Extensible Business Reporting Language (XBRL) Version 0.9 Specification

4 April 2000

Editors:

Walter Hamscher, PricewaterhouseCoopers LLP

David Vun Kannon, KPMG Consulting Inc.

AICPA Working Group on XBRL

Specification Group

Co-Chairs:

Walter Hamscher, PricewaterhouseCoopers LLP

David Vun Kannon, KPMG Consulting Inc.

This version:

http://www.xbrl.org/TR/00-04-04/xbrl-00-04-04.doc (in Word)

http://www.xbrl.org/TR/00-04-04/xbrl-00-04-04.pdf (in PDF)

with separate provision of the DTD, Schema and US GAAP C&I taxonomy described herein. All components, along with non-normative samples and certain auxiliary DTDs are available in a single Zip format archive.

Abstract

XBRL 0.9 is an exposure draft of the specification for the eXtensible Financial Reporting Markup Language. This document proposes XML elements and attributes that can be used to express information used in the creation, exchange and comparison of financial reporting. XBRL 0.9 consists of a core language of XML elements and attributes used in document instances as well as a language used to define new elements and taxonomies of elements referred to in document instances.

Status of this document

This is a working draft of XBRL 0.9 for review by the public and by members of the eXtensible Financial Reporting Markup Language Working Group (WG).

It has been reviewed by the WG, and the WG has agreed to its publication. The WG believes this draft to be near "feature-complete": the functionality included here is substantially complete and is expected to be stable. We do not expect to add major new functionality, or to make major changes to the functionality described in this draft. Some sections of the draft (in particular those on entity naming), and some aspects of the design (in particular details of the references to footnotes and other document structuring forms), on the other hand, may be substantially revised.

The WG expects to spend April 2000 through June 2000 working out details, clarifying points of uncertainty that arise in the review of this draft, cleaning up inconsistencies, reviewing the design of the syntax, and making editorial improvements.

Following that period of review, it is the WG's intent to issue a Last Call for Review sometime during June, 2000, and to freeze the specification for publication in June, 2000. This schedule may vary, depending on the comments of the public and of other working groups on this draft. Such comments are instrumental in the WG's deliberations, and we encourage readers to review the draft and send comments to xfrml-public@egroups.com (archive).

Although the Working Group does not anticipate further substantial changes to the functionality described here, this is still a working draft, subject to change based on experience and on comment by the public and other working groups. The present version should be implemented only by those interested in providing a check on its design or by those preparing for an implementation of the final recommendation. *The WG will not allow early implementation to constrain its ability to make changes to this specification prior to final release.*

Table of contents

1 Introduction

XBRL allows software vendors, programmers and end users who adopt it as a specification to enhance the creation, exchange, and comparison of financial reporting information. Financial reporting includes, but is not limited to, financial statements, general ledger information and regulatory filings such as annual and quarterly financial statements. XBRL is a component of the XBRL framework.

1.1 Documentation Conventions

This Working Draft document will eventually be produced using an [XML] DTD and an [XSLT] stylesheet.

The following highlighting is used to present technical material in this document:

```
XML Declarations
```

The following highlighting is used for non-normative commentary in this document:

Example
A non-normative example illustrating use of the XBRL language, or a related instance.
`<schema name="http://www.muzmo.com/XMLSchema/1.0/mySchema" >`
And an explanation of the example.

[Issue]: A recorded issue.

Ed. Note: Notes from the editors to themselves or the Working Group.

NOTE: General comments directed to all readers.

1.2 Purpose

The purpose of XBRL is documented in the XBRL Functional Specification, dated March 3rd, 2000 (originally titled "XFRML Functional Specification). The main purposes are recapitulated here for convenience.

The XBRL specification is meant to maximize benefits to all stakeholders that will use its specifications. The specification is intended to benefit three categories of users: financial information preparers, intermediaries in the preparation and distribution process, and users of financial information. There is also a fourth category of beneficiary, the vendors who supply software and services to one or more of these three types of user. The overall intention is to balance the needs of these groups creating a product that provides benefits to all groups.

The needs of end users of financial information will generally have precedence over other needs when it is necessary to make specification design decisions that may be perceived as benefiting one community at the possible expense of another.

XBRL must be extensible, and will support the framework without any major revisions.

XBRL is intended to improve the financial statement product. It should only comply with, not change or set new accounting standards. However, XBRL should facilitate possible changes in financial reporting over the long term.

XBRL will provide users with a standard format in which to *prepare* financial reports that can be subsequently presented in a variety of ways. XBRL will provide users with a standard format in which financial information can be *exchanged* between different software applications. XBRL will permit the automated, efficient and reliable *extraction* of financial information by software applications. XBRL will facilitate the automated comparison of financial information, accounting policies, notes to financial statements between companies, and other items which users may wish make comparisons that today are performed manually.

XBRL should facilitate "drill down" to detailed information, authoritative literature, audit and accounting working papers. XBRL should include specifications for as much information about the reporting entity as may be relevant and useful to the process of financial reporting and the interpretation of the information.

XBRL should support international accounting standards and languages other than the American dialect of English.

XBRL should be extensible by any adopter to increase its breadth of applicability, and its design should encourage reuse via incremental extensions. XBRL should specify the format of information that would be reasonably expected in an electronic format for securities filings by public entities. XBRL should facilitate business reporting in the long term, and should not be limited to financial and accounting reporting.

XBRL focuses on the genuine information needs of the user and adheres to the spirit of reporting standards that deprecate the use of bold, italics, and other stylistic techniques that may be used to distract from the true and fair presentation of financial results. Therefore, there is no functional requirement that XBRL documents need to support any particular text formatting conventions.

1.3 Relationship to Other Work

XBRL uses several World Wide Web consortium (w3c) recommendations, XML 1.0, XML Namespaces, and refers indirectly to XSL. It also relies extensively on the 25 February 2000 working draft of XML Schema.

Discussions have taken place with other bodies issuing XML specifications in the financial arena, including OAG (Open Applications Group), OMG (Object Management Group), FpML (Financial Products Markup Language), finXML (Financial XML), OFX/IFX (Open Financial Exchange) and ebXML (e-Business XML). The scope of XBRL includes financial reporting and contemplates extensive detail in the representation and use of accounting conventions, which distinguishes it from these other efforts. Also, the design of XBRL is deliberately drawn so as to allow the embedding of isolated XBRL items into other XML documents, which is key to future interoperability with these other specifications.

1.4 Terminology

The terminology used in XBRL frequently overlaps with terminology from other fields, and the following short list is provided to reduce the possibility of ambiguity and confusion.

item	A fact or facts reported within a given period of time about a given business entity. Corresponds to an XML element "item" in XBRL.
taxonomy	An XML Schema that defines new elements each corresponding to a concept that can be referenced in XBRL documents. XBRL taxonomies can be regarded as extensions of XML Schema.
entity	A business entity, the subject of XBRL items. Where the XML/SGML concept of syntactic "entity" is meant, this will be pointed out
statement	Text containing a collection of items that concern one or more entities during one or more time periods.
period	An instant or a duration of time. In business reporting, financial numbers and other facts are reported "as of" an instant or for a period of a certain duration. Items that report on instants and durations are both common.
element	An XML element type, in the sense used in XML 1.0 and XML Schema.
instance	An XML document containing XBRL elements that together constitute one or more statements. The financial statements of IBM, expressed in XBRL, would be an instance. So

	would an HTML file that had various XBRL items embedded in it.
may	Conforming documents and consuming applications are permitted to but need not behave as described.
must	Conforming documents and consuming applicaitons are required to behave as described; otherwise they are in error.
error	A violation of the rules of this specification; results are undefined. Conforming software may detect and report an error and may recover from it.
fatal error	An error which a consuming application must detect and report. After encountering a fatal error, the application may continue processing the data to search for further errors and may report such errors. In order to support correction of errors, the processor may make unprocessed data from the document (with intermingled character data and markup) available to the application. Once a fatal error is detected, however, the processor must not continue normal processing (*i.e.*, it must not continue to pass character data and information about the document's contents to the application in the normal way).
at user option	Conforming software may or must (depending on the modal verb in the sentence) behave as described; if it does, it must provide users a means to enable or disable the behavior described.

2 XBRL Framework

The main ideas in the XBRL conceptual framework are *items, elements, taxonomies* and *statements.* This section bears careful reading, since these terms are used in a precise way within XBRL.

Items. In the XBRL framework, the most fundamental concept is that of the *item.* An *item* is meant to correspond to a fact or facts—often but not necessarily numeric facts—that are being reported with respect to a given period of time about a given business entity. For example, the fact that the company whose ticker symbol is SAMP reported revenues of $7m for the year 1998 is an item. This is an example of a numeric item. An example of a non-numeric item would be a paragraph of text describing the principles of consolidation used to combine reports from the subsidiaries of SAMP. Although the latter is not numeric, this is nevertheless a set of facts being reported with respect to a given period of time (1998) about a given business entity (SAMP).

XBRL defines a syntax in which many different kinds of facts can be represented and their context defined in such a way so that software applications can efficiently and reliably find, extract, and interpret relevant items in their appropriate context.

