract information based on a common taxonomy of tags to facilitate efficient and effective information retrieval. A taxonomy is a
hierarchical classification system. Items contained in a taxonomy are known as taxons, which are defined and classified by their attributes. Taxonomies enforce the organization and inheritance of related concepts. All child taxons inherit the attributes of parent taxons all the way back to the root (top taxon). A taxonomy leaf has no children (bottom taxon).

Physically fragmented segments of text can be linked together by conceptually based tags. When the system is queried, the appropriate data elements are aggregated into the user interface saving much research time. The stronger the taxonomy, the easier it is for companies to create instance documents and queries. In addition, XBRL enables industries to build and use their own taxonomies.

Taxonomic analysis can facilitate the efficient investigation of a domain by:

- Defining the boundaries of the domain.
- Facilitating a systematic organization of diversity within the domain.
- Mapping the relationships within the domain.
- Providing rules regarding the identification and classification of members, items, objects, or concepts in the domain.
- Resolving redundancy and reducing obsoleteness in a domain (Everett, Bobrow, Stole, Crouch, Paiva, Condoravdi, Berg and Polanyi 2002).

Taxonomies lay the foundation for SGML and XML systems. Without a stable foundation, the entire system fails the user. The logical model or taxonomy is expressed as XML or SGML code and is called a DTD (document type definition) (Rubinsky and Maloney 1997; Travis and Waldt 1995). The DTD is the program code that defines the tags that will be used to markup an information domain.

**History and Application of Markup Languages**

Luca Pacioli would surely be amazed by the Internet and its vast web of information and technology. However, the current financial reporting process has not fully harnessed the power of the information superhighway. Today’s reporting processes are time consuming and laborious for many firms. Financial documents and managerial reports are very often reworked on an ad hoc and non-repeatable basis. This can cause delays in the financial reporting process and slow down the production of financial statements, tax returns, regulatory reports and managerial information. This is a very labor intensive process that can be streamlined with markup technologies.

The amount of financial information sent over the Internet with SGML, XML and XBRL encoded documents is increasing rapidly. XML was created with the Standard Generalized Markup Language (SGML). SGML is a meta-language developed specifically to solve problems associated with text-based systems. It is a flexible language that allows users to create text-tagging languages specifically targeted to a particular profession or domain of knowledge. Accordingly, SGML research and applications are extremely diverse. The Extensible Markup Language (XML) is a subset of SGML that provides 80% of the functionality and only 20% of the complexity (Rezae, Hoffman and Marks 2001).

The most significant XML implementation related to accounting documents is XBRL. Formally known as XFRML (Extensible Financial Reporting Markup Language), this accounting markup standard can help make financial applications such as consolidations, SEC filings, ERP systems, and posting financial information to the Web more efficient (Strand, McGuire, Watson and Hoffman 2001).

XBRL is a text and data markup language similar to HTML (Hyper Text Markup Language). XBRL allows content such as financial statements to be electronically labeled with concept based tags. It is an open standard, free of license fees. In addition, it is extensible so companies can modify the basic XBRL taxonomy to fit their needs.

All publicly traded companies in the US must file financial reports with the SEC such as the 10-K Annual Report, 10-Q Quarterly Report and 8-K Material Corporate Events (e.g., the company changes auditors). According to SEC rule 33-8496, the SEC is now accepting the filing of financial statements and other information using XBRL. For now the program is voluntary, however, it may be mandatory in the future. Therefore, accountants should be aware of XBRL and the EDGAR (Electronic Data Gathering And Retrieval) system.

EDGAR is the gigantic electronic database that contains the SEC filings of thousands of companies. EDGAR contains reports in ASCII (American Standard Code for Information Interchange), HTML, and now XBRL. In the future, more of the information contained in the database will be XBRL compliant.
TEACHING NOTES AND IMPLEMENTATION GUIDELINES

Teaching Notes

The goal of this paper is to create a classroom exercise that will help students understand how markup languages such as XML and XBRL fit into the financial information reporting process. We do this by giving the students a hands-on experience with tagging financial information and compiling it in the form of financial statements. A reporting system is not only required to produce financial statements, but it must provide a wide variety of reports to internal and external users. Managers, regulators, taxing authorities, and other stakeholders demand a multitude of reports. Once data is defined with XML, these reports can be generated more efficiently. Therefore, this application should demonstrate to the students the underlying theory of how to structure financial data so that it can be used to fulfill multiple reporting requirements.

