SMART Document Overview: Frequently-Asked Questions

MISMO eMortgage Workgroup SMART Doc Focus Group

Version: 1.0
Date: 10/9/2002

Abstract
This document provides answers to SMART Doc Frequently-Asked Questions in several topic areas.

I. Introduction

The SMART Doc Specification is a technical framework for representing documents in an electronic format. This format links data, the visual representation of the form, and signature. The visual representation of the documents can utilize a variety of technologies, such as XHTML, PDF, and TIFF. A SMART Doc can be secured to prevent tampering. Therefore, the Specification allows system validation to ensure that what the borrower sees and signs on the computer screen is the exact document that will be stored. It also ensures that the data displayed on the screen will be the exact data used for processing the loan.

The development of the SMART Doc specification was based on the following objectives and goals:

1) Develop a smart, electronic document format that binds together data and presentation along with other information in a single, immutable file.

2) Create the capability for maintaining integrity of information in electronic mortgage documents that will support legal and credit policy requirements and designed to comply with UETA and ESIGN legislation.

3) Facilitate interoperability between software systems written by different vendors to create and manage electronic mortgages.

4) Enable automated down-stream processing, and quality and completeness reviews through reconciliation of data and its presentation.

5) Provide for multiple views for flexibility in execution and connection with existing systems.

6) Leverage existing technologies and standards to ease development and implementation.

7) Allow for the electronic packaging of multiple mortgage documents.

The following sections provide answers to Frequently-Asked Questions (FAQs) in the areas of Business, Application, and Technology.

II. Business FAQs

1. What does SMART represent?

   Previously, SMART documents were called eMortgage documents. In order to better describe the actual capabilities of the technology, the word "eMortgage" was replaced. The acronym SMART represents:
   - Securable
   - Manageable
   - Archivable
   - Retrievable
   - Transferable

2. Where does this fit in my business process?

   SMART Docs are applicable anywhere you are currently exchanging documents internally or with business partners. The current trend towards automating paper-based transactions creates the need for standardized electronic documents. Just as the paper uniform instruments create a fungible commodity today, SMART Documents allow for fungible electronic transactions.

   For example, by linking document data and images, SMART Documents eliminate the re-keying of data and facilitate quality assurance during the investor purchase process. SMART Docs facilitate the exchange or verification of data. Specific benefits are determined on a trading partner basis.

3. What systems might gain efficiencies from the use of SMART Docs?

   Any system associated with closing, servicing, delivery to the secondary market, or other forms-related data interchange.

III. Application FAQs

1. What documents can be in SMART Doc format?

   Any document can be in SMART Doc format. Our current focus is on core Closing documents for conforming loans. Any additional docs could be implemented by the SMART Doc specification as desired (for example, application disclosures, insurance, title policies, appraisals, and ancillary products).

2. Can a loan package have SMART Docs and paper docs?

   Yes. Consult investor guidelines for details.

3. Can a loan pool have electronic notes (i..e., SMART Docs) and paper notes?

   Yes. Consult investor guidelines for details.

4. Can electronic notes be delivered for both cash and MBS/Guarantor?

   Yes. Consult investor guidelines for details.

5. Should a SMART Doc be used to hold one large data set and views of multiple docs?

   While the SMART Document Specification allows for the inclusion of multiple documents, the intended implementation is for each SMART Doc to represent a single document.

6. How do I use electronic signatures with SMART Documents?

The current SMART Doc Specification contains the technical requirements for the use of electronic signatures. Detailed instructions on electronic signature use in connection with SMART Documents will be published in an implementation guide in the near future.

IV. Technical FAQs

1. Does each mortgage document (like note, assignment, loan application, etc.) have a separate DTD?

   No. The architectural design of the SMART Doc DTD allows other data sets to "plug in" in a modular fashion.

2. Why does the SMART Document Specification use XHTML instead of HTML?

   HTML is the set of codes in a document that allows the document to be displayed on the World Wide Web. HTML 4 is the current version. HTML documents describe the presentation of information on a computer screen ("a web page in a browser").

   XHTML is the next logical step to HTML. An XHTML document is an HTML document that can be viewed as a web page and can be processed by XML software. XHTML can be modularized and extended as long as the document is a web page (that is, the document begins with <html>). Strictly speaking to the XHTML recommendation from the W3C, all XHTML documents must begin with <html>.