Elements and Taxonomies. An equally important part of the XBRL framework is the concept of an *element* and its relationships to other elements within a *taxonomy.* In XBRL, the notion of a taxonomy element corresponds exactly to the notion of an element within an XML Schema [SCHEMA-1]. An important taxonomy for the purposes of the current specification is the particular taxonomy consisting of elements that correspond to well defined concepts within the US Generally Accepted Accounting Principles (GAAP) when those principles are applied to Commercial and Industrial (C&I) companies. For example, the concept of "allowance for bad debts," and the fact that it is subtracted from "accounts receivable" to arrive at a figure for "net accounts receivable" are different parts of that particular taxonomy.

Although any given item can only refer to one taxonomy, within any given XML document any number of XBRL items can refer to any number of taxonomies, and taxonomies can be composed together to extend other taxonomies. For example, although the current release of the XBRL specification provides a particular taxonomy, any given XML document may refer to a taxonomy that defines additional terms and relationships. Suppose, for example, that a significant portion of expenses are (in a hospital, say) "physician salaries"; because that term does not exist in the US GAAP C&I taxonomy as such, a new (small) taxonomy would be defined which defined the term "physician salaries" and referred to the US GAAP taxonomy so as to relate this to the concept of "expenses" that already exists there.

Statements. In XBRL, a *statement* is a set of related *items* that can appear in any order and that in fact can be interspersed among other text and elements in any XML document. There is, therefore, no "XBRL document type" as such. It is possible in principle to embed an XBRL item in *any* document, such as a press release that is otherwise formatted in HTML. The intention of XBRL instance documents is just the transmission of some set of financial facts. There is no constraint on how much or how little they contain. A single item can be a valid XBRL document, for instance when all the information being conveyed is (say) what Cost of Goods Sold was last quarter. An XBRL document can be a database dump. It can be anything in between. This provides a great deal of flexibility and is meant specifically to achieve the goals of allowing XBRL to be reused within other specifications and for application software to be able to most easily extract financial data from otherwise arbitrarily formatted documents. It is expected that for most uses of XBRL, in fact many instance documents will be created that consist almost exclusively of items.

3 Syntax of Instance Documents

The syntax chosen for XBRL instance documents corresponds to a recommended syntax for serializing graphs of data in XML [CANONICAL]. XBRL uses this canonical syntax to exploit the features of XML attributes, specifically their order independence, irredundancy, ability to accommodate enumerated types, and the ability to have default (#IMPLIED) values. In XBRL there are relatively few XML elements, but there is a rich set of attributes that are applicable to most elements, and furthermore, the allowed values for those attributes are elements within one or more taxonomies.

The core syntax for statements is defined using an XML DTD. The elements defined there are the item, label, and group. The item and group elements have the same set of attributes, which are in some sense the more important part of the XBRL representation. The set of attributes is defined as follows.

```
<!ENTITY % att_AttributeHolder  "
  id                  CDATA       #IMPLIED
  period              CDATA       #IMPLIED
  schemaLocation      CDATA       #IMPLIED
  scaleFactor         CDATA       #IMPLIED
  precision           CDATA       #IMPLIED
  type                CDATA       #IMPLIED
  unit                CDATA       #IMPLIED
  entity              CDATA       #IMPLIED
  decimalPattern      CDATA       #IMPLIED
  formatName          CDATA       #IMPLIED
">
```

Each attribute is described separately below.

3.1 id

This attribute is not required, since the use of XPointer is encouraged for referring to specific elements in an XBRL instance document. The content must start with an alpha character.

Examples:

```
C2424
```

3.2 period

Every item applies to a particular instant or duration. This attribute uses the ISO 8601 date representation. A duration is a pair of dates separated by a solidus (/). See http://www.iso.ch/markete/8601.pdf for authoritative definitions; see http://www.w3.org/TR/NOTE-datetime for a summary.

`Example`	Meaning
`1999`	The calendar year 1999
`2000-04`	The month of April 2000

`1999-05-31`	The day 31 May 1999
P1Y2M3DT10H30M12.3S	1 year, 2 months, 3 days, 10 hours, 30 minutes, 12.3 seconds.
`P3M/2000-03-25`	The three months ended 25 March 2000.
`2000-04-01/P91D`	The thirteen weeks beginning 1 April 2000.
`1999-12-29/2000-03-27`	From December 29, 1999 through March 27, 2000.
`2000-01/2000-03`	The first three months (first quarter) of 2000.
`P1Y/1999-05-31`	The full year ended 31 May 1999.
`P3M/1999-05-31`	The quarter ended 31 May 1999.

Note: At http://www.w3.org/TR/xmlschema-2#timeDuration, XML Schema defines a timeDuration type. We have chosen not to follow their lead, but rather to retain a single concept of "period" and not to factor the period attribute into different attributes like "startDate" "endDate" and/or "duration" and thereby get (*e.g.*) duration="P91D" end="2000-04-01" to mean "the thirteen weeks ended 1 April 2000."

When dealing with financial information, all of the items can generally be categorized as "as-of" or "point in time" measures, such as Assets, Cash, Liabilities; or are "period" measures of aggregate activity during a defined period, such as Revenues and Expenses. As a general rule, the semantics of an item with an "as-of" measure is that it is whatever balance appeared at the *end* of the period in question.

Since any date by itself is can be interpreted as a period (*e.g.*, "1999-04" refers to all the days in the entire month of April) this leads to some subtleties. If a financial report has its year end on April 30th 1999, a producing application that indicates only that the period is "1999" is conveying incorrect information to consuming applications; it should specify at least the period "P1Y/1999-04" which means "one year, whose end coincides with the end of April".

Robust consuming applications will *not* assume that the duration of an item is related to the date specified, in other words, just because period="2001-05" it does not mean that it is referring to a one-month period.

3.3 entity

An entity specifies a system for identifying business entities and a particular identifier within that system. A business entity does not have to be a full corporate entity; it could be a subsidiary, a division, even an individual: any organization for which there is a financial statement. The entity is a QName so as to provide a framework for referencing naming authorities. It does not imply that the AICPA is a naming authority for business entities.

`Example`	Meaning
`SAMP`	Some entity known only as SAMP within the current namespace.
`NASDAQ:SAMP`	The company with NASDAQ ticker symbol SAMP.
`DUNS:0236503276`	The company or subsidiary with DUNS number 0236503276.
`CUSIP:41009876AB`	The entity with CUSIP number 41009876AB (e.g. a mutual fund).
`URI:www.w3c.org`	The non profit organization owning the URI www.w3c.org.

3.4 type

The type attribute provides the name of an element within a taxonomy. Its purpose is to facilitate comparison of information contained in different instance documents. A convention followed in the

AICPA USGAAP C&I taxonomy is that the name of a type is is a dot-separated pair of camel-case identifiers representing a human readable name for the concept and its parent.

The reason for the "parent.child" naming convention is that within a taxonomy, it is important for an element name to be unique. A single name such as "NetIncome" is inadequate because it could appear at multiple points in a taxonomy. Adopting the (parent.child) naming convention helps, but still turns out to be no guarantee. An extreme solution, which was discarded early in the design, would be to use a number. For further discussion see [Unique Elements].

`Example`	Meaning
`aicpa:currentAssets.receivables`	Receivables, in the AICPA namespace
`aicpa:accountantsReport.reportingMethod`	Reporting Method, etc.
`aicpa:notesToFinancialStatements.goingConcern`	A "Going Concern" note, etc.

Note that the type is a QName. The type attribute content must contain the correct namespace prefix, and all namespaces used in attributes must be declared in the document instance. However, it is still necessary to have a schemaLocation attribute to attach the namespaces to the right files.

Note that an element name string like "parent.child" remains merely a string, the dot is not recognized as a special character by XML, it does not have any object oriented implications, etc. It could be a "@" or "/" for all that it matters, and at one point in the design it was a "/". The colon ":" by contrast is signficant because it marks the end of a namespace. Therefore, instance documents containing type attributes whose content looks like "parent.namespace:child" are almost certainly the result of processing errors.

3.5 schemaLocation

The schemaLocation attribute is used to hold the base URI of the taxonomy which includes the concept of which the item is an instance.

Note that the way that namespaces are defined in XML, there is no guarantee that a URI with which a namespace is associated can be dereferenced to something useful. For example, the attribute content `xmlns:NASDAQ="http://www.nasdaq.com/XBRL/ticker"` does not imply that any such URI actually points to any service having to do with ticker symbol lookup. XML Schema defines the schemaLocation attribute (http://www.w3.org/TR/xmlschema-1/#xsi:schemaLocation) that can be used in a document to provide hints as to the physical location of schema documents to be used for validation. There is further discussion of this in the XML Schema Primer [SCHEMA-0]. Because this is exactly the purpose intended for this attribute in XBRL, the same attribute name has been used.

`Example`	Meaning
`http://www.iasc.org/xbrl/airline/00-07-07`	A taxonomy for the airline industry conforming to IASC guidelines.
`http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04`	The AICPA's US GAAP taxonomy for commercial and industrial companies.
`http://www.xbrl.org/cica/canada/media/00-06-02`	A Canadian Institute of Chartered Accountants' taxonomy for media companies.

By publishing a taxonomy structure for US GAAP, the AICPA hopes to facilitate the comparability of data from many sources. However, creators of XBRL data may refer to specific authoritative sources via the schemaLocation attribute, rather than defining the AICPA as the only source for taxonomy. Business entities, governments, software vendors, standards bodies and auditors can all create taxonomic resources that are publicly referenceable. The voluntary extension and refinement of published taxonomies will allow for the flexibility in reporting concepts that most users of XBRL require, especially in the international arena.