We have used this in class and it was very well received. Students seem to sincerely understand XML and XBRL after the two hour exercise. This exercise makes an excellent computer lab as part of an AIS (Accounting Information Systems) class. This exercise can also be expanded into a much longer case study, showing how to start and maintain the file as multiple-period financial reports are generated. Some enterprise system packages such as SAP and MS Great Plains will output XML compliant data from the general ledger. Therefore, the Notepad file we created manually would be automatically generated by the accounting system’s general ledger and financial reporting module. This makes it very easy to produce spreadsheets that do multiple applications with the data, such as financial statement analysis, budget variances, and benchmarking.

Implementation through Systems Planning and Analysis

In the rapidly changing world in which we live, organizations are constantly changing, replacing, and updating their information systems. This creates very complex and expensive projects. Managers employ a variety of techniques to maintain control of system projects. One of the more common techniques is a process known as the systems development life cycle.

The teaching process employed in this paper follows the systems development life cycle (SDLC). The SDLC adds structure to the design and implementation of a new system (Romney and Steinbart 2006). This five stage process has been used to manage the complexity of information technology for over 30 years. The methodology followed in this classroom lab is listed below:

1. System Analysis: Identify and describe the domain under analysis. In this exercise, the domain information is a trial balance (see Exhibit 1).
2. Conceptual Design: Identify and define the data elements within the domain. The data elements will be each account balance within the trial balance. Next we classify the elements into a taxonomy schema. In this exercise, the taxonomy is used to classify the accounts within the general ledger (see Exhibit 2).
3. Physical Design: Create a tag for each element. Tags should be meaningful titles, as they can also be used as labels. Tags are usually defined in a DTD. However, some software such as Microsoft Excel will back into a DTD from a well structured XML compliant document.
4. Implementation: Create a file in Notebook to tag the data. Then markup the data with XML tags (see Exhibit 3). Structure the spreadsheet to accept the XML compliant data (see Exhibits 4, 5, 6 and 7).
5. Operation: Load the data into the Microsoft Excel Spreadsheet and use the system (see Exhibits 8, 9 and 10).
DETAILED INSTRUCTIONS FOR THE COMPUTER LAB

In this exercise, we use a simple trial balance from the fictitious company Dynamite Tax Services, Inc. This trial balance is included in Exhibit 1. Students should be instructed to examine the trial balance to gain an understanding of the information domain under analysis.

Exhibit 1: The Information Domain; A Trial Balance

<table>
<thead>
<tr>
<th>No.</th>
<th>Account Title</th>
<th>Debit</th>
<th>Credit</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>Cash</td>
<td>12,000</td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>Accounts Receivable</td>
<td></td>
<td>15,000</td>
</tr>
<tr>
<td>201</td>
<td>Land</td>
<td>50,000</td>
<td></td>
</tr>
<tr>
<td>202</td>
<td>Building</td>
<td>250,000</td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>Accumulated Depreciation</td>
<td></td>
<td>20,000</td>
</tr>
<tr>
<td>301</td>
<td>Accounts Payable</td>
<td></td>
<td>5,000</td>
</tr>
<tr>
<td>302</td>
<td>Wages Payable</td>
<td></td>
<td>3,000</td>
</tr>
<tr>
<td>303</td>
<td>Unearned Revenue</td>
<td></td>
<td>2,000</td>
</tr>
<tr>
<td>401</td>
<td>Bonds Payable, 8%</td>
<td></td>
<td>50,000</td>
</tr>
<tr>
<td>402</td>
<td>Mortgage Payable, 5%</td>
<td></td>
<td>100,000</td>
</tr>
<tr>
<td>501</td>
<td>Common Stock, $5 PV</td>
<td></td>
<td>50,000</td>
</tr>
<tr>
<td>502</td>
<td>Additional PIC in Excess of Par</td>
<td></td>
<td>20,000</td>
</tr>
<tr>
<td>503</td>
<td>Retained Earnings</td>
<td></td>
<td>67,000</td>
</tr>
<tr>
<td>504</td>
<td>Dividends</td>
<td></td>
<td>5,000</td>
</tr>
<tr>
<td>601</td>
<td>Service Revenue</td>
<td>75,000</td>
<td></td>
</tr>
<tr>
<td>701</td>
<td>Wages Expense</td>
<td></td>
<td>30,000</td>
</tr>
<tr>
<td>702</td>
<td>Office Supplies Expense</td>
<td></td>
<td>7,000</td>
</tr>
<tr>
<td>703</td>
<td>Insurance Expense</td>
<td></td>
<td>5,000</td>
</tr>
<tr>
<td>704</td>
<td>Utilities Expense</td>
<td></td>
<td>3,000</td>
</tr>
<tr>
<td>705</td>
<td>Maintenance Expense</td>
<td></td>
<td>6,000</td>
</tr>
<tr>
<td>706</td>
<td>Interest Expense</td>
<td></td>
<td>9,000</td>
</tr>
<tr>
<td></td>
<td>Totals</td>
<td>392,000</td>
<td>392,000</td>
</tr>
</tbody>
</table>