   XHTML documents are XML conforming. They are readily viewed, edited, and validated with standard XML tools. XHTML documents can be written to operate as well or better in existing HTML browsers as well as in new, XHTML 1.0 conforming browsers.

3. Is the SMART Document Specification an XHMTL specification or an XML specification?

   Technically to the true letter of the XML and XHTML specifications, the SMART Document specification is an XML specification. If the embedded XHTML View is extracted from the file, the extracted subset is an XHTML file.

4. Why does the SMART Document Specification use XML?

   XML is a platform, vendor, and application independent technology. It provides rules for describing document content and structure in any type of document. It will ease the transition from the paper "original" to the electronic "original". When combined with Extensible Stylesheet Language (XSL), XML is a technology that represents information in both an electronic and printed form.

   - XML is a technology that meets the goals and objectives of electronic mortgage documents:
   - XML allows for a smart, electronic document format that binds together data and presentation along with other information in a single, immutable file.
   - XML provided for the exchange of data across diverse systems and does not require proprietary software.
   - XML information allows for additional software to be developed that will permit a lights-out quality and completeness review of the information.
   - XML enables automated feeds of data for downstream processing
   - XML provides processing flexibility and allows for multiple views or presentations of the information
   - The use of XML leverages existing technologies and standards to ease development and implementation

5. What is XSL?

The XSL specification defines the Extensible Stylesheet Language (XSL). XSL is a language for expressing stylesheets. It consists of two parts: XSL Transformations (XSLT), a language for transforming XML documents to other representation such as XHTML, and an XML vocabulary for specifying formatting semantics (XSL Formatting Objects) for pagination and print-based formatting.

Given an XML document, designers use an XSL stylesheet to express their intentions about how that structured content should be presented; i.e., how the source content should be styled, laid out, and paginated onto some presentation medium, such as a window in a Web browser or a set of physical print pages.

The XSL specification defines the syntax and semantics of XSLT, which is a language for transforming XML documents into other XML documents, including XHTML.

An XSL stylesheet processor accepts a document or data in XML and an XSL stylesheet and produces the presentation of that XML source content that was intended by the designer of that stylesheet. The software that performs the transformation may be client-side in Internet Explorer (versions 5.0 or higher) or may be on a server.

6. What was the philosophy behind a single file format that includes the XML data linked to the XHTML view?

   A SMART Document must include both the XML information and the view of the information to maintain the integrity of what was seen on the computer screen by the person signing the document and what is used later by downstream processing systems. Linking the XML data and the XHTML view provides a mechanism to "trust" the document. Including the XML data and the XHTML in a single immutable file is a conservative approach to electronic documents. No legal precedent exists for SMART Documents. The SMART Document specification was developed to ensure that the electronic document is as valid as a paper document.

7. Isn't the information redundant by being in both the data and view sections?

   Yes. This redundancy maintains the integrity of what was seen on the computer screen. In the following example, "tenth" is displayed and "10" is used by down stream processing systems.

   For instance, the day of the month may be represented in the XML data section as:

   <EXECUTION_Day="10">

   and in the View section as:

   <p> This BALLOON NOTE ADDENDUM is made this tenth day of … </p>

8. Why doesn't the SMART Document specification use XML with an XSLT stylesheet?

   XSL could be used to dynamically generate what was previously viewed on the computer screen. It is possible within the current specification to support XSLT. However the inclusion of XSL is targeted for a future version of the specification due to overcoming technical issues with validation and conversion of the dynamically generated view. The MISMO working group needs to resolve how an XSLT stylesheet will be validated and how standard conversions will be performed in XSLT. Validation and consistently generated conversions will ensure that what was seen on the computer screen the first time is immutably the same as those generated at a later date.

XSLT validation will provide a guarantee that post-processing of the document results in correct and consistent generation from verifiable source fields.

The XSLT script must generate the document correctly and generate the same document each time it is executed. This problem arises because XSLT is a powerful language with conditional expressions and looping constructs. Here is a snippet of an XSLT script with a conditional:

```
<xsl:if test=”count(//AUDIT_TRAIL/Entry) > 3”>
    <p> Property value is $200,000.000 </p>
</xsl:if>
<xsl:if test=”count(//AUDIT_TRAIL/Entry) <= 3”>
    <p> Property value is $215,000.00 </p>
</xsl:if>
```

The above snippet of XSLT checks the number of entries in the audit trail and generates a different result when the number of entries is greater than 3. Thus, this XSLT script acts like a sleeping menace, waiting for its preset-condition to be triggered.

One potential way to eliminate the need for correctness validation is to archive the result generated by the XSLT Script. This eliminates the need to re-run the XSLT; however, this does not provide a guarantee that the values in the XML portion of the document (the DATA section) were the values used to generate the result (see the next paragraph). Additionally, archiving will significantly increase the size of files. Its performance impact must be analyzed.

The second requirement is the XSLT script extracts the values to generate the HTML presentation from the XML data section (called DATA) that will be fed to the back-end systems. In other words, since the intent is to take action on the XML, you need to guarantee the values in the XML (especially critical attribute values such as the loan amount) match the value in the generated output. In the current specification this is achieved by specific intra-document links (called ARCs) that can be traced. With XSLT this guarantee becomes more difficult to ensure. One potential method would be to ensure the elements in the XML section were accessed via “select” or “match” attributes; however, while that guarantees the value was used it does not guarantee the value generated the output (especially if a conversion was necessary). It is these types of validation complexities that need to be resolved prior to implementing the XSLT specification.

Additionally, standard conversions like translating numbers to their ordinal equivalent, formatting a date, formatting money and many others would need to be developed in XSLT and standardized in order to provide the validation guarantees discussed above.

9. Is it possible to use XSLT to generate a specific document state?

    Yes. The specification leaves the generation of each standard state up to the implementer. XSLT could easily be used to create the View sections of the SMART document.

10. Why doesn’t the SMART Document specification use standalone XHTML with the XML data tags embedded in the XHTML and eliminate the need for linking?

    While standalone XHTML may include data elements inside the presentation elements, this constrains the XML to the document’s presentation. For example, if there is a LOAN element with attributes for value and interest rate, how is that represented in a document that contains the information in different paragraphs?

```
<p> This property will sell for <LOAN amount=”100,000.00”> $100,000 </LOAN> dollars.
</p>
<p> more stuff here … the borrower will … </p>
```

```
<p> the property is located at … <PROPERTY street = "111 anywhere" … /> 111
anywhere lane, anywhere 112211<p>
<p> The interest rate of the loan will be <LOAN rate="8.5"> 8.5 </LOAN> percent. </p>
```

The above example has two serious problems: first, it would be preferable to have a single LOAN element. Secondly, it is not possible to assume any ordering to the XML elements since different document presentations and types will order the information differently. It is not possible to assume any cardinality (such as only one LOAN element). The most serious restriction is that XHTML favors an element-only approach to the exclusion of attributes that are useful. There are many XML representations that cannot be represented within an XHTML view due to the restrictions demonstrated above (and others).

11. Do you support other image formats like tiff?

    No. At this time we only support the GIF and JPEG file formats.

12. What are the Meta elements in the header section for?

    The Meta element allows you to include any additional metadata about the document that is not covered in the specification. The element has the same form as the HTML Meta element. For example, to add a system-specific identifier to a document, use the Meta element in the following way:

```
<SMARTDOC>
  <HEADER>
   …
    <Meta name="systemId"   content="cb01-3987" />
  </HEADER>
</SMARTDOC>
```

13. Is it possible to attach a signature to a non-XHTML view?

    Yes, it is possible by the proper combined use of the SIG_MODEL with the View. The SIG_MODEL links the signature "target" and the signature itself. For example:

```
<SMARTDOC>
  <HEADER>
    <SIG_MODEL>
      <Signer targets="V01" signature="BorrowerSig" />
    </SIG_MODEL>
  </HEADER>
  <VIEW Id="V01" type="image/jpeg" encoding="base64">
   aosnuthosatuhrchsnthaoesutheontuhuchuhsuchasokbkhddrc
   asctehgul.chustuhachuskbkschckaosntuhckhnstkhenthkrcne
   …
  </VIEW>
  <SIGNATURES>
    <Signature id="BorrowerSig" > … <!-- digital signature -->
  </SIGNATURES>
</SMARTDOC>
```

14. Can proprietary elements be used in a SMART Document?

    Yes. The DATA section may contain a CUSTOM element that has no restrictions on its content. You may place proprietary XML elements within that CUSTOM element.