While not strictly necessary, the existence of this attribute separate from the type attribute leads to considerably cleaner (shorter and more readable) files. Ideally, the namespace declarations which are referenced by the prefix parts of the Qualified Names of the type attribute content would be sufficient to connect that content to the schema document. The schemaLocation attribute is used because it does not seem possible to enforce this desired behavior through namespaces.

3.6 unit

Unit specifies the standard which is relevant to the measurement. It is expected that most measurements will be monetary measurements. ISO 4217 standard currency designation is required for the units attribute in such a case. (http://www.iso.ch/cate/d23132.html) Dimensional measurements should be given in the SI system. Pure numbers and counts of people, shares and the like can be specified as quantities. Enumerations (ENUM) depend on the taxonomy in force for the item's concept to specify the datatype of the element as an enumerated datatype, and to provide the allowable values.

`Example`	Meaning
`ISO4217:GBP`	Currency, UK Pounds.
`SI:m2`	Square meters
`ISO4217:USD`	Currency, US Dollars.
`US:ft2`	Square feet
`Moodys:rating`	Credit rating

Since unit is defined as a Qualified Name, the prefixes in the above examples must have been previously defined in namespace declarations.

3.7 scaleFactor

An integer power of ten. If a scale value is not 0 (the default), the numeric value of the value attribute must have the proper multiplier applied to arrive at the actual value.

Example	Meaning
3	Thousands
-6	Millionths

[**Scale Factor**] This is actually two words, therefore the attribute should be camelCase.

3.8 precision

Precision is an integer intended to convey the arithmetic precision of a measurement, and therefore, the utility of that measurement to further calculations.

Examples:

Example	Meaning
9	Precision of nine decimal digits.

3.9 decimalPattern

decimalPattern is used to hold locale specific formatting for the value, precision, and scale attributes. It follows the usage of the XSLT Recommendation, Section 12.3 - Number Formatting. It corresponds to the second argument of the number-format function. For more information see the source documents:

- http://java.sun.com/products/jdk/1.1/docs/api/java.text.DecimalFormat.html

- http://java.sun.com/docs/books/tutorial/i18n/format/decimalFormat.html

Example	Meaning
#,###.##	Typical US numbers, default treatment of negative with a leading minus sign.
#,###.##;(#,###.##)	Comma separator every three digits, negative numbers in parentheses and no minus sign.
#.###,##	Comma used as decimal separator.

Inclusion of this and the following attribute in the specification is intended to allow for the use of XBRL in international settings. Although mainly intended for output formatting, they have parsing implications as well. For example, in JDK 1.1.6 in order to parse numbers in a form such as "1,234.56" the decimal format "#,###.##" is needed; the decimal format "#.#" would incorrectly parse "1,234.56" as the integer "1".

The referenced XSLT Recommendation of the W3C itself refers to the JDK 1.1 specification for the details of constructing number formats. This reference to the JDK is not meant as requirement to use the JDK or Java in the implementation of applications that will use XBRL, rather, it merely references a widely available source of information.

3.10 formatName

formatName refers to an element from an XSLT namespace which is used to define a decimal format. It follows the usage of the XSLT Recommendation, Section 12.3 - Number Formatting. It corresponds to the third argument of the number-format function. If present, the document containing the item should also contain an decimal-format element from the XSLT namespace whose name matches the content of this attribute.

```
<xsl:decimal-format
  name = qname
  decimal-separator = char
  grouping-separator = char
  infinity = string
  minus-sign = char
  NaN = string
  percent = char
  per-mille = char
  zero-digit = char
  digit = char
  pattern-separator = char />
```

See http://www.w3.org/TR/xslt#format-number for more information.

schemaLocation

3.11 The item element

As discussed above, an *item* represents a single fact or statement within a report. Although the content model of *item* allows parsed character data, the value is actually further restricted by the datatype given to the item type in the taxonomy. The latter constraint is not readily expressed using an XML DTD.

```
<!ELEMENT item (#PCDATA )>

<!ATTLIST item  %att_AttributeHolder; >
```

`Example (from xbrl-samp-00-04-04.xml):`

```
<item entity="SAMP" period="1998-12-31"
      xmlns="http://www.xbrl.org/core/xbrl-00-04-04"
      xmlns:aicpa="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04"
      schemaLocation="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04.xsd"
      type="aicpa:netCapitalLeasedAssets.grossCapitalLeases"
      unit="ISO4217:USD" scaleFactor="3" precision="3"
      decimalPattern="#,###" formatName="">727</item>
```

Meaning: Gross Capital Leases at calendar year end 1998 for SAMP of $727,000.

`Example (from xbrl-samp-00-04-04.xml):`

```
<item entity="SAMP" period="1998-12-31"
      xmlns="http://www.xbrl.org/core/xbrl-00-04-04"
      xmlns:aicpa="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04"
      schemaLocation="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04.xsd"
      type="notesToFinancialStatements.concentrations">
Concentration of credit risk with regard to short term
investments is not considered to be significant due to
the Company's cash management policies. These policies
restrict investments to low risk, highly liquid securities
(that is, commercial paper, money market instruments, etc.),
outline issuer credit requirements, and limit the amount that
may be invested in any one issuer.
</item>
```

Meaning: The text of the Concentrations note at calendar year end 1998 for SAMP.

The item element should be regarded as a leaf in the tree of XBRL elements within a given instance document. In general, the content of an item cannot contain other markup, and in particular cannot contain other items.

3.12 The label element

All elements within a given taxonomy have a label assigned to them in one or more languages. The label element allows applications to override that label.

```
<!ELEMENT label(#PCDATA)>
<!ATTLIST label
          href       CDATA       #IMPLIED
>
```

Note that the href attribute is CDATA, not IDREF. Producing applications should create documents that use Xpointer instead of IDRDF. If the href content is not `href="xpointer(...)"` then a consuming application can try to interpret it as an IDREF to an item with an id attribute, but the DTD/XSL/DOM hooks will not work correctly, *e.g.*, it will not be possible to use the xsl/Xpath id() function.

Note also that although `<label>` is legal in instance documents, it is really intended for use in taxonomy documents. Occurrence of label elements in the instance document is a last resort. If a company has a particular style of rendering a common accounting concept, that should be held in an extension taxonomy for that company. Labels in instance docs apply to that doc only, which implies a very temporary usage.

`Example (from xbrl-samp-00-04-04.xml):`

```
<group>
  <item type="assets.currentAssets">657</item>
  <label href="xpointer(//item[@type='Assets.currentAssets'])">Working
Capital Assets</label>
</group>
```

Meaning: Every instance of an item with type="assets.currentAssets" gets the new label instead of the one in the taxonomy that would otherwise be providing it.

3.13 The group element

The group element aggregates items by attribute, so that attributes do not have to be given in full on each item. Items inherit the value of attributes from the closest parent with an explicit reference to the attribute value. The group element provides a convenient way to group similar items together, without forcing a particular hierarchy. Entity, period and type are all useful grouping attributes, and the specification allows each document to use them in whatever order is desired.

```
<!ELEMENT group ANY>
<!ATTLIST group
          %att_AttributeHolder;
>
```

Example (from xbrl-samp-00-04-04.xml):

```
<group entity="SAMP"
       type="netPropertyPlantandEquipment.netCapitalLeasedAssets"
       xmlns="http://www.xbrl.org/core/xbrl-00-04-04"
       xmlns:aicpa="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04"
       schemaLocation="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04.xsd"
       unit="ISO4217:USD" scaleFactor="3" precision="3"
       decimalPattern="#,###" formatName="">
 <item period="1998-12-31">727</item>
 <item period="1997-12-31">635</item>
 <item type="notesToFinancialStatements.concentrations" period="1998">
Concentration  of  credit  risk  with  regard  to  short  term  investments  is  not
considered to be significant due to the Company's cash management policies.
 </item>
</group>
```

Meaning: Use of *group* with items varying by period and type.

Example (from xbrl-samp-00-04-04.xml):

```
<group entity="SAMP"
       xmlns="http://www.xbrl.org/core/xbrl-00-04-04"
       xmlns:aicpa="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04"
       schemaLocation="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04.xsd"
       unit="ISO4217:USD" scaleFactor="3" precision="3"
       decimalPattern="#,###" formatName="">
<group period="1998-12-31">
  <item type="longTermAssets.netCapitalLeases">727</item>
  <item type="notesToFinancialStatements.concentrations">
Concentration  of  credit  risk  with  regard  to  short  term  investments  is  not
considered to be significant due to the company's cash management policies.
  </item>
 </group>
 <group period="1997-12-31">
  <item type="longTermAssets.netCapitalLeases">635</item>
 </group>
</group>
```

Meaning: Use of *group* with nesting by period, and items within the groups varying by type.

The significance of these two examples is that the "group" container element is designed to allow for more compact instance documents. It is *not* intended to be a general structuring mechanism intended to convey presentation related information. Applications should not produce instance documents with *group* elements expecting all consuming applications to respect those groupings, their type, label or other attributes for any other usage than to assign attribute values to all their contained elements. The sole exception to this general principle concerns footnotes; use of the *group* element as the location of a footnote reference is allowed even though this may appear to be primarily a presentational function.

3.14 Document Types

As a practical matter, many XBRL document instances will be structured in such a way that a group element is th outermost container. Consequently, the "document type" of an XBRL instance document will usually be "group". Non-namespace aware parsers will not validate properly given the use of namespace qualifiers, so generally a declaration such as the example below will be needed.