The next step is to design a taxonomy that will conform to the financial statements we want to create. The financial statements we created were: An income statement, a statement of retained earnings, and a balance sheet. We did not create a statement of cash flows to keep the exercise simple. Our taxonomy is included in Exhibit 2.
Once a taxonomy has been developed, it is now time to tag the financial statement elements. To tag the items with XML, open a Notepad file. Notepad can be found in Windows under Accessories. The tag names created should be meaningful labels that could be used in your financial statements. Microsoft Excel will allow the user to use the tag names as labels. Tag names must be XML compliant. In order to be XML compliant, tag names must not include spaces and commas. Spaces and commas will not allow the XML file to parse properly. Tag names can include hyphens and underscores, which could be used for spaces.

Tag names need to be included within brackets, which are the “greater than” and “less than” characters on the keyboard. When starting a tag, the name simply needs to be included in brackets. Tag names must be ended by using a bracket, the / character, the element name, and an ending bracket. For example the starting and ending tags for cash would be <cash> and </cash>.

When creating the tag names, the logic in the taxonomy should be followed. Parent and child taxon inheritance properties also must be adhered to. Data can only be included in one child. To allow an easier way to follow the XML markup, consider indenting each child and grandchild, under a parent or grandparent tag. Please refer to our Exhibit 3 of XML compliant data. When entering the financial data, it is best to not use any commas, as this appears to be unstable within Microsoft Excel.

Once all of the data elements have been tagged, save the file as a Notepad file so future changes can be made. The Notepad file will also need to be saved as a XML file. Using the “Save As” function on the File menu, save the Notepad as a separate file.
with a XML extension. The XML extension will have to be added manually in the “Save As” dialog box. Then open the XML file within a Web browser. Double clicking on the file should automatically open it within a browser. If the file parses correctly, it will open like a Web-page. If it does not open, an error message will appear and the user must go back to debug the XML tags in the Notepad file.

Students tend to make the same common errors. For example, some of the more common errors we have seen are:

- The opening tag name and closing tag name are not the same, due to typos.
- Closing tag name does not include a / character.
- Tag names include improper characters, such as spaces or commas.

All errors will need to be corrected in order to open the file properly within a Web browser. As with most computer languages, it is helpful to the debugging process to write the code in small pieces and then debug that piece before expanding the program. In our example, we tagged the assets first, then the liabilities, and finally the equity (including the income statement accounts).

**Exhibit 3: XML Compliant Data**

```xml
<General_Ledger>
  <CoName>Dynamite Tax Services, Inc.</CoName>
  <Date>December 31, 2005</Date>
  <Assets>
    <Current_Assets>
      <Cash>12000</Cash>
      <Accounts_Receivable>15000</Accounts_Receivable>
    </Current_Assets>
    <Noncurrent_Assets>
      <Land>50000</Land>
      <Building>250000</Building>
    </Noncurrent_Assets>
  </Assets>
  <Liabilities>
    <Current_Liabilities>
      <Accounts_Payable>5000</Accounts_Payable>
      <Wages_Payable>3000</Wages_Payable>
      <Unearned_Revenue>2000</Unearned_Revenue>
    </Current_Liabilities>
    <Noncurrent_Liabilities>
      <Bonds_Payable>50000</Bonds_Payable>
      <Mortgage_Payable>100000</Mortgage_Payable>
    </Noncurrent_Liabilities>
  </Liabilities>
  <Equity>
    <Common_Stock>50000</Common_Stock>
    <Additional_Paid-in_Capital>20000</Additional_Paid-in_Capital>
    <Retained_Earnings>
      <Beginning_Retained_Earnings>67000</Beginning_Retained_Earnings>
      <Dividends>5000</Dividends>
      <Service_Revenue>75000</Service_Revenue>
      <Expenses>
        <Wages_Expense>30000</Wages_Expense>
        <Office_Supplies_Expense>7000</Office_Supplies_Expense>
        <Insurance_Expense>5000</Insurance_Expense>
        <Utilities_Expense>3000</Utilities_Expense>
        <Maintenance_Expense>6000</Maintenance_Expense>
        <Interest_Expense>9000</Interest_Expense>
      </Expenses>
    </Retained_Earnings>
  </Equity>
</General_Ledger>
```

