Proposed Directions for *xmlmortgage.org* Engineering

VARIOUS PROPOSALS AND ENGINEERING STANDARDS

1. Naming convention

Proposal: Elements that may contain other elements should be all caps. Elements that are at the data level (i.e., those of #PCDATA type) should be first-letter, Microsoft-style capitalized.

Example:

```
<LOAN>
 <BORROWER>
  <SocialSecurityNumber>123-45-
6789</SocialSecurityNumber>
  <FirstName>George</FirstName>
 </BORROWER>
</LOAN>
```

2. Data typing

Proposal: For each data-level element, we describe the data type: Enumerated, Number, String, or Date; the permissible range of values (using the half-open Microsoft range notation for ranged elements, or a list of possible values for enumerated elements); data size information (number of bytes for number elements, or number of characters for string elements); and any other descriptive information (such as expected formats for dates and formatted strings like SSNs).

3. Elements vs. Attributes

Proposal: Only elements will be used – no attribute information will be present. While this will prevent us from specifying enumerated information as attributes in DTDs, it will not solve the more basic problem of constraint management. By making the data points elements and including comments in the DTDs describing the allowable values and meanings for enumerated columns, a parser does not need to make the distinction between enumerated data (attributes) and string, numeric, or date data (elements). Note that we are making the distinction here between descriptive attributes - that is, attributes that are used to describe the type of information contained in an element - and informational attributes. We want to avoid element declarations in valid XML documents of the form:

```
<ASSET AccountNumber="123456789"></ASSET>
```

4. Key values

Proposal: Parent-child relationships should be shown via element containment (since the data design will be snowflake in character) – this allows us to remain system-independent and avoid cumbersome key translation issues. In those cases where using identifiers is unavoidable, the special element types ID and IDREF should be used to connect elements together.

5. Element Normalization Strategy

Proposal: Required data-level elements should not be broken out of their parents if they only belong to one parent. Optional data-level elements with only one parent may be broken out into separate parent elements or remain as optional data-level elements in their parents (without an extraneous parent element interposed). Data-level elements that may have more than one parent should always be broken out from their parents, as should data-level elements that may occur multiple times for the same parent.

6. Enumerated element definition

Proposal: The elected data typing mechanism used to describe the specific content of data elements should provide some sort of descriptive mechanism for enumerated values. Given the relative verbosity of XML, we should probably go with descriptive values for these fields as well – for example, "SingleFamilyResidence" instead of "SFR" or "S".

7. The Purpose and Role of Namespaces

Proposal: We have one namespace: www.xmlmortgage.org. This namespace would then be specified in all entity declarations in valid documents prepared by users of the specification. As far as duplicate element names in our namespace go, we have two options: one would be to have some repository of data-level elements (in the relational database world, known as columns) that are then referred to by grouping elements that describe their relationships. So, we might have:

```
<!ELEMENT Person (#PCDATA)>
<!ELEMENT Street (#PCDATA)>
<!ELEMENT LemonadePrice (#PCDATA)>
<!ELEMENT SaleDate (#PCDATA)>
<!ELEMENT SalePrice (#PCDATA)>
<!ELEMENT BUYER (Person)>
<!ELEMENT SELLER (Person)>
<!ELEMENT LEMONADESALES (BUYER, SELLER, SaleDate,
SalePrice)
<!ELEMENT LEMONADESTAND (Street, LEMONADESALES*)>
```

A second option (and a better one, in our opinion) would be to give similar attributes different names depending on the element space they reside in, as in the following:

```
<!ELEMENT Buyer (#PCDATA)>
<!ELEMENT Seller (#PCDATA)>
<!ELEMENT Street (#PCDATA)>
<!ELEMENT LemonadePrice (#PCDATA)>
<!ELEMENT SaleDate (#PCDATA)>
<!ELEMENT SalePrice (#PCDATA)>
<!ELEMENT LEMONADESALES (Buyer, Seller, SaleDate,
SalePrice)>
<!ELEMENT LEMONADESTAND (Street, LEMONADESALES*)>
```

If we are planning to support XML Schemas (see below), the Buyer and Seller elements (in this example) could be inherited from a Person archetype. Each could then contain information specific to its role, such as a thirst rating for the buyer and a greediness rating for the seller.

Please note that the Namespace specification is under heavy debate by the W3C at this time, and its future role is not clearly defined. Potentially, if namespaces are adopted and formalized, they could be used to help discriminate between (for example) different financial institution's methods for calculating APR. For now, however, we should assume that all data elements reside in one namespace, and

plan accordingly.

8. Element Description and Documentation

Proposal: We have previously discussed creating a data dictionary and having it be the main repository for the information about the database, and having XML be automatically generated from this dictionary. We would prefer to continue with this approach, because it will allow us to also automatically generate an XML Schema (see below). We believe it's important that all documents we provide, including the XML Schema, the DTDs, and the data dictionary in Excel, provide (if only in comment form) information about the elements they contain. This should include, but not necessarily be limited to, the following:

- Data type (including size information)
- Enumerated values (for enumerated elements)
- Business definition of the element, including business process(es)
- Unit information (for those values that may have units associated with them)

9. XML Schema

Proposal: The process that builds the supporting documents for the XML Mortgage data structures should also construct an XML Schema document. While this format is not yet formally endorsed by the W3C, it appears to be the direction the industry is moving for data-focused XML definitions (strong data typing, true inheritance from parent objects or "archetypes", data constraints, and so on), and will probably replace DTD eventually in data-centric applications of XML. As third party tools become available that take advantage of this type of descriptive document, we will be one step ahead by already providing access to an up-to-date version of the XML Schema for our XML.

Background information on XML schemas may be obtained from

http://www.xml.com/pub/1999/07/schemas/whatis.html.

The W3C's current working draft proposal for XML schemas may be obtained from

http://www.w3.org/1999/05/06-xmlschema-1/

and

http://www.w3.org/1999/05/06-xmlschema-2/.

10. Data File (DTD) Rules

Proposal: Data Item is defined to be a specific data set, closely related around the subject name.

Data point is defined to be a data field defined within (and thus part of) a data item.

- For a data item to be a candidate for the Core Data specification DTD, the data item must span across more than one business process within mortgage.
- For the data item to be a candidate for a business process specification, the data item must be shared functionally across more than one financial institution, but not a core data item.
- For the data item to be a financial institution data item, it need only be submitted by that institution. The proposed naming structure will be financial institution name followed by the data point name, each word capitalized.
- For a data item to be a software vendor data item, it need only be submitted by that software vendor. The proposed naming structure will be software vendor name followed by the data point name, each word capitalized.

Note that namespaces might be useful in making the determination of financial institution or software vendor.