Example:

```
<!DOCTYPE group SYSTEM " http://www.xbrl.org/core/xbrl-core-00-04-04.dtd"
[<!ATTLIST group
  xmlns:USGAAP  CDATA #FIXED "http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04"
  xmlns:ISO4217 CDATA #FIXED "http://www.iso.org/4217"
  xmlns:DUNS    CDATA #FIXED "http://www.Dun-and-Bradstreet.com/"
>]>
```

Meaning: these are three namespaces that are used in other examples in this specification, with hypothetical bindings in the case of the ISO4217 and DUNS name spaces.

3.15 Additional attributes (Non-Normative)

The XBRL Core DTD allows the use of Dublin Core metadata attributes on all of the previous elements. Of particular interest is the keyword attribute. In principle, the keyword attribute might be used to include meta-information about items, such as "restated", "pro-forma", "budget", "actual", and "projected", as in

Example	Meaning
`<group dc:keyword="actual"/>`	Deprecated usage.

However, this usage is deprecated on the grounds that some distinctions, such as budget *vs.* actual, are common to many financial reporting situations and cross application interoperability requires a common treatment while still satisfying internationalization requirements. Only lack of time has prevented this issue from being addressed in the current specification. It is not clear, for example, whether the introduction of contexts such as "budgeted" simultaneously introduces the need for a built in versioning mechanism (*e.g.*, the budget as of 2000-02-01, the budget as of 2000-02-05) or whether versioning can be adequately dealt with by standards which would work generically for any XML document.

Hence, the working group expects a future revision of this specification to define one or more such attributes in the entity `%att_AttributeHolder` along with enumerated data types for their content. Implementers should suggest, and anticipate the inclusion of, additional context attributes. In the meantime, applications should use a form such as the following to extend the XBRL core definitions if necessary:

Example

Meaning

```
<!ATTLIST item context
          (actual | budget | projected | pro-forma | restated)
          #IMPLIED>
```

Preferred usage.

with the understanding that the final specification will be informed but not constrained by early implementation.

3.16 #IMPLIED resolution

If an attribute that is specified as #IMPLIED is not present in an instance of an item, it must be available attached to a container element that is an ancestor (in the XPath sense) of the item element.

The XPath expression "`ancestor-or-self::*[@implied-attribute][1]/@implied-attribute`" finds the nearest value of an attribute, which may appear either attached to an element of higher up in the document tree.

The implication of this is that an XBRL item element is always fully specified in terms of all of its attributes, even if some of those attributes are not directly attached to the item element itself. The *id* attribute is the obvious exception; an item without an id does not inherit one from its containing parent.

There are no default values for any of the attributes that can appear on an item or group. This is a key requirement for full internationalization. Every XBRL instance document must specify, for all items, all relevant attributes—in particular, item types do *not* default to US dollars, US measurements, US number formatting conventions, US accounting principles, or US English for anything other than the names of elements and attributes.

3.17 Design Rationale (Non-normative)

Some of the features used in XBRL instance documents appear to be at odds with conventional definitions of XML document types. Order independence and the heavy use of attributes are relatively novel but offer crucial advantages to meet both defined current requirements and known future requirements.

3.17.1 Order independence

Although the ordering of financial information in presentation to a human is important, ordering is irrelevant insofar as the exchange of data between software applications is concerned. Therefore, in XBRL the ordering of *item* elements is unimportant and there is no document structure defined within the core specification. The main reason for this is that it greatly increases modularity. For one thing, it allows any XML document in the world that happens to describe financial information to include an XBRL item to describe it. Consider an HTML press release with an embedded XBRL item:

`Example (from xbrl-samp-00-04-04.html):`

```
<P>San Sushi, CA, 7 December 2002:  Sample Data Inc. today
announced net revenues of $<item
xmlns="http://www.xbrl.org/core/xbrl-core-00-04-04"
xmlns:aicpa="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04"
schemaLocation="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04.xsd"
type="aicpa:netSalesRevenue.grossSalesRevenue"
period="2002-07-01/2002-10-01" entity="NASDAQ:SAMP"
precision="2" scaleFactor="6" decimalFormat="#.#"
units="USD">7.2</item>m for the third quarter of 2002.</P>
```

XBRL embedded in HTML.

There are many reasons for wanting this embeddability feature; support for XML-aware search engines, and embedding XBRL items within the documents of other electronic commerce protocols are only two. As a historical note, this was one of the earliest decisions of the XBRL specification working group (Chicago, October 1999), to factor out the structure into a separate document, instead of burdening every instance document with its repetition.

Order independence also simplifies the combination of financial information from different periods or entities, or even for the same entity under different reporting regimes, since in most cases an XBRL instance document can be created by concatenating other XBRL instance documents.

Finally, order independence makes it easier for individual reporting entities to define incremental new types, labels in different languages, *etc.* This level of extensibility is known to be a key requirement for meeting the reporting needs of a substantial number of business entities and has been an overriding consideration in the design of the language.

> [**Instance Includes**] Since modularity is good for schemas, does the same apply to instance documents? If instance documents are long lived rather than merely a transfer syntax, it suggests that functionality such as `<include href=""/>` might be useful for instance documents.

3.17.2 Use of Attributes

The interpretation of most items is embedded within attributes. Applications that process the information in an XBRL document, such as (say) rendering as HTML will process each item and perform a "lookup" into the appropriate taxonomy in order to extract properties such as its appropriate text label in a given human language, to determine the order in which it should be presented in a table relative to other items, *etc.* In principle, all of the attributes of items (except *id*) could have been done as an optional sub element. However, this would have sacrificed the ability to rely on the semantics of #IMPLIED attributes, as well as making the document instance unnecessarily verbose.

The use of a general *group* construct to assign content to those attributes, mean that pure XBRL documents resemble a database more than they resemble a presentation-oriented document. This is intentional, since future extensions of XBRL into the arena of internal reporting will in fact require XBRL to serve, in effect, as a neutral format for passing multidimensional data from one application to another. Again, this was one of the earliest decisions of the XBRL specification working group (Chicago, October 1999), to use a minimal set of elements and to exploit attributes as the preferred way of establishing a rich semantic context for any given financial number.

> **[Footnote References]** Although it is clear how Footnotes will be captured through the use of items and rollups, it is still necessary to show how references to them will work in general. Similarly for Notes.

4 Syntax of Taxonomies

Although only one XBRL taxonomy exists as of the release of this specification, there will be many. Each taxonomy consists of a list of new element definitions along with relations between these elements. The definition of a taxonomy is done by extending the XML schema using several linking structures that will be described here. The syntax of a taxonomy is defined in a metamodel that contains definitions for certain simple and complex data types, each of which will be described here.

4.1 The monetary and pure datatypes

The XBRL metamodel defines a datatype "monetary" that specializes the "decimal" type. Monetary strings are interpreted with respect the enclosing decimalFormat for any item where they appear. A taxonomy which includes numeric elements that are meant to be interpreted as monetary values should use this datatype rather than "string", which is the default.

The empty string "" is not a valid instance of the monetary datatype.

A negative number is a valid instance of the monetary datatype. Any item with datatype "monetary" can have a negative number as its value. The presentation of negative numbers (often in parentheses) is relatively rare in financial reporting; it is the responsibility of the producing application to ensure that the sign of the number indeed indicates a negative balance, *i.e.,* negative with respect to the normal balance for a given type.

The datatype "pure" also specializes the "decimal" type. While items with "monetary" datatype should have a measure from the ISO 4217 namespace of currencies, items with "pure" datatype are to be used for pure numbers such as ratios, percentages, and number of shares, that is, financial numbers other than amounts of money.

4.2 element

An element has a name and data type. Because the US GAAP C&I Taxonomy contains several hundred items, a method was needed to prevent name clashes and this led to the convention of using names such as "MarketableSecurities.availableForSale" containing both the colloquial name of the item as well as its immediate parent in the taxonomy. This convention is not a requirement of any taxonomy, although it is the case that all element names must be unique within a given taxonomy.

`Examples`

```
<element name="dividends.preferred"
         type="monetary"/>
<element name="summaryofSignificantAccountingPolicies.stockBasedCompensation"
         type="string"/>
```

Meaning: From US GAAP C&I Taxonomy.

The usage of the "element" element in an XBRL taxonomy is syntactically no different that its usage in XML Schema. For those familiar only with DTDs, these definitions are equivalent to use of a definition such as `<!ELEMENT dividends.preferred>`, with the additional power of Schema Constraints, in this case, data type constraints. This does *not* mean, however, that XBRL instance documents should be construed as containing forms such as `<dividends.preferred>` as structural elements; they do not.

The US GAAP C&I Taxonomy included 1124 element definitions as of 31 March 2000.

> [**Unique Elements**] XSchema gives you the ability to specify uniqueness, so that one can specify that each 'foo' element has to be unique, or some such. Taxonomy documents validate against the XBRL Metamodel Schema, so that is where the uniqueness constraint on element names would be enforced. In practice, tools do not support validation against this file yet, but at some point we may want this constraint to go in, and take advantage of it to ensure that element names are unique. This approach still needs to be complemented with a naming convention that reduces the likelihood that a taxonomy builder will create non-unique names in the first place. The underlying issue is that taxonomies are easier to understand when they are strictly hierarchic, but it creates problems of its own: (1) it forces the taxonomy and any custom taxonomies to capriciously choose to retitle entries that may appear in more than one place; (2) if the accepted verbiage for an entry changes, one cannot easily change the historical links for numbers appearing against the old verbiage; (3) it also opens the door for duplicate names with no obvious verification. There is, in fact, no syntactic nor semantic requirement that a given element appear as the "from" attribute of only one rollup; that is, any element can have multiple parents. Hence, it is possible to rearrange any taxonomy into a tangled hierarchy that would take advantage of this and thereby reduce or eliminate name clashes, multiple names for identical concepts, *etc.* A future release of the US GAAP C&I Taxonomy may do this, so implementers are strongly cautioned not to write consuming applications that assume a given element has only one parent.
>
> [**Long Names**] Partly as a consequence of the parent.child naming convention, partly because of the way the taxonomy was constructed, the AICPA US GAAP C&I Taxonomy element names range from 10 characters to 199, and average 52 characters per name. Implementation experience shows that this makes application programming inconvenient, alternatives are being considered that would also address the unique naming problem.

4.3 rollup

The rollup element defines how elements are related to one another in a parent-child relationship. The actual declaration within the XBRL metamodel defines a RollupType, with the rollup element being an element of that type.

```
<complexType name="RollupType">
  <attribute name="from" type="QName"/>
  <attribute name="to" type="QName"/>
  <attribute name="sense" type="string" default="none">
    <enumeration value="add"/>
    <enumeration value="subtract"/>
    <enumeration value="none"/>
  </attribute>
  <attribute name="order" type="decimal" default="1">
  <attribute name="reference" type="string"/>
</complexType>
```

There are two required attributes, *from* and *to* and three optional attributes, *sense, order,* and *reference.*

4.3.1 from

A qualified name that indicates the child element in the relation.

4.3.2 to

A qualified name that indicates the parent element in the relation.

4.3.3 sense

Indicates whether the relationship is simply one of inclusion (*e.g.*, discussion of interim periods is part of the management discussion and analysis, and both are text), or is an arithmetic relationship and whether its mathematical sense is additive or subtractive. Enumerated values are none, add, and subtract; it defaults to "none".

Appearance of sense="subtract" is expected to be quite rare. For example, "depreciation and amortization" has the sense="subtract" in relation to its parent, "fixed assets". However, its components, depreciation and amortization, when reported separately, will each have sense="add" in relation to their parent, "depreciation and amortization".

4.3.4 reference

An optional text string that provides a human readable reference to supporting accounting or other literature.

4.3.5 order

A nonnegative decimal number indicating how sibling elements are normally ordered for presentation within their parent element. It defaults to "1". A consuming application is in principle free to ignore this order parameter.

`Examples` (From US GAAP C&I Taxonomy)

```
<rollup from="mandatorilyRedeemableSecurities.votingCharacteristics"
        to="liabilitiesandStockholdersEquity.mandatorilyRedeemableSecurities"
        sense="none" order="3"/>
<rollup from="assets.currentAssets"
        to="balanceSheet.assets" sense="add" order="1"/>
<rollup from="accountsReceivable-tradeNet.accountsReceivable-trade"
        to="receivables.accountsReceivable-tradeNet"
        sense="add" order="1"/>
```

Meaning: Voting Characteristincs are just a text string, and so the sense of any rollup with that as its "from" or "to" attribute is "none". Assets and Accounts receivable, on the other hand, are monetary and therefore can be rolled up to (and from) with sense "add" or "subtract" (or "none").

The US GAAP C&I Taxonomy includes several hundred rollup elements.

The syntax of the "from" and "to" elements is essentially shorthand for XPointer syntax, for example:

Example

```
<rollup from=xpointer(element[@name="assets.currentAssets "])
        to=xpointer(element[@name="balanceSheet.assets"])
        sense="add" order="3"/>
```

Meaning: Full XPointer syntax

4.4 label

One of the key internationalization features of XBRL is that although each taxonomy defines a single set of elements representing a coherent set of accounting concepts, the label—a string used to present the name of that concept—is declared separately with an indication of the language using the XML standard `lang` attribute. Thus, a given set of financials could be presented by a single application in a language selected by the user (although recasting the underlying financials under a different set of national accounting principles is a far more complex matter).

```
<complexType name="LabelType" content="textOnly">
  <attribute name="href" type="QName"/>
  <attribute name="xml:lang" type="language"/>
</complexType>
```

Examples:

Example (From US GAAP C&I Taxonomy)

```
<label href="balanceSheet.cashandCashEquivalents"
       xml:lang="fr">Argent Comptant</label>

<label href="balanceSheet.cashandCashEquivalents"
       xml:lang="en">Cash and Cash Equivalents</label>
```

Meaning : Consuming applications may display the item to speakers of English and French in the appropriate language.

These definitions can be in separate files; ordering is unimportant. Labels can be overriden in an instance document. The latter feature is important because individual companies routinely adjust the wording of an otherwise standard category, to reflect their particular circumstances.

> [**Label Processing**] Assembling elements with their labels is one of the most basic operations that a program using the taxonomy has to do. As the specification stands to do so requires one to go through a O($n*m$) operation (n - number of elements, m - number of labels). At a minimum there will be n labels (means O(n^2)). In the US GAAP C&I taxonomy n is already > 1100. If the `label` element had to appear inside the `annotation` of an element, this would allow faster processing; a different element (e.g., `relabel`) could do overriding.

4.5 links

The links element is a container for rollup and label elements in a Taxonomy. Elements do not need to be within the scope of the links element.

```
<complexType name="LinksType">
       <element name="rollup" type="RollupType"/>
       <element name="label" type="LabelType"/>
</complexType>
<element name="links" type="LinksType"/>
```

`Example` (this is the structure of the US GAAP C&I Taxonomy schema file):

```
<schema targetNamespace="http://www.xbrl.org/core/xbrl-00-04-04"
        xmlns="http://www.xbrl.org/core/xbrl-00-04-04.xsd">
  <element name="statements" type="string"/>
  <element name="statements.balanceSheet" type="monetary"/>
  *******
  <element name="notestoFinancialStatements.noncurrentLiabilities"
           type="string">
  <links>
  <rollup from="statements.balanceSheet" to="statements"
          sense="none" order="3"/>
       *******
  <label href="notestoFinancialStatements.noncurrentLiabilities"
         xml:lang="en">Noncurrent liabilities</label>
  </links>
</schema>
```

Meaning: rollup and label elements should be contained within the <links> element.

> [**Flush Links**] The purpose of the `links` is unclear. If we can flush them we should.

4.6 Additional attributes (Non-normative)

As noted earlier, there are additional attributes of instance document items that are anticipated for future specification releases. A similar cautionary note applies also to the attributes of both elements and rollups. Implementers wishing to extend the attribute set should do so in a modular fashion by defining an implementation specific Metamodel Schema with an appropriate definition of the additional attributes.

> **[Taxonomy Attributes]** Actually, this seems drastic. Surely there is some way to declare attributes without having to trash Metamodel.xsd.

4.7 Design Rationale (Non-normative)

There are several strengths displayed by the current design of XBRL taxonomies. First, it leverages existing standards, specifically, XML and those portions of XML Schema which are very unlikely to change.

Second, it supplies a commonly used set of elements (in this case, US GAAP for Commercial and Industrial companies) and therefore does not require every document instance to be designed "from scratch".

Third, it is fully extensible. Alternative approaches to extensibility were considered and failed to meet the requirements for complete independence from language, set of accounting principles, and document types. This required a novel use of XML schema not merely as a replacement for DTDs, but as a kind of concept definition language.

5 Semantics of Instance Documents

The semantics of instance documents and their contents are specified here only insofar as they impact the operation of software applications that use this specification. The primary topics in this regard are:

- Protocol independence
- Processing by consuming applications
- Validation
- The parent-child relationship

5.1 Protocol independence

XBRL, unlike many other XML based specifications, is designed to work equivalently under many different transport and transaction protocols. It is designed solely as a format for exchanging data between an arbitrary producing application and an arbitrary consuming application. Details of how the XBRL data (the actual text strings containing the elements and their content) is written, read, and parsed is outside the scope of the specification. The simplest possible scenario, one normally used for exposition, is that a producing application writes a well formed file (an instance document) to a file systems and another application reads, parses and manipulates that file content using the DOM. XSL processing of an XBRL data file is an instance of this general scenario. Applications designed to edit XBRL data are regarded as primarily consuming applications that must incidentally produce XBRL data as well.

5.2 Processing by consuming applications

While some consuming applications may be able to perform processing on an XBRL data file without referring to any of the taxonomies that it references, normally, the interpretation and processing of any given XBRL item is relative to the contents of a named taxonomy.

For example, to correctly produce a table of values with rows corresponding to an ordered set of types and columns representing different periods, at a minimum it is necessary to dereference the appropriate shemaLocation attribute in order to find the `label` elements and the `order` attributes corresponding to each item type. This is similar to a relational database join, where the document instance contains an "item" table, some of whose columns (e.g. type) are used as foreign keys into tables in the taxonomy.

Treatment of relative pathnames and cacheing of the taxonomy file is implementation dependent. For example, if a document instance contains a relative URL as the location of a schemaLocation attribute, it is up to the consuming application to dereference it; it is an error if the underlying taxonomy cannot be found.

5.3 Validation

Validation of an instance document against the XBRL core DTD is expected but not required of any consuming application. Validation of an instance document against all of the taxonomies to which it refers by using a generalized validation is also possible, although it is not expected. Thorough validation of an instance document would, for example, require at least the following processing for each item:

- Retrieve the content of the item. Call this *itemContent*.
- Retrieve the value of the type attribute of the item, computing the value via inheritance from ancestors if necessary. Call this *itemType*.
- Retrieve the value of the schemaLocation attribute of the item, computing the value via inheritance from ancestors if necessary. Call this *itemTaxonomy*.
- Retrieve the contents of the elment in *itemTaxonomy* whose name attribute equals *itemType*. Call this *itemTypeElement*.
- Retrieve the content of the type attribute of *itemTypeElement*. Call this *itemDatatype*.
- Test that *itemContent* satisfies the *itemDatatype* schema constraint.

One of the unique properties of the XBRL language is that because it specifies no document root, for any validation or processing purposes, anything appearing outside the boundaries of an item element can be ignored. The following should pass validation as an XBRL document instance, and should be processable by any XBRL compliant application:

```
Example (from xbrl-samp-00-04-04.html):
<xbrl:group entity="SAMP"
       type="netProperty_PlantandEquipment.netCapitalLeasedAssets"
       xmlns:xbrl="http://www.xbrl.org/core/xbrl-core-00-04-04"
       xmlns:aicpa="http://www.xbrl.org/us/aicpa-gaap-us-ci-00-04-04"
       schemaLocation="http://www.xbrl.org/us/aicpa-gaap-us-ci-00-04-04.xsd"
       unit="ISO4217:USD" scaleFactor="3" precision="3"
       decimalPattern="#,###" formatName="">
<h2>Capital Leases</h2>
<table><thead><tr>
<td width="30%"></td>
<td width="10%">1998</td>
<td width="10%">1997</td>
</tr></thead>
<tbody>
<tr><td>Net Capital Leases (in 000s)</td>
<td>$<xbrl:item period="1998-12-31">727</xbrl:item></td>
<td>$<xbrl:item period="1997-12-31">635</xbrl:item></td>
</tr>
</tbody>
</table>
<h2>Concentrations</h2>
<p>
<xbrl:item type="notesToFinancialStatements.concentrations"
           period="P1Y/1998-12-31">
Concentration of credit risk with regard to short term investments is
not considered to be significant due to the Company's cash management
policies.
</xbrl:item>
<p>
</group>
</body></html>
```

which displays in a browser approximately as follows:

Capital Leases

	1998	1997
Net Capital Leases (in 000s)	$727	$635

Concentrations

Concentration of credit risk with regard to short term investments is not considered to be significant due to the Company's cash management policies.

The content of text items is highly constrained. No other markup (*e.g.*, presentation related HTML tags for bold, italics, images) may occur inside of text items, and in particular, occurrences of other XBRL elements within the content of an item element is an error. Any application claiming to be an XBRL validator should, at a minimum, detect the occurrence of XBRL elements inside the content of an XBRL item and signal an error.

This is not true of group elements; the content model of the group element allows for any embedded markup, including style information. Some consuming applications need styling information on their data. The problem is that not all producing applications can be relied upon to put it there. In reality, any particular consuming application could in fact specify any XML attributes it wants anyway, expect them to be "optionally present", and just bomb out or default them when they aren't there. The fact that the application has to store its own state in a transfer file suggests that it has poor modularity is poor, but so long as nobody is under the illusion that optional means anything other than optional, the compromise position that XBRL takes is that although items can't contain markup, the group element can. Therefore, the group element is expected to be used to carry various kinds of presentation related information, leaving the item elements inside to remain easily accessible and parseable by software applications that only need access to the data, not the styling information.

5.4 The Parent-Child relationship

There is no nesting of XBRL items. Whatever structural relationships as might be desirable in an XBRL document instance are captured in rollups. Suppose a taxonomy contains the following rollups:

Example

```
<rollup from="notestoFinancialStatements.businessCombinations"
        to="statements.notestoFinancialStatements" sense="none"
        order="5"/>
<rollup from="businessCombinations.poolingofInterestsMethod"
        to="notestoFinancialStatements.businessCombinations"
        sense="none" order="1"/>
<rollup from="businessCombinations.purchaseAccounting"
        to="notestoFinancialStatements.businessCombinations"
        sense="none" order="2"/>
```

Meaning: From US GAAP C&I Taxonomy

What this says is that a financial statement note concerning Business Combinations may include not only its own text, but may also include two different types of sections, one of which concerns pooling of interests and the other concerns purchase accounting. It may include any number of either type of subsection.

> **[Everything Repeatable]** The simplest approach to dealing with repeatability is to let everything be repeatable. It is up to the consuming application (which might be a validator) to worry if the repeated elements are inconsistent. If they aren't inconsistent, it didn't matter anyway. When creating the US GAAP C&I Taxonomy, information about which elements could normally be repeated was captured and has been preserved for future use, however, this information is not reflected in the resulting taxonomy definition file.

The XBRL instance document might contain the following, which shows a parent with two children of the same type:

Example (from xbrl-samp-00-04-04.xml):

```
<item type="notestoFinancialStatements.businessCombinations">
During 1998, Sample Data Incorporated completed the acquisition of Small Fry
Systems, a Delaware corporation.  Sample Data Incorporated also acquired
Exemplar Software Inc., also a Delaware corporation.
</item>
<item type="businessCombinations.purchaseAccounting">
The acquisition of Small Fry Systems for $8.7m, an excess of $3.7m over its
book value of $5m, was accounted for with a charge to expenses for in-process
R&D of $2.5m and the remaining excess of purchase price over book value being
assigned to Goodwill.
</item>
<item type="businessCombinations.purchaseAccounting">
The acquisition of Exemplar Software Inc. for $2.2m, an excess of $1.2m over
its book value of $1m, was accounted for with a charge to expenses for in-
process R&D of $1.1m and the remaining excess of purchase price over book value
being assigned to Goodwill.
</item>
```

(This example is only meant to illustrate a legal arrangement of items in a document instance, is not normative with respect to US GAAP or any other taxonomy, and is certainly not a realistic sample of the actual language of a financial statement note). Taxonomies are also free to include numeric items that roll up into non-numeric items, such as in the case of a tax reconciliation Note which has some numeric information that must then roll up into a note which is otherwise largely text.

The following example should further clarify the role of parent items. In general, producing applications are encouraged to include all the parent items of non-empty items, since an important class of consuming applications would benefit from their inclusion. The following is a fragment of a taxonomy file:

Examples

```
<rollup from="stockholdersEquity.additionalPaidInCapital"
        to="liabilitiesandStockholdersEquity.stockholdersEquity"
        sense="add" order="3"/>
<rollup from="stockholdersEquity.retainedEarnings_-deficit-"
        to="liabilitiesandStockholdersEquity.stockholdersEquity"
        sense="add" order="4"/>
<rollup from="stockholdersEquity.treasuryStock"
        to="liabilitiesandStockholdersEquity.stockholdersEquity"
        sense="subtract" order="5"/>
```

The following is a legal fragment of XBRL (assuming that other item attributes are available from an enclosing group container):

```
<item type="liabilitiesandStockholdersEquity.stockholdersEquity">201</item>
<item type="stockholdersEquity.additionalPaidInCapital">301</item>
<item type="stockholdersEquity.retainedEarnings_-deficit-">100</item>
<item type="stockholdersEquity.treasuryStock">200</item>
```

In this example, Stockholders Equity = Additional Paid-in Capital + Retained Earnings - Treasury Stock, that is, 201 = 301 + 100 - 200. Three child items and the parent are all present (there are other children as well, but their values are zero in this example). From the consuming application's point of view, the meaning of several variations on this scenario are worth considering as well.

1. The Stockholder's Equity item is missing entirely. In this case, the input is valid, but an application might be unable to process the input unless it makes the assumption that the given list of children is complete (*i.e.*, a "closed world assumption", the assumption that there is no other relevant information aside from what was provided).
2. The Stockholder's Equity item is present, but its content is empty. The consuming application may treat this as a syntax error, since the data type specified was "monetary" which disallows the empty string.
3. The Stockholder's Equity item is present, but its value is 200, which suggests that the producing application performed rounding. The consuming application may choose to use either the reported value (200), use the precision attributes to see whether the value is within an acceptable range, raise an exception, or perform any other processing appropriate to its purpose. Obviously, an audit application would treat this situation differently than a rendering program.
4. The Stockholder's Equity item is present, but its value is 2000. The consuming application's options are the same as under case 3.
5. There are two instances of the Stockholder's Equity item, both with the same value, 201. In general, arithmetic rollups are meant to allow only one child of a given type under a given parent item. In this case, since the items are identical, either can be ignored.
6. There are two instances of the Stockholder's Equity item, one with the value 71 and the other with the value 130. The meaning is undefined, however, a validating application should flag this as an exception.
7. The Stockholder's Equity item is present, its value is 201, but the Treasury Stock item is absent. Mathematically, this is really no different in principle from case 4 under a closed world assumption. To see why this is so, consider the following four mathematically equivalent statements.

   1. A = B+C-D
   2. B = A-C+D
   3. C = A-B+D
   4. D = C-A+B

   (Note: this is only four of at least 36 reorderings all equivalent to the first). Clearly, each of these might legitimately be the way that one presents the relationships of these four quantities. Each one leads to a different set of roll ups:

   1. B roll up (+) to A and C roll up (+) to A and D roll up (-) to A
   2. A roll up (+) to B and C roll up (-) to B and D roll up (+) to B
   3. A roll up (+) to C and B roll up (-) to C and D roll up (+) to C
   4. C roll up (+) to D and A roll up (-) to D and B roll up (+) to D

> There are no conflicts between any of these rollups. A taxonomy could in principle just include all 12 of them. It is only because a taxonomy implicitly expresses normal, conventional ways of presenting certain categories of accounting data that one would select one or the other. Returning to the missing Treasury Stock example, it is only because rollup 1 instead of rollup 4 was used that it appears to be a different case.
>
> The consuming application's options are the same as under case 4, and it should behave differently depending on whether it is only meant to render the reported information in a browser (in which case it need not flag an error) or an audit program meant to detect discrepancies (in which case it should flag an error).

Consideration of these and similar cases indicates why producing applications are encouraged to provide all parent item contents for any child that has content. The argument for doing so is that (a) it is easier for any producing application to comply with this than for a consumer to reconstruct the information (b) it removes ambiguity about intent, (c) it is definitely more convenient for any consuming application, which retains the option of either using the value directly or doing more work and checking it against its children (d) the space penalty is relatively small.

Note that XBRL does not force a consuming application to assume completeness nor prevent it from doing so. XBRL allows for producing applications to create documents which that are either complete or incomplete, to serve to consuming applications that do or do not assume completeness. Naturally, a document created given one set of assumptions to an application making a different set of assumptions and problems will arise.

5.5 Data Integrity and Confidentiality

There are many applications which require financial information to be transmitted securely, with a particular emphasis on data integrity (leading to the use of hash totals, etc., in financial data) and confidentiality (leading to the use of cryptographic means of protection). XBRL deliberately provides neither of these mechanisms, since its focus is on transmission of actual financial content in an agreed-upon format; it is assumed that like any other block of data, data integrity can be enhanced by adding redundant error correction bytes, by cryptographic hashing and signing with a private key, etc. These mechanisms are all outside the scope of XBRL.

6 Semantics of Taxonomies

Extensibility of taxonomies is a critical feature of XBRL. Taxonomies must be extended to accommodate virtually any business entity's unique reporting requirements while maintaining some comparability across entities, or else XBRL will fail. These reporting requirements may be either external (the primary focus of this release) or internal (a goal for the near future). XBRL taxonomies may be constructed in such a way as to refer to other taxonomies.

6.1 IMA Example

The following extended example, taken from the annual report of the IMA (Institute of Management Accountants), a non-profit organization, illustrates how this should be done.

The source of the example can be found in ima-00-04-04.xml (the document instance) and ima-00-04-04.xsd (the schema) (in a realistic example, both the name of the document containing the IMA statements as well as the name of the IMA taxonomy would be more likely to be dated as of the actual financial report itself, such as June 30, 1999, rather than the date of the XBRL version, April 4, 2000).

LIABILITIES, DEFERRED REVENUES and NET ASSETS	**1999**	**1998**
Accounts payable and accrued expenses	$3,356	$2,686
Bonds payable, net of discount	3,382	3,575

LIABILITIES, DEFERRED REVENUES and NET ASSETS	**1999**	**1998**
Total liabilities	6,738	6,261
Deferred revenues		
Membership dues	4,083	3,530
Other	472	510
Total deferred revenues	4,555	4,040
Net assets		
Unrestricted		
IMA		
Current Operating Fund	4,020	5,986
Reserve Fund	8,708	8,658
ICMA	(35)	201
IMAMEF		
Board Designated	100	100
Undesignated	3,455	3,030
IMAFAR	70	—
Total net assets	16,318	17,975
Total liabilities, deferred revenues and net assets	$27,611	$28,276

There are three features that distinguish this portion of the balance sheet from that of the prototypical commercial and industrial company. Keep in mind that the purpose is not to recreate the presentation in every detail, but rather to capture and correctly relate to one another all of the numbers that have been reported in such a way that they can be more easily compared with comparable organizations.

6.2 Overriding the label of an existing element

First, there is no equity section; instead there is a Net Assets section. In XBRL, functionally equivalent terms can be handled through relabeling. In this case we can refer to an existing concept and give it a new label in our IMA taxonomy.

```
<label href="aicpa:balanceSheet.liabilitiesAndStockholdersEquity"
       xml:lang=en>Liabilities, Deferred Revenues and Net Assets</label>
```

This element, and all the others in this section, could be included in a single taxonomy file which is included with this specification.

6.3 Creating a new rollup structure

Second, the rollup structure differs in many details from that of the basic US GAAP C&I liabilities section, which, for example, has no deferred revenues section. Here, we define four new elements and assign rollups and labels. Note that the existing US GAAP concept of "Accounts Payable and Accrued Expenses" is being given a new rollup target, a newly defined "Liabilities" parent. A child may have multiple parents but only one parent should be used in a given document instance.

```
Example (from ima-00-04-04.xsd):
<element name="liabilitiesAndStockholdersEquity.liabilities"
```

```
         type="monetary">
<annotation><documentation>
Total Liabilities, an alternative US GAAP rollup node
</documentation></annotation></element>

<element name="liabilities.deferredRevenues" type="monetary"/>
<element name="deferredRevenues.membershipDues" type="monetary"/>
<element name="deferredRevenues.other" type="monetary"/>

<links>
<rollup from="liabilitiesAndStockholdersEquity.liabilities"
        to="aicpa:BalanceSheet.liabilitiesAndStockholdersEquity"
        sense="add" order="1"/>

<rollup from="liabilities.deferredRevenues"
        to="liabilitiesAndStockholdersEquity.liabilities"
        sense="add" order="1"/>
<rollup from="aicpa:currentLiabilities.accountsPayableAndAccruedExpenses"
        to="liabilitiesAndStockholdersEquity.liabilities"
        sense="add" order="2"/>

<rollup from="deferredRevenues.membershipDues"
        to="liabilities.deferredRevenues"
        sense="add" order="1"/>
<rollup from="deferredRevenues.other"
        to="liabilities.deferredRevenues" sense="add" order="1"/>

<label href="liabilitiesAndStockholdersEquity.liabilities"
       xml:lang="en">Liabilities</label>
<label href="liabilities.deferredRevenues"
       xml:lang="en">Deferred Revenues</label>
<label href="deferredRevenues.membershipDues"
       xml:lang="en">Membership Dues</label>
<label href="deferredRevenues.other"
       xml:lang="en">Other</label>
<label href="liabilitiesAndStockholdersEquity.netAssets"
       xml:lang="en">Net Assets</label>
<links>
```

[**Taxonomy Extensions**] The impact of individuals and organizations developing their own taxonomies in isolation is not a desirable outcome. How should the sharing and reuse of these be encouraged? Creation of a repository for custom taxonomies is, in some sense, presumed, but at the level of national reporting jurisdictions and industry specific taxonomies, not at the level of companies.

6.4 Periods and Item types (Non-normative)

As noted in a previous section with the discussion of the rollup relation, as a general guideline it is important when designing a taxonomy not to confuse the mathematical and logical containment relationships (which is what the taxonomy represents) with presentational elements (which is the job of consuming applications). It is also important to avoid redundancy in both the taxonomy and in instance documents referring to it. These guidelines, along with the interpretation of all items of "as-of" measures as being at the end of the named period, mean that sometimes it is useful to have a "priorPeriodBalance" child.

```
Example (instance):
<item type="statement.stockholders_Equity"          period="1999">100</item>
<item type="stockholders_Equity.priorPeriodBalance" period="2000">100</item>
<item type="stockholders_Equity.netIncome"          period="2000">200</item>
<item type="statement.stockholders_Equity"          period="2000">300</item>
Example (taxonomy):
<rollup from="stockholders_Equity.priorPeriodBalance"
        to="statement.stockholders_Equity" sense="add" order="1"/>
<rollup from="stockholders_Equity.netIncome"
        to="statement.stockholders_Equity" sense="add" order="2"/>
```

Meaning: This is the recommended way of structuring financial information that combines point-in-time balances with measures of activity during a period.

In this example, note that the prior period balance for the year 2000 is identical to the parent value for the prior year. Although it is redundant information, it nevertheless results in a very simple rollup structure in which the parent (the end of period balance) is equal to the sum of its children (a prior period balance plus the net income).

[Alternate Breakdowns] On a related note, it is often the case that a financial number can be reported by using one of several methods of computing it. The general case is that when there are multiple alternative and common ways of breaking down a parent item, then so long as the parent item's value would be the same when derived using any of the breakdowns, then a taxonomy *may* create multiple children of that parent, one to represent each of the methods. The taxonomy *should* include those children which represent the most common alternatives.

6.5 Reporting on Subsidiaries (Draft)

Third, the structure of the liabilities section of the statement distinguishes between four different subsidiary organizations within IMA and Affiliates, Inc. One way to handle subsidiaries is to create an entity-specific taxonomy and embed the names of the four individual subsidiaries into the names of the items.

While this is simple and fits completely within the current framework, anything which is so specific to an organization destroys comparability. A more general solution would be to extend the current notion of taxonomy so that it can accommodate a taxonomy of subsidiary organization rollups.

```
Example: (from ima-00-04-04.xsd):
<!-- Intended Treatment of Subsidiaries, not Supported -->
<!-- Requires (at least) addition of an "entity" data type to Metamodel -->

<element name="IMAandAffiliates" type="entity"/>
<element name="IMA" type="entity"/>
<element name="ICMA" type="entity"/>
<element name="IMAMEF" type="entity"/>
<element name="IMAFAR" type="entity"/>

<links>
<label href="IMA" xml:lang=en>IMA</label>
<label href="ICMA" xml:lang=en>ICMA</label>
<label href="IMAMEF" xml:lang=en>IMAMEF</label>
<label href="IMAFAR" xml:lang=en>IMAFAR</label>

<rollup from="IMA" to="IMAandAffiliates" sense="add" order="1"/>
<rollup from="ICMA" to="IMAandAffiliates" sense="add" order="2"/>
<rollup from="IMAMEF" to="IMAandAffiliates" sense="add" order="3"/>
<rollup from="IMAFAR" to="IMAandAffiliates" sense="add" order="4"/>
</links>

<element name="netAssets.currentOperatingFund" type="monetary"/>
<element name="netAssets.reserveFund" type="monetary"/>
<element name="netAssets.boardDesignated" type="monetary"/>
<element name="netAssets.undesignated" type="monetary"/>
```

```
<links>
<label href="netAssets.currentOperatingFund"
       xml:lang=en>Current Operating Fund</label>
<label href="netAssets.reserveFund"
       xml:lang=en>Reserve Fund</label>
<label href="netAssets.boardDesignated"
       xml:lang=en>Board Designated</label>
<label href="netAssets.undesignated"
       xml:lang=en>Undesignated</label>

<rollup from="netAssets.currentOperatingFund"
        to="liabilitiesandStockholdersEquity.netAssets"
        sense="add" order="1"/>
<rollup from="netAssets.reserveFund"
        to="LiabilitiesandStockholdersEquity.netAssets"
        sense="add" order="2"/>
<rollup from="netAssets.boardDesignated"
        to="liabilitiesandStockholdersEquity.netAssets"
        sense="add" order="3"/>
<rollup from="netAssets.undesignated"
        to="liabilitiesandStockholdersEquity.netAssets"
        sense="add" order="4"/>
</links>
```

6.6 Items and Groups

Although this section is about taxonomies, it is important to see how the newly created taxonomy is referred to in the instance document. Note that the outer group uses the default US GAAP CI taxonomy, while the second inner group uses the IMA taxonomy, which is assumed here to be hosted on the IMA web site.

```
Example (from ima-00-04-04.xml):
```

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE group SYSTEM "xbrl-core-00-04-04.dtd">
<group entity="IMA"
       xmlns="http://www.xbrl.org/core/00-04-04"
       xmlns:aicpa="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04"
       schemaLocation="http://www.xbrl.org/us/aicpa-us-gaap-ci-00-04-04.xsd"
       decimalPattern="#,###" scaleFactor="3" precision="5">

    <group type="balanceSheet.liabilitiesAndStockholdersEquity">
      <item period="1998-06-30">28276</item>
      <item period="1999-06-30">27611</item>
    </group>
<!-- Liabilities and Stockholders Equity -->
    <group xmlns:ima="http://www.ima.org/xbrl/00-04-04"
           schemaLocation="http://www.ima.org/xbrl/ima-00-04-04.xsd">

    <group type="ima:liabilitiesAndStockholdersEquity.liabilities">
      <item period="1998-06-30">6261</item>
      <item period="1999-06-30">6738</item>
    </group>
    <group type="ima:liabilities.accountsPayableAndAccruedExpenses">
      <item period="1998-06-30">2686</item>
      <item period="1999-06-30">3356</item>
    </group>
    <group type="ima:liabilities.bondsPayable">
      <item period="1998-06-30">3575</item>
      <item period="1999-06-30">3382</item>
    </group>
    <group type="ima:liabilitiesAndStockholdersEquity.deferredRevenues">
      <item period="1998-06-30">4040</item>
      <item period="1999-06-30">4555</item>
    </group>
    <group type="ima:deferredRevenues.membershipDues">
      <item period="1998-06-30">3530</item>
      <item period="1999-06-30">4083</item>
    </group>
    <group type="ima:deferredRevenues.other">
      <item period="1998-06-30">510</item>
      <item period="1999-06-30">472</item>
    </group>
    <group type="ima:liabilitiesandStockholdersEquity.netAssets">
      <item period="1998-06-30">17975</item>
      <item period="1999-06-30">16318</item>
    </group>
    <group type="ima:netAssets.currentOperatingFund" entity="IMA">
      <item period="1998-06-30">5986</item>
      <item period="1999-06-30">4020</item>
    </group>
    <group type="ima:netAssets.reserveFund" entity="IMA">
      <item period="1998-06-30">8658</item>
      <item period="1999-06-30">8708</item>
    </group>
    <group type="ima:liabilitiesAndStockholdersEquity.netAssets"
           entity="ICMA">
      <item period="1998-06-30">201</item>
      <item period="1999-06-30">-35</item>
    </group>
    <group type="ima:netAssets.boardDesignated" entity="IMAMEF">
      <item period="1998-06-30">100</item>
      <item period="1999-06-30">100</item>
    </group>
    <group type="ima:netAssets.undesignated" entity="IMAMEF">
      <item period="1998-06-30">3030</item>
      <item period="1999-06-30">3455</item>
    </group>
    <group type="ima:liabilitiesAndStockholdersEquity.netAssets"
           entity="IMAFAR">
      <item period="1998-06-30">0</item>
      <item period="1999-06-30">70</item>
    </group>
  </group>
</group>
```

[Entity Rollup] Entity rollups (and period rollups) will be very useful, but what namespace are they in? We need to decide whether all attributes share a single schemaLocation attribute, or do they each have their own? In the long run, clearly it would be necessary for each attribute to refer to a different schema, so we need to design for this.

7 References (Non-normative)

This is a partial list of key references.

[CANONICAL] Bosworth, A., A. Layman and M. Rys. Serializing Graphs of Data in XML. BizTalk.org Library, Microsoft Corporation, 1999.

[SCHEMA-0] World Wide Web Consortium. XML Schema Part 0: Primer.

[SCHEMA-1] World Wide Web Consortium. XML Schema Part 1: Structures.

[SCHEMA-2] World Wide Web Consortium. XML Schema Part 2: Datatypes.

8 Issues

This is an alphabetical list of issues specifically enumerated in this document.

[Alternate Breakdowns]

[Entity Rollup]

[Everything Repeatable]

[Footnote References]

[Flush Links]

[Instance Includes]

[Instance Rationale]

[Label Processing]

[Long Names]

[Taxonomy Extensions]

[Unique Elements]

9 Change Log

00-04-02 [Hamscher] In the taxonomy, eliminated "total" from element names or changed them to "gross" as appropriate. In the taxonomy, changed "cash flow" to "cash flows". In the taxonomy, changed "intangible assets" in long term assets to "intangibles". Added additional examples of the period attribute. Deleted the [Instance Rationale] note, since the design rationale discussion covers all the necessary points. Removed the [Style Everywhere] note, since we have a current compromise which allows the group element to contain elements other than items. Added section discussing the meaning of "period" and why a specific date and duration is a good idea. Added section discussing prior period balances and how that interacts with taxonomies. Added note on alternate breakdowns. Added cautionary note about applications assuming duration. Fixed all the capitalization problems in the examples to agree with 00-04-04 release of the files.

00-03-29 [Hamscher] Miscellaneous typo corrections. Continuing repairs to text that concerns the fact that of markup is forbidden inside items. Changed all "CamelCase" names to "camelCase". Added an additional paragraph explaining the "sense" attribute. Checked for references to "footnote" that should have been references to Notes. Added the [Long Names] note.

00-03-28 [Hamscher] Added the "pure" datatype, deleted the [unit examples] issue. Reverted to original explanation of the item tag disallowing embedded markup. Changed wording of the paragraph contrasting namespaces with the schemaLocation attribute. Added [Instance Includes] suggestion raised by David Vun Kannon. Added explanation of parsing implications of decimalPattern. Got rid of the [Time Duration] issue and changed to an explanation that we are differing from XML Schema convention. Miscellaneous typo corrections.

00-03-24 [Hamscher] Changed text references to "taxonomy attribute" to schemaLocation. Fixed typo in example of 3.12. Fixed the period definition with a better reference for ISO 8601 than the incomplete summary given in the W3C material. Miscellaneous typo corrections.

00-03-23 [Hamscher] Added change log. Changed "taxonomy" to schemaLocation. Repaired broken definition of period attribute, raised new timeDuration issue. Included new "unique elements" issue. Raised issue of deleting "links". Added XML Schema: Primer reference. Changed text of the Unit Examples text, fixing the Moody's example and removing the PURE example. Added issue regarding label processing. Got rid of the Parents Required issue, left the discussion. Added historical notes regarding the fundamental decisions agreed to at the Chicago meeting. Changed scalefactor to scaleFactor. Changed taxonomy to schemaLocation. Added distinction between financial presentation and accounting, in the context of order independence. Similar distinction with respect to negative balances. Added discussion of the unique naming issue. Fixed the non-negative-integer datatype of order. Added taxonomy extensions issue, from Eric Cohen. Miscellaneous typo corrections.

00-03-19 [Hamscher] First released version.