Once the data has been properly tagged within XML, it is time to populate the Microsoft Excel spreadsheet with the tagged data. The first step is to open an Microsoft Excel file. Next, open the XML file from within Microsoft Excel, by selecting open from the file menu. Microsoft Excel will recognize the file as an XML file and an “Open XML” dialog box will appear. Select “Use the XML Source task pane” and select “OK”. See Exhibit 4 for a screenshot of the dialog box in Microsoft Excel.
The user then receives a dialog box that states: “The specified XML source does not refer to a schema. Microsoft Excel will create a schema based on the XML source data.” The user may simply click “OK”. The user can also check the box: “In the future, do not show this message.” See Exhibit 5 for a screenshot of the dialog box in Microsoft Excel.

The tagged data will be shown in the sidebar of the Microsoft Excel spreadsheet. Please refer to Exhibit 6 for a screenshot of the sidebar containing the XML source. Microsoft Excel will create the taxonomy of tags from a well structured XML compliant document. Therefore, no DTD will need to be programmed, which is a significant savings of labor.
Exhibit 6: XML Source Map in a Microsoft Excel Spreadsheet

Once all of the XML tags are shown in the source task pane, it is a simple process to map the data elements to the cells. Two methods can be used to map the data: (1) click on the data element and drag it to the target cell or (2) select the target cell on the spreadsheet and double click on the data item shown in the source task pane.

Data elements can only be mapped to one cell. One cannot have the same element mapped to two different cells. If the data is needed in more than one cell, simply add a cell formula, such as +A2 to put the data in a second cell. It may be helpful to show the actual numerical data in the XML Source task pane while mapping. To do so, simply click the options button on the bottom of the XML Source task pane and select Preview Data in Source Pane. Once mapped, the cell has a thin blue border, and in the source task pane, the data element is bolded to indicate this element has already been mapped. The data will appear once the import function is complete.

XML tags can be used as labels for the data. To use XML tags as labels, check the “My data has headings box” in the options menu under the XML source task pane. This box is usually checked as a default. Once the data element has been mapped, the label button appears at the bottom-right corner of the cell. If one clicks on that button, one can choose three options: (1) My Data Already Has a Heading (2) Place XML Heading to the Left and (3) Place XML Heading Above. Choosing option (2) or (3) will automatically place the XML tag onto the spreadsheet. See Exhibit 7 for a screenshot of these options.
Once the spreadsheet is mapped and labeled, it is time to import the data. We chose to map the data and add cell formulas for subtotals, and calculations such as net income, prior to importing the data. However, the formulas could be added after importing the data. Formatting can also be done after importing the data.

To import the data, simply place the cursor in any cell. Go to the Data Menu, click on XML and click Import. See Exhibit 8 for a screenshot of the XML Data Import Menu in Microsoft Excel.

A XML Import dialog box will appear. Select the file with the XML data in it and click the import button. The data will be pulled into the spreadsheet cells. See Exhibit 9 for a screenshot of the XML Data Import dialog box in Microsoft Excel.
See Exhibit 10 for a screenshot of an Microsoft Excel Spreadsheet with Imported XML Compliant Data. One can now format the data with commas and decimals as well as place lines and borders as appropriate. Lines, borders, commas and decimals can be formatted prior to the importing of the data. As a further extension, one can set up data analysis, such as financial statement ratios calculated from the imported data.

Exhibit 10: Microsoft Excel Spreadsheet with Imported XML Compliant Data
SUMMARY, CONCLUSIONS AND FUTURE RESEARCH

Summary and Conclusions

Since before the time of Luca Pacioli, financial information has been contained in documents. It is now possible to identify and store financial information electronically. Markup languages such as XML and XBRL allow us to tag bits (grains, elements) of financial data so that it can be used and reused in a variety of reports.

There are concrete business purposes for using XML tagging that are based on cost savings, efficiency and accuracy. There is a consensus and a great deal of investment within government agencies, industries, countries and regulatory partners. Many organizations are working collaboratively to design taxonomies for regulators such as the IASC, SEC, and IRS. XBRL now has standards, specifications, many applications, and market acceptance. These taxonomies could be used to generate a large amount of comparable data by downloading multiple financial statements once tagged.

The technology is in place but the education community must produce materials to train students how to effectively use the current markup languages. This paper can be used by an AIS professor to demonstrate in a computer lab how to tag financial information. The lab exercise has been successfully employed in a classroom setting.

Limitations of this Study and Suggested Future Research

The method and exercise presented in this paper is a good start. However, more complex case studies need to be written. For example, this exercise could be expanded to include a statement of cash flow or a financial analysis section. Another classroom application could be a case where the students are required to load the XBRL taxonomy and analyze a company’s XML compliant financial statements. Companies such as Microsoft, 3M, Dow Chemical, Xerox and XM Satellite Radio have already submitted XML compliant SEC reports (SEC, 2006). Finally, open-ended case studies could be employed where students select an accounting information domain (such as a budget) and build the taxonomy as well as create an XML compliant document related to the domain.

After educational methods and materials are developed, we will need to test the effectiveness of the results with survey questions and exam questions. See the Appendix for an example of a survey instrument organized around Bloom’s taxonomy of thinking skills. With the rapid deployment of markup technologies well under way, the education community must start producing training materials to educate students so they are ready for 21st century technologies.

REFERENCES

Appendix: Survey Instrument Using Bloom’s Taxonomy of Thinking Skills

Participant Demographic Information

1. Education Level:
   - 1 = Freshman
   - 2 = Sophomore
   - 3 = Junior
   - 4 = Senior
   - 5 = Graduate Student

2. Major:
   - 1 = Accounting
   - 2 = Business
   - 3 = Information Systems
   - 4 = Computer/Information Science
   - 5 = Other

3. Age:
   - 1 = 17 to 22
   - 2 = 23 to 28
   - 3 = 29 to 34
   - 4 = 35 to 40
   - 5 = 41 or greater

4. Gender:
   - 1 = Male
   - 2 = Female

5. How many years of professional work experience do you have?
   - 1 = 0
   - 2 = 1 to 3
   - 3 = 4 to 6
   - 4 = 7 to 9
   - 5 = 10 or greater

Educational Impact of the Exercise

1. This exercise helped me to recall facts and definitions related to markup languages such as XML and XBRL.
   - 1 = Strongly disagree
   - 2 = Disagree
   - 3 = Neutral
   - 4 = Agree
   - 5 = Strongly agree

2. This exercise helped me to understand the principles of markup languages such as XML and XBRL.
   - 1 = Strongly disagree
   - 2 = Disagree
   - 3 = Neutral
   - 4 = Agree
   - 5 = Strongly agree

3. This exercise helped me to understand how to apply XML to financial information problems.
   - 1 = Strongly disagree
   - 2 = Disagree
   - 3 = Neutral
   - 4 = Agree
   - 5 = Strongly agree

4. This exercise helped me identify the parts of a financial report and determine the taxonomic links between the parts.
   - 1 = Strongly disagree
   - 2 = Disagree
   - 3 = Neutral
   - 4 = Agree
   - 5 = Strongly agree

5. This exercise helped me to reorganize the elements from a general ledger or trial balance and organize those elements into financial reports.
   - 1 = Strongly disagree
   - 2 = Disagree
   - 3 = Neutral
   - 4 = Agree
   - 5 = Strongly agree

6. This exercise helped me to judge the effectiveness of the process for compiling financial information from general ledger data.
   - 1 = Strongly disagree
   - 2 = Disagree
   - 3 = Neutral
   - 4 = Agree
   - 5 = Strongly agree

Comments: