



**eRecording XML
Implementation Guide
For Version 2.4.1**

**Revision 2
Updated 03/05/2007**

Document Date	Changes By	Change List
		Pre-publication drafting efforts
7/12/04	Charlie Epperson, John Jones, Carol Foglesong	Change MISMO template to PRIA, etc.
10/21/05	Charlie Epperson, Mark Ladd	Updated all chapters and prepared release candidate for work group.
10/24/05	Mark Ladd	Updated chapter 6 with proper column and organization for TOC corrections and added updates to overall chapters and flows
10/26/05	Charlie Epperson	Updated overall chapters and flows with PRIA standard updates and updated DTD/Schema layouts
5/16/2006	Mark Ladd	Updated chapters 1-5 based on XML WG review
6/01/06	Charlie Epperson	Updated the examples to match PRIA tags/content
8/02/06	Mark Ladd	Updated chapter 6 to include final published elements, XML WG comments and formatting changes.
9/11/06	Mark Ladd	Posted for IPR review
10/11/06	Mark Ladd	Passed IPR review without objection
3/5/2007	Mark Ladd	Added Security chapter – Posted for IPR review

Copyright 2004-2007 – Property Records Industry Association (PRIA)

All rights reserved

Permission to use, copy, modify, and distribute the PRIA DTD and its accompanying documentation for any purpose and without fee is hereby granted in perpetuity, provided that the above copyright notice and this paragraph appear in all copies. The copyright holders make no representation about the suitability of the DTD or documentation for any purpose.

XML Implementation Guide: General Information – Version 2

Document Revision Date: August 2, 2006

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION.....	1-1
PURPOSE OF THIS DOCUMENT.....	1-1
WHAT ELSE YOU WILL NEED.....	1-1
WHAT IS XML?.....	1-2
WHAT IS A DTD?.....	1-4
WHAT IS AN LDD?	1-6
WHAT IS PRIA?	1-7
COMMENTS.....	1-8
CHAPTER 2: PRIA XML ARCHITECTURE OVERVIEW	2-1
TRANSITIONING FROM PRIA VERSION 1 TO PRIA VERSION 2.....	2-1
ELEMENTS AND ATTRIBUTES: THE DTD BUILDING BLOCKS.....	2-3
CONTAINMENT AND POINTING RELATIONSHIPS.....	2-9
CHAPTER 3: THE PRIA TRANSACTION ENVELOPE	3-11
THE REQUEST ENVELOPE	3-11
THE RESPONSE ENVELOPE.....	3-18
CHAPTER 4: XML IMPLEMENTATION ISSUES.....	4-1
XML RESERVED CHARACTERS	4-1
SPECIFYING THE DTD IN THE XML DATA FILE.....	4-3
WHITE SPACE IN XML DOCUMENTS	4-4
HANDLING ATTRIBUTES WITH NO DATA.....	4-5
ORDERING ELEMENTS AND ATTRIBUTES	4-5
CHAPTER 5: HTTP POST NAME/VALUE PAIR RECOMMENDATION	5-1
CHAPTER 6: SECURITY PRINCIPLES.....	6-2
GENERAL SECURITY PRINCIPLES	6-2
<i>Authentication</i>	6-2
<i>Confidentiality</i>	6-2
<i>Integrity</i>	6-3
<i>Non-repudiation</i>	6-3
<i>Privileged-Based Access Control</i>	6-4
GENERAL RECOMMENDATIONS FOR SECURELY TRANSPORTING SENSITIVE RECORDING INFORMATION.....	6-5
SECURE MESSAGING REQUIREMENTS FOR SUPPORTING ELECTRONIC RECORDING PROCESSES	6-6
RECOMMENDED SECURITY SOLUTIONS	6-7
TRADING PARTNER-TO-TRADING PARTNER DIRECT SCENARIO	6-8
<i>Security Requirements</i>	6-8
<i>Possible Security Solutions</i>	6-8
TRADING PARTNERS THROUGH A PORTAL SCENARIO.....	6-10
<i>Security Requirements (Variation 1)</i>	6-11
<i>Security Requirements (Variation 2):</i>	6-11
<i>Possible Security Solutions</i>	6-12
MULTI-SERVICES SCENARIO.....	6-13
<i>Security Requirements</i>	6-13
<i>Possible Security Solutions</i>	6-14
SECURITY DEFINITIONS	6-16

CHAPTER 7: PRIA DTD BEST PRACTICES	7-1
PRIA DTD	7-1
[PRIA_DOCUMENT] ELEMENT	7-2
PRIA_DOCUMENT DTD DEFINITION	7-2
PRIA_DOCUMENT EXAMPLES.....	7-2
PRIA_DOCUMENT ATTRIBUTES	7-3
PRIA_DOCUMENT DEPRECATIONS	7-3
[GRANTOR] ELEMENT	7-4
GRANTOR DTD DEFINITION.....	7-4
GRANTOR EXAMPLES	7-4
GRANTOR ATTRIBUTES	7-4
GRANTOR DEPRECATIONS.....	7-6
[_ALIAS] ELEMENT.....	7-7
_ALIAS DTD DEFINITION	7-7
_ALIAS EXAMPLES	7-7
_ALIAS ATTRIBUTES.....	7-7
_ALIAS DEPRECATIONS	7-8
[GRANTEE] ELEMENT	7-9
GRANTEE DTD DEFINITION	7-9
GRANTEE EXAMPLES.....	7-9
GRANTEE ATTRIBUTES.....	7-9
GRANTEE DEPRECATIONS	7-11
[PROPERTY] ELEMENT	7-12
PROPERTY DTD DEFINITION.....	7-12
PROPERTY EXAMPLES	7-12
PROPERTY ATTRIBUTES	7-13
PROPERTY DEPRECATIONS.....	7-14
[_IDENTIFICATION] ELEMENT.....	7-15
_IDENTIFICATION DTD DEFINITION.....	7-15
_IDENTIFICATION EXAMPLES	7-15
_IDENTIFICATION ATTRIBUTES.....	7-15
_IDENTIFICATION DEPRECATIONS	7-17
[PARSED_STREET_ADDRESS] ELEMENT.....	7-18
PARSED_STREET_ADDRESS DTD DEFINITION	7-18
PARSED_STREET_ADDRESS EXAMPLES	7-18
PARSED_STREET_ADDRESS ATTRIBUTES.....	7-18
PARSED_STREET_ADDRESS DEPRECATIONS	7-19
[_LEGAL_DESCRIPTION] ELEMENT	7-20
_LEGAL_DESCRIPTION DTD DEFINITION	7-20
_LEGAL_DESCRIPTION EXAMPLES	7-20
_LEGAL_DESCRIPTION ATTRIBUTES.....	7-20
_LEGAL_DESCRIPTION DEPRECATIONS	7-21
[PARCEL_IDENTIFICATION] ELEMENT	7-22
PARCEL_IDENTIFICATION DTD DEFINITION.....	7-22
PARCEL_IDENTIFICATION EXAMPLES	7-22
PARCEL_IDENTIFICATION ATTRIBUTES	7-22
PARCEL_IDENTIFICATION DEPRECATIONS	7-23
[PLATTED_LAND] ELEMENT.....	7-24
PLATTED_LAND DTD DEFINITION	7-24
PLATTED_LAND EXAMPLES	7-24

PLATTED_LAND ATTRIBUTES.....	7-25
PLATTED_LAND DEPRECATIONS	7-26
[UNPLATTED_LAND] ELEMENT	7-27
UNPLATTED_LAND DTD DEFINITION.....	7-27
UNPLATTED_LAND EXAMPLES.....	7-27
UNPLATTED_LAND ATTRIBUTES	7-27
UNPLATTED_LAND DEPRECATIONS.....	7-29
[PARTIES] ELEMENT.....	7-30
PARTIES DTD DEFINITION.....	7-30
PARTIES EXAMPLES	7-30
PARTIES ATTRIBUTES	7-30
PARTIES DEPRECATIONS	7-30
[_RETURN_TO_PARTY] ELEMENT.....	7-31
_RETURN_TO_PARTY DTD DEFINITION.....	7-31
_RETURN_TO_PARTY EXAMPLES	7-31
_RETURN_TO_PARTY ATTRIBUTES	7-31
_RETURN_TO_PARTY DEPRECATIONS.....	7-32
[PREFERRED_RESPONSE] ELEMENT	7-33
PREFERRED_RESPONSE DTD DEFINITION.....	7-33
PREFERRED_RESPONSE EXAMPLES	7-33
PREFERRED_RESPONSE ATTRIBUTES	7-33
PREFERRED_RESPONSE DEPRECATIONS.....	7-34
[NON_PERSON_ENTITY_DETAIL] ELEMENT	7-35
NON_PERSON_ENTITY_DETAIL DTD DEFINITION	7-35
NON_PERSON_ENTITY_DETAIL EXAMPLES.....	7-35
NON_PERSON_ENTITY_DETAIL ATTRIBUTES	7-35
NON_PERSON_ENTITY_DETAIL DEPRECATIONS.....	7-36
[AUTHORIZED_REPRESENTATIVE] ELEMENT	7-37
AUTHORIZED_REPRESENTATIVE DTD DEFINITION	7-37
AUTHORIZED_REPRESENTATIVE EXAMPLES	7-37
AUTHORIZED_REPRESENTATIVE ATTRIBUTES.....	7-37
AUTHORIZED_REPRESENTATIVE DEPRECATIONS	7-37
[CONTACT_DETAIL] ELEMENT.....	7-38
CONTACT_DETAIL DTD DEFINITION.....	7-38
CONTACT_DETAIL EXAMPLES	7-38
CONTACT_DETAIL ATTRIBUTES	7-38
CONTACT_DETAIL DEPRECATIONS	7-38
[CONTACT_POINT] ELEMENT	7-39
CONTACT_POINT DTD DEFINITION	7-39
CONTACT_POINT EXAMPLES.....	7-39
CONTACT_POINT ATTRIBUTES	7-39
CONTACT_POINT DTD DEPRECATIONS.....	7-40
[_PREPARED_BY_PARTY] ELEMENT	7-41
_PREPARED_BY_PARTY DEFINITION	7-41
_PREPARED_BY_PARTY EXAMPLES	7-41
_PREPARED_BY_PARTY ATTRIBUTES.....	7-41
_PREPARED_BY_PARTY DEPRECATIONS	7-42
[TAXABLE_PARTY] ELEMENT.....	7-43
TAXABLE_PARTY DTD DEFINITION	7-43
TAXABLE_PARTY EXAMPLES.....	7-43

TAXABLE_PARTY ATTRIBUTES	7-43
TAXABLE_PARTY DEPRECIATIONS	7-44
[BILL_TO_PARTY] ELEMENT	7-45
BILL_TO_PARTY DTD DEFINITION	7-45
BILL_TO_PARTY EXAMPLES	7-45
BILL_TO_PARTY ATTRIBUTES	7-45
BILL_TO_PARTY DEPRECIATIONS	7-46
[WITNESS] ELEMENT	7-47
WITNESS DTD DEFINITION	7-47
WITNESS EXAMPLES	7-47
WITNESS ATTRIBUTES	7-47
WITNESS DEPRECIATIONS	7-47
[EXECUTION] ELEMENT	7-48
EXECUTION DTD DEFINITION	7-48
EXECUTION EXAMPLES	7-48
EXECUTION ATTRIBUTES	7-48
EXECUTION DEPRECIATIONS	7-48
[MORTGAGE_CONSIDERATION] ELEMENT	7-49
MORTGAGE_CONSIDERATION DTD DEFINITION	7-49
MORTGAGE_CONSIDERATION EXAMPLES	7-49
MORTGAGE_CONSIDERATION ATTRIBUTES	7-49
MORTGAGE_CONSIDERATION DEPRECIATIONS	7-49
[CONSIDERATION] ELEMENT	7-50
CONSIDERATION DTD DEFINITION	7-50
CONSIDERATION EXAMPLES	7-50
CONSIDERATION ATTRIBUTES	7-50
CONSIDERATION DEPRECIATIONS	7-50
[RECORDABLE_DOCUMENT] ELEMENT	7-51
RECORDABLE_DOCUMENT DTD DEFINITION	7-51
RECORDABLE_DOCUMENT EXAMPLES	7-51
RECORDABLE_DOCUMENT ATTRIBUTES	7-51
RECORDABLE_DOCUMENT DEPRECIATIONS	7-51
[_ASSOCIATED_DOCUMENT] ELEMENT	7-52
_ASSOCIATED_DOCUMENT DTD DEFINITION	7-52
_ASSOCIATED_DOCUMENT EXAMPLES	7-52
_ASSOCIATED_DOCUMENT ATTRIBUTES	7-52
_ASSOCIATED_DOCUMENT DEPRECIATIONS	7-54
[SIGNATORY] ELEMENT	7-55
SIGNATORY DTD DEFINITION	7-55
SIGNATORY EXAMPLES	7-55
SIGNATORY ATTRIBUTES	7-55
SIGNATORY DEPRECIATIONS	7-56
[ELECTRONIC_SIGNATURE] ELEMENT	7-57
ELECTRONIC_SIGNATURE DTD DEFINITION	7-57
ELECTRONIC_SIGNATURE EXAMPLES	7-57
ELECTRONIC_SIGNATURE ATTRIBUTES	7-57
ELECTRONIC_SIGNATURE DEPRECIATIONS	7-57
[TEXT_SIGNATURE] ELEMENT	7-58
TEXT_SIGNATURE DTD DEFINITION	7-58
TEXT_SIGNATURE EXAMPLES	7-58

TEXT_SIGNATURE ATTRIBUTES	7-58
TEXT_SIGNATURE DEPRECATIONS	7-58
[EMBEDDED_SIGNATURE_FILE] ELEMENT	7-59
EMBEDDED_SIGNATURE_FILE DTD DEFINITION	7-59
EMBEDDED_SIGNATURE_FILE EXAMPLES	7-59
EMBEDDED_SIGNATURE_FILE ATTRIBUTES.....	7-59
[DOCUMENT] ELEMENT	7-61
DOCUMENT ELEMENT DTD DEFINITION	7-61
DOCUMENT ELEMENT EXAMPLES	7-61
DOCUMENT ELEMENT DTD ATTRIBUTES.....	7-61
DOCUMENT ELEMENT DEPRECATIONS	7-61
[EXTERNAL_SIGNATURE_FILE] ELEMENT.....	7-62
EXTERNAL_SIGNATURE_FILE DTD DEFINITION	7-62
EXTERNAL_SIGNATURE_FILE EXAMPLES	7-62
EXTERNAL_SIGNATURE_FILE ATTRIBUTES.....	7-62
EXTERNAL_SIGNATURE_FILE DEPRECATIONS	7-63
[OTHER_SIGNATURE] ELEMENT	7-64
OTHER_SIGNATURE DTD DEFINITION	7-64
OTHER_SIGNATURE EXAMPLES	7-64
OTHER_SIGNATURE ATTRIBUTES.....	7-64
OTHER_SIGNATURE DEPRECATIONS	7-64
[NOTARY] ELEMENT.....	7-65
NOTARY DTD DEFINITION	7-65
NOTARY EXAMPLES.....	7-65
NOTARY ATTRIBUTES.....	7-65
NOTARY DEPRECATIONS	7-66
[_CERTIFICATE] ELEMENT	7-67
_CERTIFICATE DTD DEFINITION.....	7-67
_CERTIFICATE EXAMPLES	7-67
_CERTIFICATE ATTRIBUTES	7-67
_CERTIFICATE DEPRECATIONS.....	7-68
[_SIGNER_IDENTIFICATION] ELEMENT	7-69
_SIGNER_IDENTIFICATION DTD DEFINITION.....	7-69
_SIGNER_IDENTIFICATION EXAMPLES	7-69
_SIGNER_IDENTIFICATION ATTRIBUTES	7-69
_SIGNER_IDENTIFICATION DEPRECATIONS	7-69
[RECORDING_ENDORSEMENT] ELEMENT.....	7-70
RECORDING_ENDORSEMENT DTD DEFINITION	7-70
RECORDING_ENDORSEMENT EXAMPLES	7-70
RECORDING_ENDORSEMENT ATTRIBUTES.....	7-70
RECORDING_ENDORSEMENT DEPRECATIONS	7-71
[_VOLUME_PAGE] ELEMENT	7-72
_VOLUME_PAGE DTD DEFINITION.....	7-72
_VOLUME_PAGE EXAMPLES	7-72
_VOLUME_PAGE ATTRIBUTES	7-72
_VOLUME_PAGE DEPRECATIONS.....	7-72
[_FEES] ELEMENT	7-73
_FEES DTD DEFINITION.....	7-73
_FEES EXAMPLES	7-73
_FEES ATTRIBUTES	7-73

_FEES DEPRECATIONS	7-73
[_RECORDING_FEE] ELEMENT	7-74
_RECORDING_FEE DTD DEFINITION	7-74
_RECORDING_FEE EXAMPLES	7-74
_RECORDING_FEE ATTRIBUTES	7-74
_RECORDING_FEE DEPRECATIONS	7-74
[_EXEMPTIONS] ELEMENT	7-75
_EXEMPTIONS DTD DEFINITION	7-75
_EXEMPTIONS EXAMPLES	7-75
_EXEMPTIONS ATTRIBUTES	7-75
_EXEMPTIONS DEPRECATIONS	7-75
[EMBEDDED_FILE] ELEMENT.....	7-76
EMBEDDED_FILE DTD DEFINITION	7-76
EMBEDDED_FILE EXAMPLE.....	7-76
EMBEDDED_FILE ATTRIBUTES	7-77
EMBEDDED_FILE DEPRECATIONS.....	7-78
[DOCUMENT] ELEMENT	7-79
DOCUMENT DTD DEFINITION	7-79
DOCUMENT EXAMPLE	7-79
DOCUMENT ATTRIBUTES	7-79
DOCUMENT DEPRECATIONS.....	7-79

Chapter 1: Introduction

Purpose of this Document

This document is designed to assist individuals who are implementing the PRIA XML standards by providing helpful information and sample XML data. Although it is not intended as an XML tutorial, certain aspects of XML that are important for the proper implementation of the standard are highlighted. The guide will also give a brief background of the PRIA effort, followed by an overview of the data architecture and sample XML data.

What Else You Will Need

This guide covers general aspects of the PRIA standard. To fully implement the standard, PRIA provides other files that can be downloaded from the PRIA web site (www.pria.us).

- **DTD Files** – The Document Type Definition (DTD) files define the structure of the data sets. These files are utilized to develop, write and read XML data files. This iGuide includes four DTDs: Document, Notary, Request and Response
- **LDD Files** – A Logical Data Dictionary (LDD), which defines each data element used in the DTD, is provided which incorporates all four DTDs.
- **Glossary** – A Glossary, which also incorporates all four DTDs, is also provided. The purpose of the Glossary is to provide definitions for many of the terms used.
- **Your Own Data** – The normal purpose of implementing an XML standard is to either convert your own data to an XML format or to convert your data from an XML format.

What is XML?

XML is an acronym for **Extensible Markup Language**. (If you're wondering why it's not called EML, you're not alone.) Data alone does not provide the information a computer needs to properly process and store the data. When we add "markup language" to the data, the purpose of each element of the data becomes clear. It may be obvious to us (but not the computer) that **Jonathan** is a first name and that **Consumer** is a last name. The markup language also tells us that Jonathan is a buyer or grantee, and that the address information provided is his residence.

Data without Markup Language

```
JONATHAN CONSUMER
3750 S BRANDYWINE ST # 242
LAS VEGAS NV 89103
```

Data with Markup Language

```
<GRANTEE _FirstName="JONATHAN" _LastName="CONSUMER">
  <_RESIDENCE _StreetAddress="3750 S BRANDYWINE ST"
    _City="LAS VEGAS" _State="NV" _PostalCode="89103" />
</GRANTEE>
```

What about the Extensible part of XML? To extend or add to the existing data, all we need to do is add the new data along with its markup language label. (Some PRIA XML transactions may contain pre-defined methods for adding new data.)

Data with Markup Language - **Extended**

```
<GRANTEE _FirstName="JONATHAN" _LastName="CONSUMER"
  _NativeLanguage="ENGLISH" >
  <_RESIDENCE _StreetAddress="3750 S BRANDYWINE ST"
    _City="LAS VEGAS" _State="NV" _PostalCode="89103" />
</GRANTEE>
```

Elements and Attributes

In the above example, *GRANTEE* and *RESIDENCE* are **elements** of our sample data. Element names begin with a bracket (<) and end, after any attributes, with another bracket (>).

Elements can also have **attributes** that describe them more completely. Attribute names are followed by an equal sign (=) and the data enclosed in quotes. *GRANTEE* has attributes of First Name, Last Name and Native Language. *RESIDENCE* has attributes of Street Address, City, State and Postal Code.

There is more discussion on how elements and attributes are used in the PRIA Version 2 standards, beginning on page 2-3.

Additional Information

You can locate additional background information regarding XML at web sites such as www.xml.com and www.xml.org.

What is a DTD?

DTD is an acronym for **Document Type Definition**. This is a file that defines the “markup language” that will be used to describe the data. The following sample is the DTD that describes the borrower data on the previous page. There are plenty of books and web sites available that explain DTDs. All we are doing here is showing what a simple DTD looks like.

DTD for the GRANTEE Sample Data

```
<!ELEMENT GRANTEE (_ALIAS*)>
<!ATTLIST GRANTEE
  _StreetAddress CDATA #IMPLIED
  _StreetAddress2 CDATA #IMPLIED
  _City CDATA #IMPLIED
  _State CDATA #IMPLIED
  _PostalCode CDATA #IMPLIED
  _County CDATA #IMPLIED
  _Country CDATA #IMPLIED
  _FirstName CDATA #IMPLIED
  _MiddleName CDATA #IMPLIED
  _LastName CDATA #IMPLIED
  _NameSuffix CDATA #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _CapacityDescription CDATA #IMPLIED
  MaritalStatusType (Married | NotProvided | Divorced | Separated |
Unknown | Unmarried) #IMPLIED
  NonPersonEntityIndicator (Y | N) #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED

<!ELEMENT _ALIAS EMPTY>
<!ATTLIST _ALIAS
  _FirstName CDATA #IMPLIED
  _MiddleName CDATA #IMPLIED
  _LastName CDATA #IMPLIED
  _NameSuffix CDATA #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  AliasType (FormerlyKnownAs | NowKnownAs | AlsoKnownAs) #IMPLIED
  AliasTypeOtherDescription CDATA #IMPLIED
```

What about Schemas?

The DTD is only one of several methods available for describing what markup language will be used with a particular set of XML data. A newer method is called the **Schema** or **XSD** format, which provides more flexibility and the ability to more precisely describe the data. While designed as a DTD, version 2.4 of the PRIA standard includes a “zero-delta” schema for those who prefer to work in this format. PRIA is currently working towards its Version 3 specifications, which will use the Schema/XSD format for defining XML data.

Schema for the Grantee Sample Data

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="GRANTEE">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="_ALIAS"
          minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
      <xsd:attribute name="_FirstName" use="optional" type="xsd:string"/>
      <xsd:attribute name="_LastName" use="optional" type="xsd:string"/>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

```
</xsd:element>

<xsd:element name="GRANTOR">
  <xsd:complexType>
    <xsd:attribute name="_StreetAddress" use="optional"
      type="xsd:string" />
    <xsd:attribute name="_City" use="optional" type="xsd:string" />
    <xsd:attribute name="_State" use="optional" type="xsd:string" />
    <xsd:attribute name="_PostalCode" use="optional" type="xsd:string" />
  </xsd:complexType>
</xsd:element>

</xsd:schema>
```

What is an LDD?

LDD is an acronym for the **Logical Data Dictionary**. The LDD is in a table format that provides definitions for each of the elements used in a PRIA DTD. In **PRIA Version 2**, there is a consistent linkage between the LDD name and the name used in the XML file. An underscore in front of an attribute or element indicates that its LDD entry would also include the name of its containing element.

```
<GRANTEE _FirstName="JONATHAN" _LastName="CONSUMER">
  <_ALIAS _UnparsedName="JONNY B CONSUMER"
    MaritalStatusType=Unknown"/>
</GRANTEE>
```

- **To derive the LDD entry for *_FirstName*...**

GRANTEE + _FirstName = "Grantee First Name"

- **To derive the LDD entry for *_UnparsedName*...**

GRANTEE + _ALIAS + _UnparsedName = "Grantee Alias Unparsed Name"

- **To derive the LDD entry for *MaritalStatusType***

MaritalStatusType = "Marital Status Type"

Sample Entries from Version 2 Logical Data Dictionary

Name	Description	Source	Context	Processes	Datatype
Grantee First Name	The first name of the borrower. Collected on the URLA in Section III (Grantee Name).	URLA	Grantee	PRIA	String
Grantee Alias Unparsed Name	Alternate unparsed name of the GRANTEE	URLA	Grantee Alias	PRIA	String
Marital Status Type	The marital status of the party as disclosed by the party	URLA	Marital Status Type	Credit Reporting, MI, Secondary, AUS, PRIA	Enumerated

What Is PRIA?

The Property Records Industry Association (PRIA) was created in 2003, formed out of the Property Records Joint Task Force, which in turn had been created by the two national associations of county recorders, IACREOT and NACRC. PRIA gathered individuals from a widely varied group of property records industry leaders from both the public and private sectors with extensive business knowledge about the industry. This body of individuals created the **Logical Data Dictionary (LDD)** that defines the meaning of each business data element used within the recording industry. The creation of the LDD has been the key to the success of the PRIA effort. This data dictionary is the seed for generating the XML structures or any other type of structure that may be used in the future. PRIA has closely aligned and coordinated its work products and efforts with MISMO, the Mortgage Industry Standards Maintenance Organization, a subsidiary of the Mortgage Bankers Association.

The result is a single common data set for the recording industry. The seller, buyer, property and other commonly used information have a common data definition, no matter which process is using the data.

The PRIA Development Process

The first and probably most important product developed by the PRIA work group is the logical data dictionary (LDD) that was mentioned earlier. We identified and examined the existing “core data” elements that are used in common by most systems involved in the recording process. In fact, several of the items were used earlier in the eMortgage process and had already been established and defined by MISMO. We then worked to define additional data elements that are needed specifically for recording, notarization, and payment. The data dictionary defines all data elements that become the basis for organizing the XML Document Type Definition (DTD) or data schema that will be used in the future.

Process area work groups (i.e., eRecording, payment, response/receipt, etc.) identify relevant data points and containers. A representative from each work group is responsible for entering the data into a web-enabled tool that warehouses the data dictionary. The work group also then defines the XML DTDs needed to support transactions for their process area. For example, for mortgage services there will normally be a DTD defined to request a service, and a DTD defined for the response from the service provider. Changes made to the data points and containers are monitored to ensure the integrity of all the DTDs.

PRIA agreed to confirm and verify its XML standards with the MISMO XML Architecture Work Group. The representatives of MISMO’s various mortgage process area work groups meet frequently to iron out issues about the data, definitions, and organization of commonly used business data.

Version and Release

This implementation guide is based on **Version 2.4** of the PRIA standard. The PRIA Logical Data Dictionary and DTDs can be downloaded from the www.PRIA.us web site.

Understanding the designation

Major releases of the PRIA standard are represented by the integer designation, while minor updates are represented by the decimal designation. When a release breaks backwards compatibility, it is considered a major release and the integer is incremented. When smaller changes are made that do not break backward compatibility the decimal is incremented. Thus, version 2.4 is not backward compatible with previous versions of the PRIA standard.

Why Version 2.4?

MISMO recently adopted a new review process which among other things, outputs a “zero delta” schema version of all published DTD’s. MISMO has designated the numbering of DTD’s thusly reviewed as Version 2.4. So even though this is the first iteration of PRIA’s version 2, we are designating it Version 2.4 to synchronize with MISMO.

Comments

Comments, questions, and suggestions for improvement of this document may be submitted in writing to the Coordinator who will forward them to the appropriate PRIA Work Group.

PRIA Coordinator (Email: technology@PRIA.us)
PRIA
PO Box 3159
Durham, NC 27715

Chapter 2: PRIA XML Architecture Overview

Transitioning from PRIA Version 1 to PRIA Version 2

Even if you never implemented the PRIA Version 1 XML standards, it may be helpful to have an overview of the basic philosophy behind the Version 1 and Version 2 architectures.

PRIA Version 1 – eRecording 1

The PRIA Version 1 architecture revolved around the concept of a single electronic document for recording.

PRIA Version 1 – Grantee Data (ALL DTDs)

```
<DOCUMENT_RECORDATION>
  ... other data ...
  <GRANTEE>... other Grantee data ...</GRANTEE>
  ... other data ...
</DOCUMENT_RECORDATION>
```

PRIA Version 2 - Transactions

In PRIA Version 2, there has been a move towards more “transactional” structures. Rather than focusing on the building of a single document recording, the PRIA Version 2 DTDs are more oriented towards the commerce of exchanging business data as transactions. For example, there is a DTD for Document, eNotary, Request and Response

Although different DTDs may re-use many of the same structures, they are organized in a manner that is optimal for each transaction. Many DTDs may include the commonly used GRANTEE data element, but depending on the transaction it may be located under different element names as shown in the following examples.

While PRIA Version 1 DTDs all used *DOCUMENT_RECORDATION* as a root element name, each PRIA Version 2 DTD may have different root element names, depending on their purpose. The PRIA Document DTD uses *PRIA_DOCUMENT* as the root element name. The PRIA Request DTD uses the PRIA transaction envelope’s *REQUEST_GROUP* element as its root element name. The PRIA Response DTD uses the *RESPONSE_GROUP* element as its root element name. (See “**Chapter 3: The PRIA Transaction Envelope**” for more information on the PRIA envelope structures).

PRIA 2 – Grantee Data (Recordable Instrument DTD)

```
<PRIA_DOCUMENT>
  ... other data ...
  <GRANTEE>... other grantee data ...</GRANTEE>
</PRIA_DOCUMENT>
```

PRIA 2 – Grantee Data (PRIA Request DTD)

```
<REQUEST_GROUP>
  ... other "envelope" data ...
  <PRIA_REQUEST>
```

```

    ... other data ...
    <PRIA_DOCUMENT>
      ... other data ...
      <GRANTEE>... other grantee data ...</GRANTEE>
    </PRIA_DOCUMENT>
  </PRIA_REQUEST>
</REQUEST_GROUP>

```

PRIA 2 – Grantee Data (PRIA Response DTD)

```

<RESPONSE_GROUP>
  ... other "envelope" data ...
  <PRIA_RESPONSE>
    ... other data ...
    <PRIA_DOCUMENT>
      ... other data ...
      <GRANTEE>... other grantee data ...</GRANTEE>
    </PRIA_DOCUMENT>
  </PRIA_RESPONSE>
</RESPONSE_GROUP>

```

PRIA Version 2 - Data Stored in Attributes

In the **PRIA Version 1** DTD, both XML attributes and elements were used for storing data. Elements require a “start” tag and an “end” tag around the data element and these were used for storing most types of data. The example below shows a simple set of grantee data represented as XML elements.

PRIA Version 1 – Grantee Data (~150 bytes)

```

<GRANTEE>
  <FirstName>MARY</FirstName>
  <MiddleName>ANNE</MiddleName>
  <LastName>SMITH</LastName>
</GRANTEE>

```

In **PRIA Version 2**, most data is stored using the XML attributes. Attributes use a tag name followed by an “equals sign” and then the data enclosed in quotes. Notice that the label identifying the data only appears once, before the data (unlike elements where the label appears before and after the data). This makes the file sizes smaller, which can be important in an environment processing high volumes of transactions.

PRIA Version 2 – Grantee Data (~100 bytes)

```

<GRANTEE   _FirstName="MARY"
           _MiddleName="ANNE"
           _LastName="SMITH" />

```

Elements and Attributes: The DTD Building Blocks

XML provides two types of structures to define data --- elements and attributes. There are no set XML rules for how these two structures are to be used, but the PRIA XML Work Group has defined guidelines for their use, so that a uniform set of XML data structures can be defined for the recording industry.

In PRIA Version 2 DTDs, **Elements** are primarily used to “contain” other elements and attributes. **Attributes** are primarily used to hold the actual data values.

Element Types

There are three types of XML elements commonly used in the PRIA DTDs:

- **Container Elements** – These elements are primarily used to contain other container elements, empty elements and may also have attributes. Container element names will always be in UPPERCASE letters, and will always have a start tag, and an end tag.

```
<GRANTEE _FirstName="MARY" _MiddleName="ANNE" _LastName="SMITH"
  _UnparsedName="MARY ANNE SMITH"
  MaritalStatusType="Married">
  </GRANTEE>
```

- **Empty Elements** – These elements will only contain attributes. Empty element names will always be in UPPERCASE letters, and will always have a start tag, and end with a forward slash and the closing bracket.

```
<GRANTEE _FirstName="MARY" _MiddleName="ANNE" _LastName="SMITH"
  _MaritalStatusType="Married" />
```

- **Repeating Elements** – In XML, elements can be defined to appear more than one time, attributes cannot. While in the PRIA Version 2 DTDs most data is stored in attributes; some DTDs may define repeating elements for holding data such as repeating lines of remarks or comments. Since repeating elements contain actual data, their element names do not use all uppercase letters in their names, like container elements and empty elements.

```
<GRANTEE _LastName="SMITH">
  <_ALIAS _LastName="Doe" />
  <_ALIAS _LastName="Sample" />
</GRANTEE>
```

Attribute Types

In the PRIA Version 2 architecture, attributes are used as the primary method for storing data. PRIA has defined several categories of attributes for this purpose.

- **Enumerated Attributes** – This type of attribute has an enumerated list of allowable values defined in the PRIA DTD or Schema. When an enumerated attribute is used in an XML data file, only a value defined in the enumerated list may be used.

The following section from the PRIA DTD shows an enumerated attribute list for the *_MaritalStatusType* attribute of the *GRANTEE* element. This attribute has six allowed values: “**Married**”, “**Not Provided**”, “**Divorced**”, “**Separated**”, “**Unknown**” and “**Other**”.

```
<!ATTLIST GRANTEE
  _MaritalStatusType ( Married |
                     NotProvided |
                     Divorced |
                     Separated |
                     Unknown |
                     Other )           #IMPLIED>
```

The following is a sample of XML data that shows how the data defined in the DTD sample above would appear. The selected Enumerated Attribute value is enclosed in quotes.

```
<GRANTEE
  _LastName="Smith"
  _MaritalStatusType="Married"
  _NonPersonEntityIndicator="N" />
```

- **String Attributes** – This type of attribute contains a text string, which could be a code, word or phrase. Unlike the enumerated attribute there is no list of valid values provided in the DTD or Schema file.

All of the attributes in the grantee data below – First Name, Last Name, Street Address, City, State and Postal Code - are string attributes.

```
<GRANTEE _FirstName ="JONATHAN" _LastName="CONSUMER
  _StreetAddress="3750 S BRANDYWINE ST" _City="LAS VEGAS"
  _State="NV" _PostalCode="89103" />
```

- **Boolean Attributes** – This is a special type of enumerated attribute that always has the values “Y” or “N” representing a “Yes” or “No” value. The *GRANTEE* section of the PRIA DTD has a Boolean attribute as shown in the partial DTD sample below.

```
<!ATTLIST GRANTEE
    _NonPersonEntityIndicator          (Y|N) #IMPLIED >
```

The following sample shows how the data defined in the DTD sample above would appear in an XML data file.

```
<GRANTEE
    _NonPersonEntityIndicator="N"
/>
```

- **xsd:Boolean Attributes** – PRIA has also adopted a second Boolean data type that corresponds to the **w3c** (World Wide Web Consortium) schema primitive data type of Boolean. It is represented in a PRIA DTD as an enumerated attribute that may have valid values of "Y", or "N".

The following shows how the **xsd:Boolean** data type is declared in the *GRANTOR* element of the PRIA_DOCUMENT DTD.

```
<!ELEMENT GRANTOR EMPTY>
<!ATTLIST GRANTOR
    _UnparsedName CDATA #IMPLIED
    NonPersonEntityIndicator (Y | N)#IMPLIED
>
```

The following samples show two equivalent representations of sample XML data for the *GRANTOR* element.

```
<GRANTOR
    _UnparsedName="John Doe"
    NonPersonEntityIndicator="N" />

<GRANTOR
    _UnparsedName="Doe Company LLC"
    NonPersonEntityIndicator="Y" />
```

- **Date/Time Attributes** – PRIA has adopted the ISO 8601 international standard for representing dates and times. The Date/Time element can hold a date only, or a combined date and time. A full date is formatted in a CCYY-MM-DD format as in the date shown below.

```
RecordedDate="2000-03-10"
```

If there is no “day” value for a date it will be stored in a CCYY-MM format as in the following sample recorded date.

```
_RecordedDate="2000-03"
```

When a time is included in the element, the date and time are separated by the letter “T” as shown in the Request Date Time element shown below. The time portion is set in a HH:MM:SS format. If the “seconds” value is not available the time is set in the HH:MM format. Time values use the 24-hour format (i.e. 11:00 = 11am, 13:00 = 1pm, 14:00 = 2pm, etc.) and should include the Coordinated Universal Time (UTC) offset. Thus 10:13 Central Standard Time on April 30, 2006 would be represented as:

```
RequestDateTime="2006-04-30T10:13:00-06:00"
```

- **Money Attributes** – This PRIA-defined attribute holds money values. Money attributes have the same character limitations as numeric attributes. Fractional dollar amounts are expressed to two decimal places. Whole dollar amounts do not have to include the “.00” decimal value and should not contain dollar signs or commas. The money attribute values are always assumed to be in U.S. dollars.

Valid Values

```
_Amount="225000.00"
```

```
_TotalAmount="240000"
```

```
_PaymentAmount="1934.85"
```

Invalid Values

(Invalid - contains a comma)

```
_Amount="225,000.00"
```

(Invalid – contains a dollar sign)

```
_TotalAmount="$225000"
```

- **Numeric Attributes** – Numeric attributes are used for “non-money” data, like social security numbers, rates, percents, counts or totals, etc. Even though DTDs cannot enforce data types, numeric attributes should only contain the numbers “0” through “9”, plus or minus signs and the decimal point.

```
_NumberOfPages="5"
```

Two specific types of numeric attributes are described here in more detail - **Rate** attributes and **Percent** attributes. A **Rate** is a numeric comparison between two values, a fraction that is expressed as a decimal. A **Percentage** is a number representing a part of a whole that is represented as a quotient multiplied by 100.

Rate attributes represent a ratio that is multiplied directly against a value to produce a result. For example, the Tax Amount on a \$500 item with a Tax **Rate** of .05 is \$25. To express the Tax Rate of .05 as a Tax **Percent** multiply the Rate times 100 (.05 x 100 = 5, or 5%). The following attribute samples both express the same value, the first one as a rate, and the second one as a percent.

```
StateSalesTaxRate=".055"
```

```
StateSalesTaxPercent="5.5"
```

- **ID Attributes** – ID attributes can be used to identify a specific instance of a repeating element. They provide a mechanism for quickly locating specific data elements. Each XML ID attribute value used in an XML data file must be unique throughout the entire file. It must also be a valid XML name – that is, it begins with a letter and is composed of alphanumeric characters and the underscore without white space (spaces, tabs, carriage returns, line feeds).

One method of making sure the ID attributes are unique while maintaining a numeric count is to combine an alpha identifier with a zero padded number. The example below shows two *GRANTEE* container elements - one with a *GranteeID* attribute of “**GraRec0001**” and the other with a *GranteeID* attribute of “**GarRec0002**”. The *GranteeID* could just as easily have values like “**Grantee-1**”, “**JonathanConsumer**”.

```
<GRANTEE ID="GraRec0001"
  _FirstName="JONATHAN" _LastName="CONSUMER"
  _UnparsedName="JONATHAN CONSUMER" />
<GRANTEE ID="GarRec0002"
  _FirstName="JANE" _LastName="CONSUMER"
  _UnparsedName="JANE CONSUMER" />
```

- **IDREF Attributes** – IDREF attributes are used to “refer to” a specific ID attribute in another element. The ID attribute for the *GRANTEE* record in the previous example, has a value of “**GraRec0001**”. In the AUS Loan Application DTD, the *ID* IDREF attribute “refers to” or identifies the other grantee element with whom the grantee jointly conveys the assets listed in the transaction. In this example, the data is saying

that the Jonathan Consumer (whose *ID* = “**GraRec0001**”), has joint assets with Jane Consumer (*ID* = “**GraRec0002**”, which is Jane’s *ID* value).

```
<GRANTEE ID="GraRec0001"  
  IDREF="GraRec0002"  
  _FirstName="JONATHAN" _LastName="CONSUMER"  
  _UnparsedName="JONATHAN CONSUMER" />  
<GRANTEE ID="GraRec0002"  
  IDREF="GraRec0001"  
  _FirstName="JANE" _LastName="CONSUMER"  
  _UnparsedName="JANE CONSUMER" />
```

NOTE: *A more detailed discussion of ID, IDREF and IDREFS attributes begins on the next page.*

- **CamelCase Representation of Data in Attributes** – It is the recommendation of the PRIA eRecording XML Workgroup that data contained in attributes be represented in CamelCase format. While many indexing systems store and display data in UPPERCASE, the documents the data will be presented in normally display in CamelCase. Also, converting from CamelCase to UPPERCASE is more readily accomplished than the reverse.

Containment and Pointing Relationships

Containment

When organizing data, it makes sense to place data like name, and aliases within the confines of a Grantee container, since that data belongs directly to each grantee. Within PRIA, this method of organization is called **containment**. The majority of the data within the PRIA DTDs, especially in Version 2 DTDs, use the containment method to organize the data.

Example Illustrating “Containment” Relationships

```
<PRIA_DOCUMENT>
```

```
<GRANTEE _UnparsedName="JOHN DOE">  
  <_ALIAS _UnparsedName="John Smith"/>  
</GRANTEE>
```

```
<GRANTEE _UnparsedName="JANE DOE">  
  <_ALIAS _UnparsedName="Jane Smith"/>  
</GRANTEE>
```

```
</PRIA_DOCUMENT>
```

Pointing

A potential problem with containment can occur when a grantee shares one or more recordings with another grantee. If we list the jointly owned asset or liability under each borrower, then when we calculate the total assets we could inadvertently count an asset or liability more than once. In this instance, it is better method to place all of the assets into a single list, and place liabilities into a single list. Then you can add a data attribute to identify which borrower(s) the asset or liability belongs to. This method is used within the PRIA Version 2 DTDs for organizing asset and liability data and is called **pointing**. It is called a pointing relationship, because each asset and liability “points to” the borrower or borrowers it is related to.

XML provides a simple mechanism that allows a container element to “point to” another container element. This mechanism is comprised of the ID attributes and IDREFS attributes discussed earlier in this guide. In simplest terms:

- An IDREF attribute “refers to” or “points to” an element with an ID attribute having the same value.
- An IDREFS attribute “points to” one, or more than one element, each having an ID attribute equal to one of the IDREFS attribute values.

To demonstrate how pointing relationships work, the following example shows some sample data provided on a fictitious transaction. The payments are made for each document recordation. In this case we have 3 recording documents and a separate payment for each. The internal code contained in `_UniqueIdentifier` is used as an identifier in the example, but any number or combinations could be used just as easily.

Example Illustrating “Pointing” Relationships

```
<PACKAGE>
```

```
<PRIA_DOCUMENT _UniqueIdentifer="111111111" />
```

```
<PRIA_DOCUMENT _UniqueIdentifer="222222222" />
```

```
<PRIA_DOCUMENT _UniqueIdentifer="333333333" />
```

```
<PAYMENT ReferenceIdentifer="111111111"  
  _AccountIdentifer="063206-2438661"  
  _Amount="15" />
```

```
<PAYMENT ReferenceIdentifer="222222222"  
  _AccountIdentifer="1361863111r31"  
  _Amount="250" />
```

```
<PAYMENT ReferenceIdentifer="333333333"  
  _AccountIdentifer="y83621s6a2"  
  _Amount="2950" />
```

```
</PACKAGE>
```

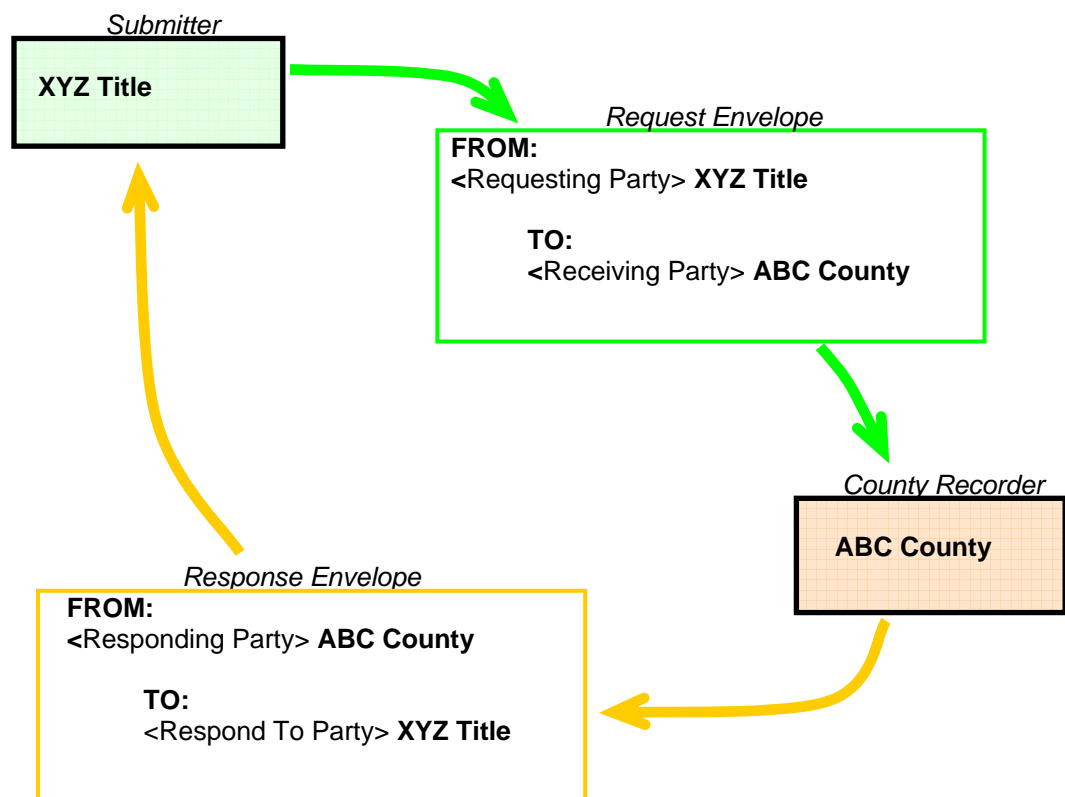
Chapter 3: The PRIA Transaction Envelope

Since the PRIA Version 2 standards are oriented towards transactions, PRIA has developed a set of Transaction Envelope DTDs that can be used to wrap the transaction data. The DTDs contain basic information common to most transactions – elements that identify the requesting party, receiving party, responding party and other reference data that is commonly exchanged between business partners.

The use of the PRIA Transaction Envelope is flexible. Some business partners may prefer to use methodologies such as **SOAP** (Simple Object Access Protocol) to wrap their PRIA transactions.

This chapter of the implementation guide covers both the Transaction Request Envelope used in requesting services or products, the Transaction Response envelope used by vendors delivering services or products, and the PRIA recommended SOAP implementation. The drawing below shows a simple request transaction between a title company and a county recorder, followed by the response transaction from the county recorder to the title company.

Request and Response Transactions



The Request Envelope

The PRIA Architecture provides a variety of request elements that allow for a flexible request structure that can be adapted to a variety of business-to-business scenarios.

Sample REQUEST with Requesting and Receiving Parties

```
<REQUEST_GROUP PRIAVersionID="2.4">
  <REQUESTING_PARTY
    _Name="XYZ Title"
    _StreetAddress="21650 Oxnard Street"
    _City="Woodland Hills" _State="CA" _PostalCode="91364"/>
  <RECEIVING_PARTY
    _Name="ABC County"
    _StreetAddress="7200 Peachtree Street"
    _City="Atlanta" _State="GA" _PostalCode="30010"/>
  <REQUEST RequestDatetime="2002-01-08T17:19:01"
    InternalAccountIdentifier="ABC-0732">
    <KEY _Name="XYZ Transaction ID" _Value="702430023"/>
    <KEY _Name="XYZ Portfolio ID" _Value="XYZ2002-0030"/>
    <REQUEST_DATA>
      ... BODY OF REQUEST GOES HERE ...
    </REQUEST_DATA>
  </REQUEST>
</REQUEST_GROUP>
```

Requesting Party Element

The Requesting Party is the ultimate customer of the service needed for the mortgage transaction. It is usually the entity that is ultimately billed for the service. **XYZ Title** is the requesting party used in the request samples in this section of the guide.

Preferred Response Element

Normally, the response format for B2B (Business To Business) transactions is defined between the trading partners beforehand and is the same for all transactions. However, the **Preferred Response** element of the Requesting Party can be used to override the pre-defined response format settings on a transaction-by-transaction basis. This element's attributes allow the Requesting Party to specify the format and destination of the response transactions. They can also specify whether a response format is to be separate from the XML data or embedded within it.

The **Preferred Response Format** attribute has the options: PCL, PDF, Text, or Other. The **Preferred Response Method** can be Fax, File, FTP, HTTP, HTTPS, Message Queue, SMTP, VAN, etc. A **Preferred Response Destination** attribute specifies a value appropriate for the method chosen. For example, its value could be a Fax phone number, file name, email address, a URL, etc.

The **Version 2.4** has added a **Version Identifier** attribute to the **Preferred Response** element to allow the requester to specify the version of the response file they would like to receive. For example, attributes set as `_Format="XML"` and `_VersionIdentifier="PRIA 2.4"` would indicate that the requester expects to receive a PRIA Version 2.4 response file.

NOTE: *Check with the service provider to verify whether the Preferred Response element is supported and which Formats, Methods, and MIME Types are recognized.*

Sample Requesting Party with Preferred Response

```
<REQUEST_GROUP PRIAVersionID="2.4">
  <REQUESTING_PARTY
    _Name="XYZ Title"
    _StreetAddress="21650 Oxnard Street"
    _City="Woodland Hills" _State="CA" _PostalCode="91364">
    <PREFERRED_RESPONSE _Format="PDF"
      _Method="File"
      _Destination="70240023.pdf"
      _UseEmbeddedFileIndicator="Y"
      MIMEType="application/pdf" />
    <PREFERRED_RESPONSE _Format="Text"
      _Method="SMTP"
      _Destination="DSmith@XYZ.com"
      _UseEmbeddedFileIndicator="N"
      MIMEType="multipart/encrypted" />
  </REQUESTING_PARTY>
  <RECEIVING_PARTY
    _Name="ABC County"
    _StreetAddress="7200 Peachtree Street"
    _City="Atlanta" _State="GA" _PostalCode="30010" />
  <REQUEST RequestDatetime="2002-09-08T15:11:47"
    InternalAccountIdentifier="ABC-0732">
    <KEY _Name="XYZ Transaction ID" _Value="70240023" />
    <KEY _Name="XYZ Portfolio ID" _Value="XYZ2002-0030" />
    <REQUEST_DATA>
      ... BODY OF REQUEST GOES HERE ...
    </REQUEST_DATA>
  </REQUEST>
</REQUEST_GROUP>
```

Receiving Party Element

The Receiving Party simply identifies the party that is on the receiving end of the request transaction. Normally it is a provider of a service or product. **ABC County** is the service provider in these samples.

Submitting Party Element

The Submitting Party is an individual or organization that makes a request for a service on behalf of the Requesting Party. For example, when a service broker makes a request for a service on behalf of the end user of the service, they would identify themselves in the request transaction as the Submitting Party, and identify the end user as the Requesting Party.

In the sample file below, the service broker – **AAA Aggregator** (Submitting Party) is generating a request for a mortgage service on behalf of **XYZ Title** (Requesting Party). AAA Aggregator sends the request to **ABC County** (Receiving Party), who processes the request. ABC County will receive payment from XYZ Title for the service.

Sample REQUEST with Requesting, Receiving and Submitting Parties

```
<REQUEST_GROUP PRIAVersionID="2.4">
  <REQUESTING_PARTY
    _Name="XYZ Title"
    _StreetAddress="21650 Oxnard Street"
    _City="Woodland Hills" _State="CA" _PostalCode="91364"/>
  <RECEIVING_PARTY
    _Name="ABC Services"
    _StreetAddress="7200 Peachtree Street"
    _City="Atlanta" _State="GA" _PostalCode="30010"/>
  <SUBMITTING_PARTY
    _Name="AAA Aggregator"
    _StreetAddress="324 Redstone Ave"
    _City="Denver" _State="CO" _PostalCode="81351"/>
  <REQUEST RequestDatetime="2002-01-08T17:19:01"
    InternalAccountIdentifier="ABC-0732">
    <KEY _Name="AAA Transaction #" _Value="23029497729"/>
    <KEY _Name="XYZ Transaction ID" _Value="702430023"/>
    <KEY _Name="XYZ Portfolio ID" _Value="XYZ2002-0030"/>
    <REQUEST_DATA>
      ... BODY OF REQUEST GOES HERE ...
    </REQUEST_DATA>
  </REQUEST>
</REQUEST_GROUP>
```

The diagram on the next page shows an original request transaction from a lender to an Internet portal (Service Aggregator). In the first request transaction, the lender is the **Requesting Party** and the Internet portal is the **Receiving Party**.

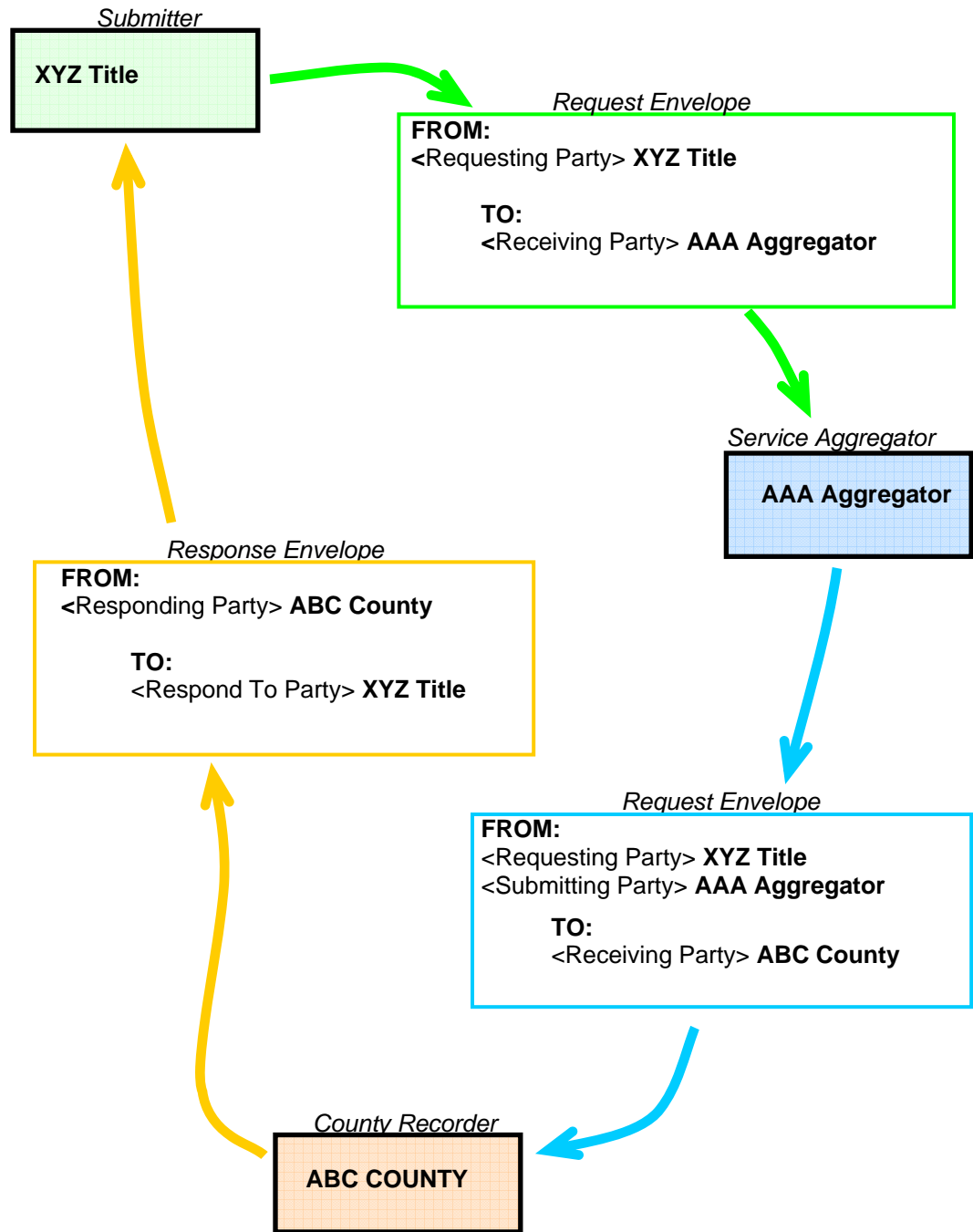
In the second request transaction from the service broker to the service provider, the lender is still the **Requesting Party**, but the service broker is the **Submitting Party** and the service provider is the **Receiving Party** (as shown in the sample above).

In the response transaction, the service provider is the **Responding Party** and the lender is the **Respond To Party**.

For **Version 2.4**, several enhancements were made to the **Submitting Party** data structure. First, this element can now appear more than once within a **Request Group**. In addition a **Sequence Identifier** attribute has been added. These two changes now allow more than one

submitting party to be listed in a request transaction, which may be needed in situations where a request is being handled and passed through by more than one web portal. A **Login Account Identifier** attribute and a **Login Account Password** attribute have also been added for situations where authentication information needs to be passed through to another party in the transaction.

Request and Response Transactions with Submitting Party



Request Element

Each PRIA 2.x Request Envelope can contain one or more **Request Elements**. The Request Element can contain one or more **Request Data Elements** that holds the actual details of the particular type of request (i.e. Recording, Fee Quote, Status Check, etc.).

The Request Element contains several optional attributes:

- **RequestDatetime** – when the Requesting Party made the request.
- **InternalAccountIdentifier** – the account number assigned to the Requesting Party by the Receiving Party for billing purposes.
- **LoginAccountIdentifier** – required by some Receiving Party vendors to gain access to their system.
- **LoginAccountPassword** – may also be required along with the Login ID above.

One additional feature of the Request Element is the **Key Element**. This element allows the passing of one or more reference or identification “keys” in the request transaction to the service provider. **The service provider will then return the same “keys” in the response transaction.**

The use of the KEY element in this manner allows the requester to send reference data in the request, which will make it easy for the requester to match the response transaction to the original request. Examples of data that could be included in the KEY element could include Loan IDs, Request IDs, Mailbox Numbers, or any other data that would allow the requester to match a response to a request. Each KEY element has a **Name** and **Value** attribute. The Key Name describes the Key Value.

Sample REQUEST with Reference KEY Elements

```
<REQUEST_GROUP PRIAVersionID="2.4">
  <REQUESTING_PARTY
    _Name="XYZ Title"
    _StreetAddress="21650 Oxnard Street"
    _City="Woodland Hills" _State="CA" _PostalCode="91364"/>
  <RECEIVING_PARTY
    _Name="ABC County"
    _StreetAddress="7200 Peachtree Street"
    _City="Atlanta" _State="GA" _PostalCode="30010"/>
  <REQUEST RequestDatetime="2002-01-08T17:19:01"
    InternalAccountIdentifier="ABC-0732">
    <KEY _Name="XYZ Transaction ID" _Value="702430023"/>
    <KEY _Name="XYZ Portfolio ID" _Value="XYZ2002-0030"/>
    <REQUEST_DATA>
      ... BODY OF REQUEST GOES HERE ...
    </REQUEST_DATA>
  </REQUEST>
</REQUEST_GROUP>
```

The Response Envelope

This section describes the various elements used in the RESPONSE element of a response transaction. In the response, the party roles names are changed. The request's Requesting Party becomes the Respond To Party. The request's Receiving Party becomes the Responding Party.

Responding Party Element

The Responding Party identifies the entity that prepared the response transaction – usually the county recorder. In the request transaction, they were the Receiving Party.

Respond To Party Element

In a response transaction, the Respond To Party is the entity receiving the response transaction. In the request transaction, they were the Requesting Party.

Sample RESPONSE with Responding and Respond To Parties

```
<RESPONSE_GROUP PRIAVersionID="2.4">
  <RESPONDING_PARTY
    _Name="ABC County"
    _StreetAddress="7200 Peachtree Street"
    _City="Atlanta" _State="GA" _PostalCode="30010" />
  <RESPOND_TO_PARTY
    _Name="XYZ Title"
    _StreetAddress="21650 Oxnard Street"
    _City="Woodland Hills" _State="CA" _PostalCode="91364" />
  <RESPONSE RequestDateTime="2002-01-08T17:19:12"
    InternalAccountIdentifier="ABC-0732">
    <KEY _Name="XYZ Transaction ID" _Value="702430023" />
    <KEY _Name="XYZ Portfolio ID" _Value="XYZ2002-0030" />
    <RESPONSE_DATA>
      ... BODY OF RESPONSE GOES HERE ...
    </RESPONSE_DATA>
  </RESPONSE>
</RESPONSE_GROUP>
```

Response Element

Each PRIA 2.4 Response Envelope can contain one or more **Response** elements. The Response Element can contain one or more **Response Data** elements that hold the actual details of the particular type of response (i.e. Recording, Fee Quote, Status Check, etc.). The Response Element also can contain a **Status** element, which can be used for reporting error or status messages about the service or product requested.

The Response Element contains several optional attributes:

- **ResponseDateTime** – when the Responding Party created the response transaction.
- **InternalAccountIdentifier** – the account number from the request transaction that was assigned to the Requesting Party by the Receiving Party for billing purposes.
- **LoginAccountIdentifier** – required by some Receiving Party vendors to gain access to their system.
- **LoginAccountPassword** – may also be required along with the Login ID above.

The service provider in the Response Transaction will return the same Key Elements that were included in the Request Transaction, in their entirety. The sample RESPONSE element below contains the same KEY elements that were initially sent by the customer in the request file.

Sample RESPONSE with Response Element and Key Elements

```
<RESPONSE_GROUP PRIAVersionID="2.4">

  <RESPONDING_PARTY
    _Name="ABC County"
    _StreetAddress="7200 Peachtree Street"
    _City="Atlanta" _State="GA" _PostalCode="30010"/>

  <RESPOND_TO_PARTY
    _Name="XYZ Title"
    _StreetAddress="21650 Oxnard Street"
    _City="Woodland Hills" _State="CA" _PostalCode="91364"/>

  <RESPONSE ResponseDateTime="2002-01-08T17:19:12"
    InternalAccountIdentifier="ABC-0732">
    <KEY _Name="XYZ Transaction ID" _Value="702430023"/>
    <KEY _Name="XYZ Portfolio ID" _Value="XYZ2002-0030"/>
    <RESPONSE_DATA>
      ... BODY OF RESPONSE GOES HERE ...
    </RESPONSE_DATA>
  </RESPONSE>

</RESPONSE_GROUP>
```

Chapter 4: XML Implementation Issues

XML Reserved Characters

The topic of XML “reserved” characters is normally covered in XML tutorials, but is worth emphasizing here because ignoring these reserved characters is a frequent cause of errors in recording data. There are five characters that are reserved and cannot be used directly in XML element or attribute data, they must be replaced with what are called – XML Entity References. Some XML software systems will automatically do the conversion of the XML reserved characters; otherwise logic must be added to perform the conversion of the characters. The following table shows the reserved character in the first column, followed by the XML Entity Reference that it is replaced with in the second column.

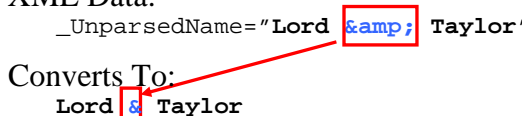
Reserved Character	Substitute With	Character Name
&	&	Ampersand
<	<	Less Than
>	>	Greater Than
'	'	Apostrophe
“	"	Quote

Processing Received XML Data

The following examples show the XML data received in a recording transaction being converted into “normal” data, as it would be stored in an internal database.

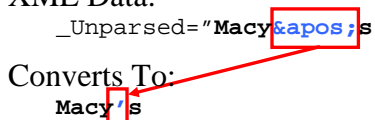
Example 1:

XML Data:
_UnparsedName="Lord & Taylor"
Converts To:
Lord & Taylor



Example 2:

XML Data:
_Unparsed="Macy 's"
Converts To:
Macy 's



NOTE: Some XML parsers may not require that the apostrophe be converted to the `'` entity, when it appears inside an attribute that is enclosed in quotes as shown in Example 2 above.

Generating XML Data

The following examples show the conversion of “normal” data into XML data using the table from the previous page.

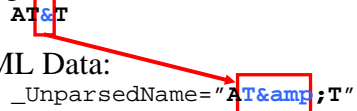
Example 1:

Original Data:

AT&T

XML Data:

`_UnparsedName="AT&T"`



Example 2:

Original Data:

William "Billy" Goat

XML Data:

`_UnparsedName="William "Billy"; Goat"`



Specifying the DTD in the XML Data File

Near the beginning of XML data files, there is a DOCTYPE section that specifies the location of the DTD that is used to validate the data in the XML file. Although the DTD itself can be included within the DOCTYPE section, it is usually maintained as a separate file. The DOCTYPE declaration specifies the name and/or location of the DTD file. The following line is a sample DOCTYPE declaration line from an XML data file.

```
<!DOCTYPE PRIA_DOCUMENT SYSTEM "PRIA2_4.DTD">
```

The first argument of the DOCTYPE declaration specifies the name of the “root” or main element of the data file – in this case, PRIA_DOCUMENT. The remainder of the line specifies the type of DTD and its name and/or location. In this case, the DTD name is shown as “PRIA_DOCUMENT2_4.DTD”. Since a directory path or URL (Uniform Resource Locator) is not specified for the DTD, the software processing the XML data file will look for the DTD in the same directory as the XML data file.

The next two DOCTYPE samples specify a DTD in a specific directory path – the first on a local hard drive and the following one located at the PRIA web site.

```
<!DOCTYPE PRIA_DOCUMENT SYSTEM "C:/DTD/ PRIA2_4.DTD">
```

```
<!DOCTYPE PRIA_DOCUMENT SYSTEM  
  "http://www.pria.us/pria/dtd/PRIA2\_4.DTD">  
<!DOCTYPE RESPONSE_GROUP PUBLIC "-//PRIA//DTD PRIA//EN//2.4"  
  "http://www.PRIA.us/pria/dtd/PRIAResponse_v2_4.DTD">
```

The important point of this section is that when sending XML data files to your trading partners, they may have differing requirements for where the DTD is to be found by their processing software. Some may not require a specific location for the DTD and will use the default DOCTYPE that specifies the DTD name without a path or URL. Other trading partners may ask you to specify a specific directory path or URL for locating the DTD.

White Space in XML Documents

All of the XML sample data shown in this document includes formatting characters to make the data more readable. Each element or attribute is on its own line, and data within element containers is indented. These space, tab, carriage return and line feed characters are known as “white space” and create a nicely formatted output as shown below.

Sample XML Data with “White Space”:

```
<GRANTEE  _FirstName="JERRY"  
          _MiddleName="VICTOR"  
          _LastName="SMITH"  
          _UnparsedName="JERRY VICTOR SMITH"  
          MaritalStatusType="NotProvided">  
  <_ALIAS _FirstName="John" _LastName="Doe" />  
</GRANTEE>
```

The XML files that are exchanged between trading partners will normally be sent without any “white space” characters as shown below.

Sample XML Data without “White Space”:

```
<GRANTEE _FirstName="JERRY" _MiddleName="VICTOR" _LastName="SMITH"  
_UnparsedName="JERRY VICTOR SMITH"  
MaritalStatusType="NotProvided"><_ALIAS _FirstName="John"  
_LastName="Doe" /></GRANTEE>
```

“White space” characters are generally eliminated from production XML files for two reasons:

1. Eliminating the “white space” characters makes the XML files smaller (approximately 7% to 9%).
2. Most Internet browser software and XML editors automatically format the XML data with “white space” to make it more readable.

Handling Attributes with No Data

When processing XML data from your business partners, you should be aware that there are two valid ways of showing (or not showing) that an attribute has no data value.

The preferred method is just to not include the attribute name in the XML file. The following data sample shows a container record for a grantee with no middle name – and no `_MiddleName` attribute.

Sample Borrower - No Middle Name Attribute:

```
<GRANTEE_FirstName="NATALIE"  
_LastName="FERGUSON" />
```

Some XML generation software will create an “empty” attribute, where the attribute name, equal sign and quotes are shown, but there is not data within the quotes.

Sample Grantee – “Empty” MiddleName Attribute:

```
<GRANTEE _FirstName="NATALIE"  
_MiddleName=""  
_LastName="FERGUSON" />
```

NOTE: *Enumerated attributes, ID attributes, and Boolean attributes cannot be displayed as “empty”. This would cause the XML files to fail validation.*

Ordering Elements and Attributes

Elements

Data elements contained within any of the major components must appear in the order specified by the DTD or schema. Below is a DTD sample for the `REQUEST_GROUP` container element. Its elements (`REQUESTING_PARTY`, `RECEIVING_PARTY`, `SUBMITTING_PARTY` AND `REQUEST`) must appear in the same order in the XML data file, as they appear in the DTD.

```
<!ELEMENT REQUEST_GROUP  
(REQUESTING_PARTY* ,  
RECEIVING_PARTY? ,  
SUBMITTING_PARTY? ,  
REQUEST*)>
```

Attributes

When an element contains multiple attributes, the attributes may appear in any order in the XML data file.

```
<!ELEMENT KEY EMPTY>  
<!ATTLIST KEY  
_Name CDATA #IMPLIED  
_Value CDATA #IMPLIED >
```

Chapter 5: HTTP Post Name/Value Pair Recommendation

NOTE: PRIA has entered into an agreement with MISMO as of October 15, 2005 and has adopted the MISMO Request & Response.

This recommendation was approved by the MISMO Architecture Work Group on December 12, 2002.

Many service providers are using the HTTP/HTTPS Post operation for web based transactions, but each one currently has different sets of name/value pairs used in the post. This recommendation provides a standard set of name value pairs for use in PRIA transactions.

The main purpose of the HTTP/HTTPS Post is to allow a document submitter to login to a recorder's system so that the request payload can be sent and processed. The recorder needs enough information to identify and validate the requester so that the request payload can be routed and processed properly. In this recommendation, the HTTP Post Names match existing elements already defined in the PRIA & MISMO Logical Data Dictionaries (LDD).

Login Account Identifier – The login ID of an individual user.

Login Account Password – The login password of an individual user.

Internal Account Identifier – This is the payment account ID assigned by the recorder to the document submitter. Some recorders may derive this data based on the *Login Account Identifier*.

Request Data – This is the final HTTP Post Name that contains the actual request “payload”.

Sample HTTP Post Transaction with Recommended Name/Value Pairs:

```
LoginAccountIdentifier=JDoe&LoginAccountPassword=e255xb&InternalAccountIdentifier=ABC+Title&RequestData={...Request transaction data goes here...}
```

Chapter 6: Security Principles

The intent of this Chapter of the PRIA eRecording Implementation Guide – General Information – Version x is to provide some education and some general guidance with respect to security principles.

Armed with knowledge, trading partners can then select their own security and risk mitigation tools and solutions to meet both their specific business needs and the needs of their trading partners.

General Security Principles

As the name implies, there are some generally accepted principles that are commonly used when addressing security concerns regarding the movement of data between various parties to a business transaction.

Authentication

Authentication is the process of establishing confidence in user identities.¹

Trading partners must perform authentication to establish a degree of confidence in the identity with who they are in communication. This can be driven by legislative, regulatory, intellectual propriety, financial and general privacy protection requirements. Regardless, trading partners must deploy a reliable mechanism to assert their identity as well as validate their partner's identity

Within this chapter, the representation of identity is referred to as a “token”. Tokens can take many forms: login string, passwords, account numbers or digital certificates. Some forms for tokens are more secure than others. Account numbers or passwords are usually clear-text strings and contain no embedded protection to provide privacy (see additional comments through out this Guide related to clear text logins and passwords); hence, confidentiality must be applied when they are used. These recommendations attempt to address protection requirements for non-secure tokens

Confidentiality

Confidentiality is reserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information. [44 U.S.C., SEC. 3542]²

Confidentiality and privacy are often used synonymously. There are several methods to achieve privacy for information from strong access controls to encryption. Robust authentication can be used to implement identity based

¹ NIST SP 800-63

² NIST SP 800-199

access controls. However, for PRIA transactions (data-in-motion) between trading partners, encryption is the preferred solution. As a general rule, private or sensitive data-in-motion should be protected, regardless of internal (enterprise) or external transports. Restated, any sensitive data that is transferred over public networks (e.g., internal eMail/IM) or stored in portable storage media (e.g., laptops, flash/USB drives) should be encrypted to protect it from unauthorized access or disclosure.

Integrity

Integrity involves guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity. [44 U.S.C., SEC. 3542]³

Integrity comprises timely, accurate, complete, and consistent data. The information must not be manipulated in any way, either through electronic errors or human intention. The use of hashing functions and digital signatures are very common in many system applications to provide data integrity services. A strong hashing function ensures that data modification does not go undetected. And by then digitally signing the hash value, one can ensure that the hash can be trusted.

Non-repudiation

Non-repudiation can provide various levels of assurance that the sender of information is provided with proof of delivery and the recipient is provided with proof of the sender's identity, so neither can later deny having processed the information.⁴

Historically, repudiate is a legal term for the ability to deny or reject validity or authority. In the world of eCommerce, the goal of non-repudiation is to prevent repudiation of valid transactions, which is critical to the success of eCommerce. The ability to prevent an entity from denying a particular act must be supported to ensure intent. There are several examples from a simple transaction between parties to an electronic signature (eSignature).

Appropriate policies and procedures, along with the security principles of authentication and integrity are combined into a single principle. This helps to ensure the identity of the entity and integrity of the associated transaction, which provides evidence against modification. A commonly used method for non-repudiation is XML digital signature (DigSig).

³ NIST SP 800-199

⁴ NIST SP 800-53, Revision 1

Privileged-Based Access Control

Privileged-based access control is the enforcement of specified authorization rules based on positive identification of users and the systems or data they are permitted to access.⁵

Of the five basic security principles, access control is outside the scope of this document. Access controls or authorization are based on privileges granted between trading partners and is not within the current scope of this standards body.

The accuracy, confidentiality and integrity of data are critical to the mortgage process. Without appropriate security solutions, issues such as fraud and privacy rights violations can severely hamper an organization's ability to migrate to the use of electronic data exchanges and other core eCommerce processes.

The application of these principles to PRIA's mission is the foundation for the recommendations included in this chapter. Authentication, confidentiality, and integrity are fundamentals applied to secure transmission of information. Each of these three principles is practiced to resolve specific requirements related to access, disclosure, or modification of sensitive information. Non-repudiation is often achieved by implementing a digital signature to a document or data element.

⁵ www.utmb.edu/is/security/glossary.htm

General Recommendations for Securely Transporting Sensitive Recording Information

As with all security programs, there are trade-offs to be made between achieving regulatory compliance, and assessing legal ramifications, privacy concerns, resources, and expenses. Businesses are required by various laws to maintain their security programs and risk mitigation controls. Authentication, confidentiality and integrity solutions can be found in many different tools and solutions.

In general, all communications between trading partners should be performed through a secure transport or tunnel. A secure tunnel can be described as an encrypted connection between two end points. For example, when a consumer uses the Internet for eCommerce (e.g., on-line banking), a secure tunnel is established between the user's browser and the remote web server. Over time, consumers have come to recognize that the "lock" icon appears as notification to the user that a secure connection has been established. All data transmitted between the two end points (browser and server) is now encrypted. Business-to-business (B2B) transactions should also be performed through a secure tunnel using protocols such as *SSL/TLS*, *SFTP/SSH* or *IPsec*. If you are not sure whether your current electronic business operations utilize secure tunnels, then it is recommended that your organization research and verify this environment. If your organization has not implemented the use of secure tunnels, then it is highly recommended that your organization implement them to ensure appropriate protection of sensitive information being transported between business entities.

Other areas of concern when transporting sensitive information are electronic mail (eMail) and Instant Messaging (IM). Both technologies offer convenience and productivity to users. In addition, both technologies offer users the ability to transmit sensitive data outside the controls of an application environment. For example, many organizations deploy a server outside the internal network in a *DMZ* (demilitarized zone) with a secure tunnel to the remote client/system and some form of authentication. These automated processes are intended to ensure access controls, confidentiality, and protection to the internal network. Tools such as eMail and IM circumvent that infrastructure. There is no guaranty of privacy for the text or attachment(s), and authentication is based on the sender's eMail or IM address. There are however, solutions that can be implemented such as *S/MIME* (Secure/Multipurpose Internet Mail Extensions), *PGP*, or other commercially available data encryption technologies to provide for a more secure message exchange. Businesses should establish policies around the use of eMail and IM, and if required deploy secure messaging technology and policies that incorporate the general security principles discussed.

Secure Messaging Requirements for Supporting Electronic Recording Processes

The MISMO Information Security Work Group (ISWG) has defined a set of requirements for securely transporting sensitive electronic mortgage information. The requirements are independent of the technology solutions and we include them in totality below. Current PRIA standards may not be able to provide support for all of the requirements, but the security requirements are expected to be made mandatory in future PRIA messaging standards.

MISMO ISWG Recommendations for Secure Messaging Requirements:

- **Multiple security tokens for authentication** – Messaging standards must have support for multiple authentication tokens. Authentication must support User ID, account number, password, and digital certificate.
- **Multiple encryption and digital signature technologies** – Messaging standards must have support for multiple encryption and digital signature algorithms that secure all or a part of the content being transported. Support should include commercially recommended hash/digest, symmetric, and asymmetric algorithms.
- **Integrity** – Messaging standards must have embedded integrity support for all or a part of the content being transported.
- **Multiple Trusted Service Providers** – Messaging standards must support separate security domains (e.g., Service Providers) providing independent authentication, confidentiality and integrity for each domain thus allowing data to flow between multiple parties within a single business transaction.
- **End-to-end message-level security** – Messaging standards must provide support that ensures all or a part of the content being transported is adequately secured (i.e., encrypted and/or digitally signed) between two parties, including if the content is being transported through an intermediary entity(ies) (e.g., portal server).

Recommended Security Solutions

Recommended security solutions are demonstrated as use case scenarios.

There are three basic use case scenarios:

1. Trading Partner to Trading Partner Direct
2. Trading Partners through a Portal
3. Multi-services (single service requesting entity to multiple providers of services)

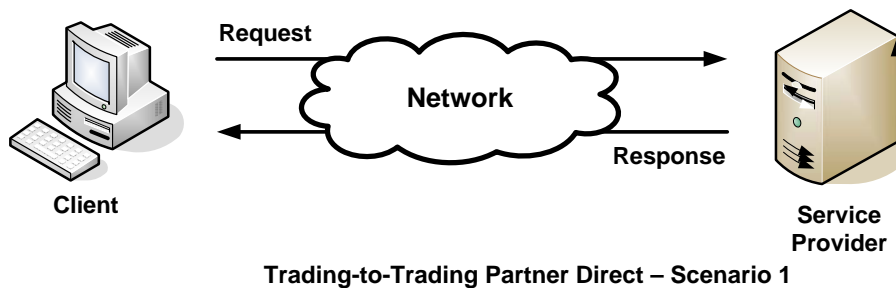
Each use case scenario will contain a description, business use case, security requirements, and possible security recommendations. The possible security recommendations will address each security requirement with one or more potential solutions. The list is not an inclusive list of all potential solutions; rather it is a list that is most applicable to the electronic mortgage environment.

Assumptions:

- All transactions are performed using a secure tunnel.
- Trading partners have pre-negotiated legal terms and conditions, and authentication tokens or privilege rights have been exchanged.
- Examples selected will embed the MISMO Envelope as a header for the process area transaction. For example, both the MISMO Credit Request and Response *DTDs* utilize the MISMO Envelope Request/Response Group *DTDs* as the header for their business process area.
- The simple diagrams are not intended to represent a complex network environment. The diagram associated with the “Client” appears to represent a Desktop PC. In reality it could be a server or some other device. Both “Client” and “Service Provider” environments could contain several domains with the application service in one domain and a communication server in a *DMZ*.

Trading Partner-to-Trading Partner Direct Scenario

This is the simplest of all use cases. There are only two parties involved in the transaction: a requesting party (client) and the responding party or Service Provider (SP). The transaction is a single document or PRIA DTD. For example, lender ABC Mortgage issues a PRIA Request DTD to XYZ County Recorder. XYZ County Recorder will respond with the corresponding PRIA Response DTD.



Security Requirements

1. Client authenticates Service Provider (SP). SP presents token to Client, and Client validates token. Client authentication of the Service Provider (server) will reduce *pharming* scams and is useful for synchronous transactions.
2. SP authenticates Client. Client presents token to SP, and SP validates token.
3. Protect any sensitive data (e.g., PI) transmitted between Client and SP. Some PRIA transactions (DTDs) contain personal information (PI). These sensitive elements should be identified to ensure appropriate protection is being applied. At a minimum, a secure tunnel is to be used to pass both the request and response transactions.
4. Ensure integrity of data being transmitted between Client and SP. A secure and mutually authenticated tunnel will provide some implicit level of integrity.

Possible Security Solutions

Authentication Solutions

- SSL/TLS – Client and SP mutual authentication using digital certificates (i.e., SSL server certificate for SP and client digital certificate for client).
- SSL/TLS – Client and SP mutual authentication using a combination of digital certificates and username/passwords. Client authenticates SP by validating SP's SSL server certificate; SP authenticates client by validating client's username/password (obtained from HTTP header).

- *Secure FTP* – Client and SP mutual authentication using a password or certificate based secured FTP connection.
- *VPN* over frame relay/dedicated lines using *PKI digital certificates* or Share Secrets for mutual authentication.
- Digitally signed email using S/MIME or PGP to provide mutual authentication of sender and recipient.

Confidentiality Solutions

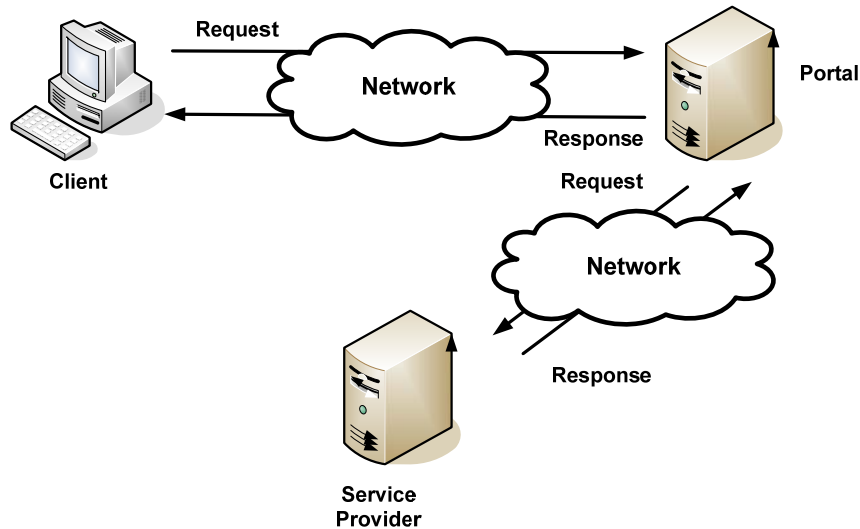
- SSL/TLS using minimum 1024 bit *RSA public keys* and 128 bit *3DES* or *AES*. SSL/TLS provides for an encrypted tunnel to transport plaintext data.
- VPN using SSL/TLS or IPsec. VPN provides for an encrypted tunnel to transport plaintext data.
- Encrypted email using S/MIME or PGP and minimum 1024 bit RSA public keys and 128 bit 3DES or AES.
- *XML encryption* to provide additional confidentiality on specific data elements within an XML document being transported through a secure tunnel.

Integrity Solutions

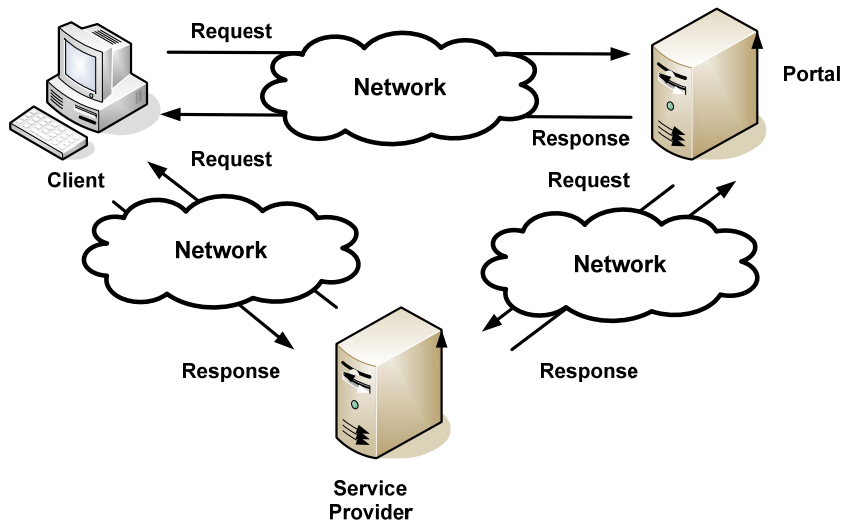
- SSL/TLS – Implicit integrity is achieved through establishment of a mutually authenticated SSL/TLS connection.
- VPN using SSL or IPsec – Implicit integrity is achieved through establishment of a mutually authenticated VPN connection
- S/MIME email – A digitally signed email also provides data integrity services.
- XML signature – Additional data integrity can be provided on specific XML data elements within an XML document being transported through a secure tunnel.

Trading Partners through a Portal Scenario

In this scenario, the client interacts with a SP via an intermediary entity such as a Portal. There are two variations on this scenario as described below. In the first variation, the Portal acts as a pass through for the requests and responses that are exchanged between the client and the SP. In the second variation, the Portal redirects the client's request to the appropriate SP.



Trading Partners through Portal - Scenario 1



Trading Partners through Portal - Scenario 1

Security Requirements (Variation 1)

1. Client authenticates Portal. Portal presents token to Client, and Client validates token.
2. Portal authenticates Client. Client presents token to Portal, and Portal validates token.
3. Portal authenticates SP. SP presents token to Portal, and Portal validates token.
4. SP authenticates Portal. Portal presents token to SP, and SP validates token.
5. Protect any sensitive data (e.g., PI) transmitted between Client, Portal and SP. Some PRIA transactions (DTDs) contain personal information (PI). These sensitive elements should be identified to ensure appropriate protection is being applied. At a minimum, a secure tunnel is to be used to pass both the request and response transactions.
6. Ensure non-disclosure of sensitive data to unauthorized entities. For example, the Portal may not be authorized to view certain information in the request/response sequence.
7. Ensure integrity of data being transmitted between Client, Portal and SP. A secure and mutually authenticated tunnel will provide some implicit level of integrity.

Security Requirements (Variation 2):

1. Client authenticates Portal. Portal presents token to Client, and Client validates token.
2. Portal authenticates Client. Client presents token to Portal, and Portal validates token.
3. Portal determines an authorized SP for the Client, and redirects Client to that SP.
4. SP authenticates Client. Client presents token to SP, and SP validates token. (Note: The Client's token may be forwarded by the Portal to the SP.)
5. Protect any sensitive data (e.g., PI) transmitted between Client, Portal and SP.
6. Ensure non-disclosure of sensitive data to unauthorized entities. For example, the Portal may not be authorized to view certain information in the request/response sequence.
7. Ensure integrity of data being transmitted between Client, Portal and SP. A secure and mutually authenticated tunnel will provide some implicit level of integrity.

Possible Security Solutions

Authentication Solutions

- SSL/TLS – Client and Portal mutual authentication using digital certificates (i.e., *SSL server certificate* for Portal and client digital certificate for client).
- SSL/TLS – Client and Portal mutual authentication using a combination of digital certificates and username/passwords. Client authenticates Portal by validating Portal's SSL server certificate; Portal authenticates client by validating client's username/password (obtained from *HTTP header*).
- SSL/TLS – SP and Portal mutual authentication using digital certificates (i.e., SSL server certificates for Portal and SP).
- SSL/TLS – SP and Portal mutual authentication using a combination of digital certificates and username/passwords. SP authenticates Portal by validating Portal's SSL server certificate; Portal authenticates SP by validating SP's username/password (obtained from HTTP header).

Confidentiality Solutions

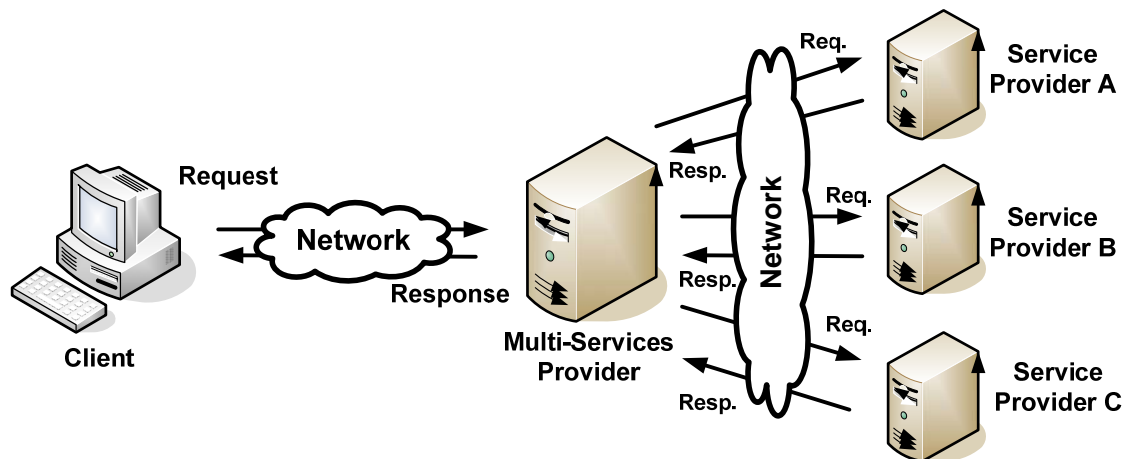
- SSL/TLS using minimum 1024 bit RSA public keys and 128 bit 3DES or AES. SSL/TLS provides for an encrypted tunnel to transport plaintext data.
- XML encryption to provide additional confidentiality on specific data elements within an XML document being transported through a secure tunnel.

Integrity Solutions

- SSL/TLS – Implicit integrity is achieved through establishment of a mutually authenticated SSL/TLS connection.
- XML signature – Additional data integrity can be provided on specific XML data elements within an XML document being transported through a secure tunnel.

Multi-Services Scenario

In this scenario, the client interacts with a multi-services provider (MSP) to receive the requested information. The client submits a single request to the MSP. The MSP then submits various requests to different SPs. In many cases the MSP uses the same document in each of its requests to the SPs to minimize the number of different requests that have to be created by the MSP. The SPs individually respond back to the MSP, which then creates and sends a single response back to the client.



Multi-Services - Scenario 1

Security Requirements

1. Client authenticates MSP. MSP presents token to Client, and Client validates token.
2. MSP authenticates Client. Client presents token to MSP, and MSP validates token.
3. MSP authenticates SP1. SP1 presents token to MSP, and MSP validates token.
4. SP1 authenticates MSP. MSP presents token to SP1, and SP1 validates token.
5. MSP authenticates SPn. SPn presents token to MSP, and MSP validates token.
6. SPn authenticates MSP. MSP presents token to SPn, and SPn validates token.
7. Protect any sensitive data (e.g., PI) transmitted between Client, MSP and SPs. Many MISMO transactions (DTDs) contain personal information (PI). These sensitive elements should be identified to ensure appropriate protection is being applied. At a minimum, a secure tunnel is to be used to pass both the request and response transactions.

8. Ensure non-disclosure of sensitive data to unauthorized entities. For example, a particular SP may not be authorized to view certain information in the request/response sequence, especially if the MSP is using a single XML document for all SP request/response sequences.
9. Ensure integrity of data being transmitted between Client, MSP and SPs. A secure and mutually authenticated tunnel will provide some implicit level of integrity.

Possible Security Solutions

Authentication Solutions

- SSL/TLS – Client and MSP mutual authentication using digital certificates (i.e., SSL server certificate for MSP and client digital certificate for client).
- SSL/TLS – Client and MSP mutual authentication using a combination of digital certificates and username/passwords. Client authenticates MSP by validating MSP's SSL server certificate; MSP authenticates client by validating client's username/password (obtained from HTTP header).
- SSL/TLS – MSP and SP mutual authentication using digital certificates (i.e., SSL server certificates for MSP and SP).
- SSL/TLS – MSP and SP mutual authentication using a combination of digital certificates and username/passwords. SP authenticates MSP by validating MSP's SSL server certificate; MSP authenticates SP by validating SP's username/password (obtained from HTTP header).
- Secure FTP – MSP and SP mutual authentication using a password or certificate based secured FTP connection.
- VPN over frame relay/dedicated lines using PKI digital certificates or Share Secrets for mutual authentication between MSP and SPs.
- Digitally signed email using S/MIME or PGP to provide mutual authentication between MSP and SPs.

Confidentiality Solutions

- SSL/TLS using minimum 1024 bit RSA public keys and 128 bit 3DES or AES. SSL/TLS provides for an encrypted tunnel to transport plaintext data.
- VPN using SSL/TLS or IPsec. VPN provides for an encrypted tunnel to transport plaintext data.
- Encrypted email using S/MIME or PGP and minimum 1024 bit RSA public keys and 128 bit 3DES or AES.

- XML encryption to provide additional confidentiality on specific data elements within an XML document being transported through a secure tunnel.

Integrity Solutions

- SSL/TLS – Implicit integrity is achieved through establishment of a mutually authenticated SSL/TLS connection.
- VPN using SSL or IPSec – Implicit integrity is achieved through establishment of a mutually authenticated VPN connection.
- S/MIME email – A digitally signed email also provides data integrity services.
- XML signature – Additional data integrity can be provided on specific XML data elements within an XML document being transported through a secure tunnel.

Security Definitions

3DES	Also referred to as <i>triple DES</i> , a mode of the DES encryption algorithm that encrypts data three times. Three 64-bit keys are used, instead of one, for an overall key length of 192 bits (the first encryption is encrypted with second key, and the resulting cipher text is again encrypted with a third key).
AES	Short for <i>Advanced Encryption Standard</i> , a symmetric 128-bit block data encryption algorithm.
Asymmetric encryption	In deploying asymmetric data encryption, the use of public and private key pairs to encrypt and subsequently de-encrypt the data is required. Use of these pairs of keys can provide for both confidentiality and authentication.
Digital signature	A digital code that can be attached to an electronically transmitted message that uniquely identifies the sender. Like a written signature, the purpose of a digital signature is to guarantee that the individual sending the message really is who he or she claims to be. Digital signatures are especially important for electronic commerce and are a key component of most authentication schemes.
DMZ	<p>Short for <i>demilitarized zone</i>, a computer or small subnetwork that sits between a trusted internal network, such as a corporate private LAN, and an untrusted external network, such as the public Internet.</p> <p>Typically, the DMZ contains devices accessible to Internet traffic, such as Web (HTTP) servers, FTP servers, SMTP (e-mail) servers and DNS servers.</p> <p>The term comes from military use, meaning a buffer area between two enemies.</p>

DTD	<p>Short for <i>document type definition</i>. A DTD states what tags and attributes are used to describe content in an SGML, XML or HTML document, where each tag is allowed, and which tags can appear within other tags. For example, in a DTD one could say that LIST tags can contain ITEM tags, but ITEM tags cannot contain LIST tags. In some editors, when authors are inputting information, they can place tags only where the DTD allows. This ensures that all the documentation is formatted the same way. Applications will use a document's DTD to properly read and display a document's contents. Changes in the format of the document can be easily made by modifying the DTD.</p>
Encryption	<p>The translation of data into a secret code. Encryption is the most effective way to achieve data security. To read an encrypted file, you must have access to a secret key or password that enables you to decrypt it. Unencrypted data is called <i>plain text</i> ; encrypted data is referred to as <i>cipher text</i>.</p> <p>There are two main types of encryption: asymmetric encryption (also called public-key encryption) and symmetric encryption</p>
Frame Relay	<p>An efficient data transmission technique used to send digital information quickly and cheaply in a relay of frames to one or many destinations from one or many end-points. Network providers commonly implement frame relay for voice and data as an encapsulation technique, used <i>between</i> local area networks (LANs) <i>over</i> a wide area network (WAN). Each end-user gets a private line (or leased line) to a frame-relay node. The frame-relay network handles the transmission over a frequently-changing path transparent to all end-users.</p>

Hash/digest	<p>Producing <i>hash values</i> for accessing data or for security. A hash value (or simply <i>hash</i>), also called a <i>message digest</i>, is a number generated from a string of text. The hash is generated by a formula in such a way that it is extremely unlikely that some other text will produce the same hash value.</p> <p>Hashes play a role in security systems where they're used to ensure that transmitted messages have not been tampered with. The sender generates a hash of the message, encrypts it, and sends it with the message itself. The recipient then decrypts both the message and the hash, produces another hash from the received message, and compares the two hashes. If they're the same, there is a very high probability that the message was transmitted intact.</p>
HTTP	<p>Short for <i>HyperText Transfer Protocol</i>, the underlying protocol used by the World Wide Web. HTTP defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands. For example, when you enter a URL in your browser, this actually sends an HTTP command to the Web server directing it to fetch and transmit the requested Web page.</p>
IPsec	<p>Short for <i>IP Security</i>, a set of protocols developed by the IETF to support secure exchange of packets at the IP layer. IPsec has been deployed widely to implement Virtual Private Networks (VPNs).</p> <p>IPsec supports two encryption modes: Transport and Tunnel. Transport mode encrypts only the data portion (<i>payload</i>) of each packet, but leaves the header untouched. The more secure Tunnel mode encrypts both the header and the payload. On the receiving side, an IPSec-compliant device decrypts each packet.</p>

PGP	<p>Pretty Good Privacy, abbreviated as <i>PGP</i>, a technique developed by Philip Zimmerman for encrypting messages. PGP is one of the most common ways to protect messages on the Internet because it is effective, easy to use, and free. PGP is based on the public-key method, which uses two keys -- one is a public key that you disseminate to anyone from whom you want to receive a message. The other is a private key that you use to decrypt messages that you receive.</p>
Pharming	<p>Pharming seeks to obtain personal or private (usually financial related) information through domain spoofing. Pharming 'poisons' a DNS server by infusing false information into it, resulting in a user's request being redirected elsewhere. Your browser, however will show you are at the correct Web site, which makes pharming a bit more serious and more difficult to detect..</p>
PKI digital certificate	<p>Short for <i>public key infrastructure</i>, a system of digital certificates, Certificate Authorities, and other registration authorities that verify and authenticate the validity of each party involved in an Internet transaction. PKIs are currently evolving and there is no single PKI nor even a single agreed-upon standard for setting up a PKI.</p>
RSA public keys	<p>An public-key encryption technology developed by RSA Data Security, Inc. The acronym stands for Rivest, Shamir, and Adelman, the inventors of the technique. The RSA algorithm is based on the fact that there is no efficient way to factor very large numbers. Deducing an RSA key, therefore, requires an extraordinary amount of computer processing power and time.</p>

S/MIME	<p>Short for <i>Multipurpose Internet Mail Extensions</i>, a specification for formatting non-ASCII messages so that they can be sent over the Internet. Many e-mail clients now support MIME, which enables them to send and receive graphics, audio, and video files via the Internet mail system. In addition, MIME supports messages in character sets other than ASCII.</p> <p>There are many predefined MIME types, such as GIF graphics files and PostScript files. It is also possible to define your own MIME types.</p> <p>In addition to e-mail applications, Web browsers also support various MIME types. This enables the browser to display or output files that are not in HTML format.</p> <p>MIME was defined in 1992 by the Internet Engineering Task Force (IETF). A new version, called S/MIME, supports encrypted messages.</p>
Secure FTP	See SFTP/SSH
Security domain	The subset of data required by a party to fulfill their respective portion of the request.
SFTP/SSH	<p>In computing, Secure Shell or SSH is a set of standards and an associated network protocol that allows establishing a secure channel between a local and a remote computer. It uses public-key cryptography to authenticate the remote computer and (optionally) to allow the remote computer to authenticate the user. SSH provides confidentiality and integrity of data exchanged between the two computers using encryption and message authentication codes.</p> <p>FTP over SSH is sometimes referred to as secure FTP or SFTP.</p>

SSL	<p>SSL, Short for <i>Secure Sockets Layer</i>, a protocol developed by Netscape for transmitting private documents via the Internet. SSL uses a cryptographic system that uses public and private key pairs to encrypt data. Both Netscape Navigator and Internet Explorer support SSL, and many Web sites use the protocol to obtain confidential user information, such as credit card numbers. By convention, URLs that require an SSL connection start with <i>https</i>: instead of <i>http</i>:</p>
Symmetric encryption	<p>In deploying data encryption, the use of a key to de-encrypt the data is required. When utilizing a symmetric algorithm, both the sender and receiver of the data share a common key.</p>
TLS	<p>Short for <i>Transport Layer Security</i>, a protocol that guarantees privacy and data integrity between client/server applications communicating over the Internet.</p> <p>The TLS protocol is made up of two layers:</p> <ul style="list-style-type: none">• The <i>TLS Record Protocol</i> -- layered on top of a reliable transport protocol, such as TCP, it ensures that the connection is private by using symmetric data encryption and it ensures that the connection is reliable. The TLS Record Protocol also is used for encapsulation of higher-level protocols, such as the TLS Handshake Protocol.• The <i>TLS Handshake Protocol</i> -- allows authentication between the server and client and the negotiation of an encryption algorithm and cryptographic keys before the application protocol transmits or receives any data.
VPN	<p>Short for <i>virtual private network</i>, a <i>network</i> that is constructed by using public wires to connect nodes. For example, there are a number of systems that enable you to create networks using the Internet as the medium for transporting data. These systems use encryption and other security mechanisms to ensure that only authorized users can access the network and that the data cannot be intercepted.</p>

XML
encryption

An encryption method for XML data that offers the ability to selectively encrypt certain data elements, while leaving the remainder in clear text.

Chapter 7: PRIA DTD Best Practices

Below you will find elements and attributes of elements that most commonly apply to “Best Practices” minimum requirements for electronic recordation. Elements are positional by the DTD (Document Type Definition). Please see the DTD at the end of this document to show how all elements and attributes are applied within the DTD.

PRIA DTD

The following nomenclature shall be used to define the following sections in this chapter:

Element

DTD Definition

Examples

Child Elements

Attributes

Deprecations

[PRIA_DOCUMENT] Element

The PRIA DOCUMENT element contains information about the document itself.

PRIA_DOCUMENT DTD Definition

```
<!ELEMENT PRIA_DOCUMENT (GRANTOR+, GRANTEE+, PROPERTY*, PARTIES, EXECUTION,
MORTGAGE_CONSIDERATION?, CONSIDERATION*, RECORDABLE_DOCUMENT*,
SIGNATORY*, NOTARY*, RECORDING_ENDORSEMENT?, EMBEDDED_FILE*)>
<!ATTLIST PRIA_DOCUMENT
  _ID ID #IMPLIED
  _Code CDATA #IMPLIED
  DocumentNonRecordableIndicator (Y | N) #IMPLIED
  PRIAVersionIdentifier CDATA #IMPLIED
  _UniqueIdentifier CDATA #IMPLIED
  RecordableDocumentSequenceIdentifier CDATA #IMPLIED
  RecordableDocumentType (AbstractOfJudgment | AffidavitOfDeath | Assignment |
AssignmentOfMortgage | AssignmentOfDeedOfTrust | BargainAndSaleDeed | BlanketAssignment |
Deed | DeedOfTrust | FederalTaxLien | Judgment |
ModificationAgreementOrConsolidationAgreement | Mortgage | Other | PartialSatisfactionOfLien |
PowerOfAttorney | QuitClaimDeed | Reconveyance | ReleaseOfLien | ReleaseOfFederalTaxLien |
ReleaseOfStateTaxLien | ReleaseOfTreasurersTaxLien | SatisfactionOfLien |
SatisfactionOfMortgage | SecurityInstrument | SignatureAffidavit | StateTaxLien |
SubordinateLienAgreement | SubstitutionOfTrustee | TreasurersTaxLien | WarrantyDeed)
#IMPLIED
  RecordableDocumentTypeOtherDescription CDATA #IMPLIED>
```

PRIA_DOCUMENT Examples

Please see the following example listed below.

None

Child Elements:

[GRANTOR]
 [GRANTEE]
 [PROPERTY]
 [PARTIES]
 [EXECUTION]
 [MORTGAGE_CONSIDERATION]
 [CONSIDERATION]
 [RECORDABLE_DOCUMENT]
 [SIGNATORY]
 [NOTARY]
 [RECORDING_ENDORSEMENT]
 [EMBEDDED_FILE]

PRIA_DOCUMENT Attributes

[_Code] Attribute

Abbreviated code used by recorder in indexing software. Specific to County document is being recorded in.

Examples: Warranty = WD, Quit Claim Deed = QCD, Mortgage = MTG, Release of Lien = REL

[DocumentNonRecordableIndicator] Attribute

An indicator that this document is included in the electronic package but is not to be recorded by the County.

[PRIAVersionIdentifier] Attribute

The PRIA DOCUMENT version.

[_Unique Identifier] Attribute

A unique system-assigned identifier (such as an audit trail number) to each document.

[RecordableDocumentSequenceIdentifier] Attribute

Identifier used to indicate the position of a document within a sequence of documents.

- If this attribute is blank, it is assumed that the order of the documents is arbitrary – recording of specific documents can take place in any order.

[RecordableDocumentType] Attribute

The type of document that is to be recorded.

- One of the values in the enumerated list of document types.
- “Other” may be used if a suitable enumerated value is not available.

[RecordableDocumentTypeOtherDescription] Attribute

The description of type of document when the Recordable Document Type enumerated value selected is “Other”.

PRIA_DOCUMENT Deprecations

None

[GRANTOR] Element

The GRANTOR element contains information pertaining to a single individual or business referenced on a document.

GRANTOR DTD Definition

```
<!ELEMENT GRANTOR (_ALIAS*)>
<!ATTLIST GRANTOR
  _ID ID #IMPLIED
  _StreetAddress CDATA #IMPLIED
  _StreetAddress2 CDATA #IMPLIED
  _City CDATA #IMPLIED
  _State CDATA #IMPLIED
  _PostalCode CDATA #IMPLIED
  _County CDATA #IMPLIED
  _Country CDATA #IMPLIED
  _FirstName CDATA #IMPLIED
  _MiddleName CDATA #IMPLIED
  _LastName CDATA #IMPLIED
  _NameSuffix CDATA #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _CapacityDescription CDATA #IMPLIED
  MaritalStatusType (Married | NotProvided | Divorced | Separated | Unknown | Unmarried)
  #IMPLIED
  NonPersonEntityIndicator (Y | N) #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED>
```

GRANTOR Examples

Please see the following examples listed below.

Person / Parsed

```
<GRANTOR _FirstName="Fred" _LastName="Flintstone" _MiddleName="R" />
```

Person / Unparsed

```
<GRANTOR _UnparsedName="Fred Flintstone" />
```

Business / Unparsed

```
<GRANTOR _UnparsedName="ACME Bank Inc." _NonPersonEntityIndicator="Y" />
```

Child Elements:

[ALIAS]

GRANTOR Attributes

[_StreetAddress] Attribute

The unstructured (Unparsed) street address of the Grantor (e.g., 123 Main Street).

[_StreetAddress2] Attribute

The unstructured (Unparsed) street address 2nd division of the Grantor (e.g., Unit 1223).

[_City] Attribute

The city in which the Grantor is located.

[_State] Attribute

The state in which the Grantor is located.

[_PostalCode] Attribute

The postal code (zip code in the US) of the Grantor. Zip code may be either 5 or 9 digits.

[_County] Attribute

The county in which the Grantor is located.

[_Country] Attribute

The name of the country in which the Grantor resides.

[_FirstName] Attribute

The first name of the GRANTOR

[_MiddleName] Attribute

The middle name (or initial) of the GRANTOR

[_LastName] Attribute

The last name of the GRANTOR

[_NameSuffix] Attribute

The suffix or lineage of the GRANTOR (ie: Jr, Sr, III)

[_UnparsedName] Attribute

The unparsed name of the GRANTOR

- Since most indexing systems accommodate parsed names it is recommended that this attribute be used for Non Person Entities. When this recommendation can not be followed, business partners should be notified of this early in the implementation cycle.

[_CapacityDescription] Attribute

The capacity in which the GRANTOR is acting (ie: Power of Attorney, Trustee)

[MaritalStatusType] Attribute

The marital status of the party as disclosed by the party

[NonPersonEntityIndicator] Attribute

When true, indicates that the party is a non person entity.

[_SequenceIdentifier] Attribute

When multiple GRANTOR's are included this indicates the order in which they should appear on the document

- If this attribute is blank, the natural order of multiple Grantors is assumed to be the order in which they should appear for display and indexing.

GRANTOR Deprecations

None

[_ALIAS] Element

The _ALIAS element contains information about alternative names.

The leading underscore indicates that its context is derived from its parent element. In the PRIA_DOCUMENT DTD it is used with both, GRANTOR and GRANTEE.

ALIAS DTD Definition

```
<!ELEMENT _ALIAS EMPTY>
<!ATTLIST _ALIAS
  _ID ID #IMPLIED
  _FirstName CDATA #IMPLIED
  _MiddleName CDATA #IMPLIED
  _LastName CDATA #IMPLIED
  _NameSuffix CDATA #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  AliasType (FormerlyKnownAs | NowKnownAs | AlsoKnownAs) #IMPLIED
  AliasTypeOtherDescription CDATA #IMPLIED>
```

ALIAS Examples

Please see the following example below.

```
<GRANTOR _FirstName="Fred" _LastName="Flintstone" _MiddleName="R">
  <_ALIAS _FirstName="Fredrick" _LastName="Flintstone" _AliasType="AlsoKnownAs" />
  <_ALIAS _FirstName="Freddy" _LastName="Flintstone" _AliasType="AlsoKnownAs" />
</GRANTOR>
```

Child Elements:

None

ALIAS Attributes

[_FirstName] Attribute

Alternate first name of the GRANTOR or GRANTEE

[_MiddleName] Attribute

Alternate middle name (or initial) of the GRANTOR or GRANTEE

[_LastName] Attribute

Alternate last name of the GRANTOR or GRANTEE

[_NameSuffix] Attribute

Suffix or Lineage of the alternate name of the GRANTOR or GRANTEE (ie: Jr, Sr, III)

[_UnparsedName] Attribute

Alternate unparsed name of the GRANTOR or GRANTEE

[_SequenceIdentifier] Attribute

When multiple GRANTOR or GRANTEE ALIAS's are included this indicates the order in which they should appear on the document

[AliasType] Attribute

A description of the type of ALIAS

[AliasTypeOtherDescription] Attribute

A description of an ALIAS type not included in the enumerated list

ALIAS Deprecations

None

[GRANTEE] Element

The GRANTEE element contains information pertaining to a single individual or business referenced on a document.

GRANTEE DTD Definition

```
<!ELEMENT GRANTEE (_ALIAS*)>
<!ATTLIST GRANTEE
  _ID ID #IMPLIED
  _StreetAddress CDATA #IMPLIED
  _StreetAddress2 CDATA #IMPLIED
  _City CDATA #IMPLIED
  _State CDATA #IMPLIED
  _PostalCode CDATA #IMPLIED
  _County CDATA #IMPLIED
  _Country CDATA #IMPLIED
  _FirstName CDATA #IMPLIED
  _MiddleName CDATA #IMPLIED
  _LastName CDATA #IMPLIED
  _NameSuffix CDATA #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _CapacityDescription CDATA #IMPLIED
  MaritalStatusType (Married | NotProvided | Divorced | Separated | Unknown | Unmarried)
  #IMPLIED
  NonPersonEntityIndicator (Y | N) #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED>
```

GRANTEE Examples

Please see the following example listed below.

Person / Parsed

```
<GRANTEE _FirstName="Fred" _LastName="Flintstone" _MiddleName="R" />
```

Person / Unparsed

```
<GRANTEE _UnparsedName="Fred Flintstone" />
```

Business / Unparsed

```
<GRANTEE _UnparsedName="ACME Bank Inc." _NonPersonEntityIndicator="Y" />
```

Child Elements:

[ALIAS]

GRANTEE Attributes

[_StreetAddress] Attribute

The unstructured (Unparsed) street address of the Grantee (e.g., 123 Main Street).

[_StreetAddress2] Attribute

The unstructured (Unparsed) street address 2nd division of the Grantee (e.g., Unit 1223).

[_City] Attribute

The city in which the Grantee is located.

[_State] Attribute

The state in which the Grantee is located.

[_PostalCode] Attribute

The postal code (zip code in the US) of the Grantee. Zip code may be either 5 or 9 digits.

[_County] Attribute

The county in which the Grantee is located.

[_Country] Attribute

The name of the country in which the Grantee resides.

[_FirstName] Attribute

The first name of the GRANTEE

[_MiddleName] Attribute

The middle name (or initial) of the GRANTEE

[_LastName] Attribute

The last name of the GRANTEE

[_NameSuffix] Attribute

The suffix or lineage of the GRANTEE (ie: Jr, Sr, III)

[_UnparsedName] Attribute

The unparsed name of the GRANTEE

- Since most indexing systems accommodate parsed names it is recommended that this attribute be used for Non Person Entities. When this recommendation can not be followed, business partners should be notified of this early in the implementation cycle.

[_CapacityDescription] Attribute

The capacity in which the GRANTEE is acting (ie: Power of Attorney, Trustee)

[_MaritalStatusType] Attribute

The marital status of the party as disclosed by the party

[_NonPersonEntityIndicator] Attribute

When true, indicates that the party is a non person entity.

[_SequenceIdentifier] Attribute

When multiple GRANTEE's are included this indicates the order in which they should appear on the document

- If this attribute is blank, the natural order of multiple Grantors is assumed to be the order in which they should appear for display and indexing.

GRANTEE Deprecations

None

[PROPERTY] Element

The PROPERTY element contains information pertaining to a property. This element and its contents are shared with the MISMO Real Property Information Workgroup (REPI) and the MISMO Origination Workgroup.

PROPERTY DTD Definition

```
<!ELEMENT PROPERTY (_IDENTIFICATION?, PARSED_STREET_ADDRESS?,
_LEGAL_DESCRIPTION*)>
<!ATTLIST PROPERTY
  _ID ID #IMPLIED
  MISMOVersionId CDATA #IMPLIED
  _StreetAddress CDATA #IMPLIED
  _StreetAddress2 CDATA #IMPLIED
  _City CDATA #IMPLIED
  _State CDATA #IMPLIED
  _PostalCode CDATA #IMPLIED
  _County CDATA #IMPLIED
  _Country CDATA #IMPLIED
  BuildingStatusType (Complete | Existing | Other | Proposed | SubjectToAlteration |
SubjectToAlterationImprovementRepairAndRehabilitation | SubstantiallyRehabilitated |
UnderConstruction | Incomplete) #IMPLIED
  LandUseType (Residential | Income | Commercial | Industrial | Vacant | Agricultural |
PublicAndSemipublic | Recreational | TransportationAndUtility | Other) #IMPLIED
  LandUseTypeOtherDescription CDATA #IMPLIED
  LandUseComment CDATA #IMPLIED
  _CurrentOccupancyType (Abandoned | AdverseOccupied | OwnerOccupied |
RemovedOrDestroyed | TennantOccupied | Unknown | Vacant) #IMPLIED
  _CurrentOccupantUnparsedName CDATA #IMPLIED
  _GatedCommunityIndicator (Y | N) #IMPLIED
  _RightsType (CommunityProperty | FeeSimple | JointTenants | Leasehold | Other |
TenantsInCommon | Unassigned) #IMPLIED
  _TaxDelinquentIndicator (Y | N) #IMPLIED
  _TitleCategoryType (Church | CommercialNonResidential | Condominium |
CondominiumOverFourStories | Cooperative | Farm | HomeAndBusinessCombined |
ManufacturedMobileHome | MixedUseResidential | MultifamilyMoreThanFourUnits | Other |
SingleFamily | Townhouse | TwoToFourUnitProperty | VacantLand) #IMPLIED
  _TitleCategoryTypeOtherDescription CDATA #IMPLIED>
```

PROPERTY Examples

Please see the following example listed below.

None

Child Elements

[IDENTIFICATION]

[PARSED_STREET_ADDRESS]

[LEGAL_DESCRIPTION]

PROPERTY Attributes

[_StreetAddress] Attribute

The unstructured (Unparsed) street address of the subject property (e.g., 123 Main Street).

- The addressing attributes under Property are likely to be sufficient for most county recording applications and should thus be the primary attributes used for this information. However, if parsed address information is required, then the Parsed Street Address element can be utilized – noting, however that it does not contain city and state attributes.

[_StreetAddress2] Attribute

The unstructured (Unparsed) street address 2nd division of the subject property (e.g., Unit 1223).

[_City] Attribute

The city in which the subject property is located.

[_State] Attribute

The state in which the subject property is located.

[_PostalCode] Attribute

The postal code (zip code in the US) of the subject property. Zip code may be either 5 or 9 digits.

[_County] Attribute

The county in which the subject property is located.

[_Country] Attribute

The name of the country in which the subject property resides.

[BuildingStatusType] Attribute

Specifies the physical status of the structure.

[LandUseType] Attribute

Identifies the designation of land use by a governmental authority described by Land Used Description.

[LandUseTypeOtherDescription] Attribute

A free-form text field used to describe the land use designation if Other is selected as the Property Land Use Type.

[LandUseComment] Attribute

A free-form text field used to further describe the land use of the property as defined by Land Use Type.

[_CurrentOccupancyType] Attribute

Specifies the property occupancy status of a subject property.

[_CurrentOccupantUnparsedName] Attribute

The unparsed name of the property's current occupant.

[_GatedCommunityIndicator] Attribute

Indicates that the property is within a gated community

[_RightsType] Attribute

Specifies the intended property ownership rights for the property.

[_TaxDelinquentIndicator] Attribute

Indicates that the owner is delinquent on taxes from a taxing authority or jurisdiction.

[_TitleCategoryType] Attribute

Specifies the broad title category for the subject property.

[_TitleCategoryTypeOtherDescription]

A free-form text field used to capture the property title category type if Other is selected as the property title category type.

PROPERTY Deprecations

None

[_IDENTIFICATION] Element

The _IDENTIFICATION element contains county and state FIPS codes, Census Tract and other local identifiers pertaining to a property.

__IDENTIFICATION DTD Definition

```
<!ELEMENT _IDENTIFICATION EMPTY>
<!ATTLIST _IDENTIFICATION
  _ID ID #IMPLIED
  CountyFIPSCode CDATA #IMPLIED
  StateFIPSCode CDATA #IMPLIED
  CensusTractIdentifier CDATA #IMPLIED
  MSAIdentifier CDATA #IMPLIED
  MunicipalityName CDATA #IMPLIED
  SchoolDistrictName CDATA #IMPLIED
  LongitudeNumber CDATA #IMPLIED
  LatitudeNumber CDATA #IMPLIED
  MapReferenceIdentifier CDATA #IMPLIED
  MapReferenceSecondIdentifier CDATA #IMPLIED
  AssessorsParcelIdentifier CDATA #IMPLIED
  AssessorsSecondParcelIdentifier CDATA #IMPLIED
  AssessorsParcelIdentifierSequenceIdentifier CDATA #IMPLIED>
```

__IDENTIFICATION Examples

Please see the following example listed below.

None

Child Elements

None

__IDENTIFICATION Attributes

[_CountyFIPSCode] Attribute

The County FIPS Code of the subject property.

[_StateFIPSCode] Attribute

The State FIPS code of the subject property.

[CensusTractIdentifier] Attribute

Identifies census tract as defined by the U.S. Census Bureau where subject property is located.

[MSAIdentifier] Attribute

Identifies Metropolitan Statistical Area (MSA) where subject property is located. A MSA is a contiguous geographic area consisting of an urban center city and its surrounding suburbs.

[MunicipalityName] Attribute

The name of the municipality in which property is located.

[SchoolDistrictName] Attribute

The name of the school district in which the property is located.

[LongitudeNumber] Attribute

The X value of the geographic coordinate system using geodetic model based on North American datum of 1983.

[LatitudeNumber] Attribute

The Y value of the geographic coordinate system using geodetic model based on North American datum of 1983.

[MapReferenceIdentifier] Attribute

A reference to a regionally specific map document that assists in locating a property. May refer to locally available published map products (e.g. Thomas Map in CA) or a county tax map.

- In some jurisdictions, this may be part of the legal description

[MapReferenceSecondIdentifier] Attribute

A secondary reference to a regionally specific map document that assists in locating a property. May refer to locally available published map products (e.g. Thomas Map in CA) or a county tax map.

- In some jurisdictions, this may be part of the legal description

[AssessorsParcelIdentifier] Attribute

The identifier that describes the location of the property as related to county, state or municipal tax records.

- All references in the DTD to "Assessor__" are generic references to the office and should be utilized in conjunction with local practices which may then apply to an Auditor, Tax Lister or Treasurer's office.
- IMPORTANT: It is recommended that the attributes under PARCEL_IDENTIFICATION be utilized for parcel identification numbers as that structure accommodates more use cases than the attributes in this element.

[AssessorsSecondParcelIdentifier] Attribute

Second parcel identifier when the subject property has more than one.

- IMPORTANT: It is recommended that the attributes under PARCEL_IDENTIFICATION be utilized for parcel identification numbers as that structure accommodates more use cases than the attributes in this element.

[AssessorsParcelIdentifierSequenceIdentifier] Attribute

If more than one Identifier is listed, this indicates the order they should appear on the document.

- IMPORTANT: It is recommended that the attributes under PARCEL_IDENTIFICATION be utilized for parcel identification numbers as that structure accommodates more use cases than the attributes in this element.

_IDENTIFICATION Deprecations

None

[PARSED_STREET_ADDRESS] Element

The PARSED_STREET_ADDRESS element contains parsed address information about a property.

PARSED_STREET_ADDRESS DTD Definition

```
<!ELEMENT PARSED_STREET_ADDRESS EMPTY>
<!ATTLIST PARSED_STREET_ADDRESS
  _ID ID #IMPLIED
  _StreetName CDATA #IMPLIED
  _DirectionPrefix CDATA #IMPLIED
  _DirectionSuffix CDATA #IMPLIED
  _StreetSuffix CDATA #IMPLIED
  _HouseNumber CDATA #IMPLIED
  _BuildingNumber CDATA #IMPLIED
  _PostOfficeBox CDATA #IMPLIED
  _MilitaryAPO_FPO CDATA #IMPLIED
  _ApartmentOrUnit CDATA #IMPLIED
  _RuralRoute CDATA #IMPLIED
  PlusFourPostalCode CDATA #IMPLIED
  SitusCarrierRouteIdentifier CDATA #IMPLIED>
```

PARSED_STREET_ADDRESS Examples

Please see the following example listed below.

None

Child Elements

None

PARSED_STREET_ADDRESS Attributes

Note: Addressing attributes under Property element are to be used as the primary addressing mechanism. This element can be used for detailed addressing information if available and needed.

[_StreetName] Attribute

The street name component of an address without any direction or quadrant indicators or street suffixes.

[_DirectionPrefix] Attribute

The Direction Prefix address component that appears before the Street Name. (N, S, E, W, NW, SW, NE, SE).

[_DirectionSuffix] Attribute

The Direction Suffix address component that appears after the Street Name. (N, S, E, W, NW, SW, NE, SE).

[_StreetSuffix] Attribute

The street suffix or street type component of an address (ST, AV, BV, CT, DR, PL, etc.) Use U.S. Postal Service approved abbreviations.

[_HouseNumber] Attribute

The house number address component.

[_BuildingNumber] Attribute

A building number component of the street address.

[_PostOfficeBox] Attribute

The post office box address component of the address, it is normally part of a rural route address.

[_MilitaryAPO_FPO] Attribute

US Military addressing system

[_ApartmentOrUnit] Attribute

The Apartment, Suite or Unit number component of the address.

[_RuralRoute] Attribute

The rural route or start route number component of an address.

[PlusFourPostalCode] Attribute

This is the four-digit postal code extension to the 5 digit postal code. Together these make up the complete Postal Code of 9 digits.

[SitusCarrierRouteIdentifier] Attribute

This is the carrier route postal code used for mass-mailing.

PARSED_STREET_ADDRESS Deprecations

None

[_LEGAL_DESCRIPTION] Element

The _LEGAL_DESCRIPTION Element contains surveying/cartographic information about a property.

LEGAL_DESCRIPTION DTD Definition

```
<!ELEMENT _LEGAL_DESCRIPTION (PARCEL_IDENTIFICATION, PLATTED_LAND?,  
UNPLATTED_LAND?)>  
<!ATTLIST _LEGAL_DESCRIPTION  
  _ID ID #IMPLIED  
  _TextDescription CDATA #IMPLIED  
  _Type (LongLegal | MetesAndBounds | Other | ShortLegal) #IMPLIED  
  _TypeOtherDescription CDATA #IMPLIED>
```

LEGAL_DESCRIPTION Examples

Please see the following example listed below.

None

Child Elements:

[[PARCEL_IDENTIFICATION](#)]
[[PLATTED_LAND](#)]
[[UNPLATTED_LAND](#)]

LEGAL_DESCRIPTION Attributes

Note: It is recommended that these attributes only be used when the attributes included in the child elements are not sufficient for the local application.

[_TextDescription] Attribute

A free-form text field used to capture the legal description of the property for the type specified in Property Legal Description Type.

[_Type] Attribute

Specifies the type of legal description of the property

[_TypeOtherDescription] Attribute

A free form text field used to collect additional information when "Other" is selected for Property Legal Description Type.

_LEGAL_DESCRIPTION Deprecations

None

[PARCEL_IDENTIFICATION] Element

The PARCEL_IDENTIFICATION Element contains information about a unique number, usually assigned by a municipality to identify a property in a system database.

Note: This is the preferred element for use with parcel identification numbers. Other attributes that appear in the DTD were inherited from MISMO and apply to specific use cases which may not fully accommodate the needs of recordable documents.

PARCEL_IDENTIFICATION DTD Definition

```
<!ELEMENT PARCEL_IDENTIFICATION EMPTY>
<!ATTLIST PARCEL_IDENTIFICATION
  _ID ID #IMPLIED
  _Type (ParcelIdentificationNumber | TaxMapIdentifier | TaxParcelIdentifier |
  AssessorUnformattedIdentifier | TorrensCertificateIdentifier | Other) #IMPLIED
  _TypeOtherDescription CDATA #IMPLIED
  _Identifier CDATA #IMPLIED>
```

PARCEL_IDENTIFICATION Examples

Please see the following example listed below.

None

Child Elements:

None

PARCEL_IDENTIFICATION Attributes

[_Type] Attribute

Specifies other parcel identification types besides Assessors Parcel identifiers that are used by taxing authorities or others to identify a parcel of property.

[_TypeOtherDescription] Attribute

A free-form text field used to describe the parcel identification if Other is selected as the Parcel Identification Type.

[_Identifier] Attribute

A free-form text field used to hold the actual identification value of the type specified by Parcel Identification Type.

PARCEL_IDENTIFICATION Deprecations

None

[PLATTED_LAND] Element

The PLATTED_LAND element contains reference information derived from plat maps that constitute the legal description of a property.

PLATTED_LAND DTD Definition

```
<!ELEMENT PLATTED_LAND EMPTY>
<!ATTLIST PLATTED_LAND
  _ID ID #IMPLIED
  _AdditionalParcelIdentifier CDATA #IMPLIED
  _AdditionalParcelDescription CDATA #IMPLIED
  _AppurtenanceDescription CDATA #IMPLIED
  _AppurtenanceIdentifier CDATA #IMPLIED
  _BuildingIdentifier CDATA #IMPLIED
  _PlatCodeIdentifier CDATA #IMPLIED
  _PlatInstrumentIdentifier CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _Type (Subdivision | Condominium | Timeshare | Other) #IMPLIED
  _TypeOtherDescription CDATA #IMPLIED
  _UnitNumberIdentifier CDATA #IMPLIED
  PropertyLotIdentifier CDATA #IMPLIED
  PropertyBlockIdentifier CDATA #IMPLIED
  PropertySectionIdentifier CDATA #IMPLIED
  PropertySubdivisionIdentifier CDATA #IMPLIED
  PropertyTractIdentifier CDATA #IMPLIED
  PlatName CDATA #IMPLIED
  RecordedDocumentBook CDATA #IMPLIED
  RecordedDocumentPage CDATA #IMPLIED>
```

PLATTED_LAND Examples

Please see the following example listed below.

Lot 1, Block 1 of Ives Grove Subdivision

```
<PLATTED_LAND _Type="Subdivision" PropertyLotIdentifier="1" PropertyBlockIdentifier="1"
  PlatName="Ives Grove Subdivision" _PlatCodeIdentifier="I105" />
```

Unit 4, Building 1 of Sundance Heights Condominium including Parking Space 104

```
<PLATTED_LAND _Type="Condominium" _UnitNumberIdentifier="4" _BuildingIdentifier="1"
  PlatName="Sundance Heights Condominium" _PlatCodeIdentifier="S210"
  _AppurtenanceDescription="Parking Space" _AppurtenanceIdentifier="104" />
```

Child Elements

None

PLATTED_LAND Attributes

[_AdditionalParcelIdentifier] Attribute

If more than one Parcel Identifier is required, additional identifiers are placed here.

- **IMPORTANT:** It is recommended that the attributes under PARCEL_IDENTIFICATION be utilized for parcel identification numbers as that structure accommodates more use cases than the attributes in this element.

[_AdditionalParcelDescription] Attribute

If more than one Parcel Description is required, additional descriptions are placed here.

[_AppurtenanceDescription] Attribute

A description of the appurtenance. i.e., Parking Space, Garage, Boat Slip

- **Note:** This definition is different from what appears in the LDD. The LDD definition was inherited from MISMO and is incorrect. This will be updated in the next version of the LDD.

[_AppurtenanceIdentifier] Attribute

The number assigned to a specific appurtenance within a plat.

- **Note:** This definition is different from what appears in the LDD. The LDD definition was inherited from MISMO and is incorrect. This will be updated in the next version of the LDD.

[_BuildingIdentifier] Attribute

The building number used to identify a building within a legal description. This may contain letters as well as digits.

[_PlatCodeIdentifier] Attribute

The unique code value assigned to a plat in a county's indexing system. Used to identify a specific plat to the indexing database.

- **Note:** This definition is different from what appears in the LDD. The LDD definition was inherited from MISMO and is incorrect. This will be updated in the next version of the LDD.

[_PlatInstrumentIdentifier] Attribute

The unique number assigned to the plat when it was originally recorded.

- **Note:** This definition is different from what appears in the LDD. The LDD definition was inherited from MISMO and is incorrect. This will be updated in the next version of the LDD.

[_SequenceIdentifier] Attribute

If more than one platted land legal description is used this indicates which order they should appear on the document.

[_Type] Attribute

This indicates what type of platted land is being described. While similar, it can be different from type of ownership.

[TypeOtherDescription] Attribute

The description of the type of platting used when "Other" is selected from the enumerated list.

[_UnitNumberIdentifier] Attribute

The unit number used to identify a condominium or other unit within a project in the legal description of the property.

[PropertyLotIdentifier] Attribute

Legal Description Lot number of property being financed.

[PropertyBlockIdentifier] Attribute

Legal description block number of property being financed.

[PropertySectionIdentifier] Attribute

Legal Description Section number of property being financed.

- This is actually part of an Unplatted Legal Description. It is recommended that this attribute not be used as it will be deprecated in a future release.

[PropertySubdivisionIdentifier] Attribute

The legal description subdivision name of property being financed.

[PropertyTractIdentifier] Attribute

A tract identifier used to legally identify a property. This identifier may be different than the census tract identifier.

[PlatName] Attribute

The name used to identify the plat in the property's legal description.

[RecordedDocumentBook] Attribute

The book or liber of the public record where the document is recorded.

[RecordedDocumentPage] Attribute

The page of the volume of the book of the public record where the document is recorded.

PLATTED_LAND Deprecations

None

[UNPLATTED_LAND] Element

The UNPLATTED_LAND element contains surveying/cartographic information which constitute the legal description of a property.

UNPLATTED_LAND DTD Definition

```
<!ELEMENT UNPLATTED_LAND EMPTY>
<!--LIST UNPLATTED_LAND
  _ID ID #IMPLIED
  _AbstractNumberIdentifier CDATA #IMPLIED
  _BaseIdentifier CDATA #IMPLIED
  _LandGrantIdentifier CDATA #IMPLIED
  _MeridianIdentifier CDATA #IMPLIED
  _MetesAndBoundsRemainingDescription CDATA #IMPLIED
  _QuarterSectionIdentifier CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _DescriptionType (GovernmentSurvey | LandGrant | MetesAndBounds |
NativeAmericanLandIdentifier | RancherolIdentifier | TownshipIdentifier | Other) #IMPLIED
  _DescriptionTypeOtherDescription CDATA #IMPLIED
  PropertyRangeIdentifier CDATA #IMPLIED
  PropertySectionIdentifier CDATA #IMPLIED
  PropertyTownshipIdentifier CDATA #IMPLIED-->
```

UNPLATTED_LAND Examples

Please see the following example listed below.

Beginning at the NW corner of the SE corner of section 22, township 23E, range 3N proceed east 350'. Turn to south 180° - proceed south 600'. Turn to west 270° - proceed west 350'. Turn to north 0° - proceed north to point of beginning.

```
<UNPLATTED_LAND _QuarterSectionIdentifier="NW,SE" PropertySectionIdentifier="22"
PropertyTownshipIdentifier="23E" PropertyRangeIdentifier="03N"
_MetesAndBoundsRemainingDescription="Beginning at the NW corner of the SE corner of
section 22, township 23E, range 3N proceed east 350'. Turn to south 180° - proceed south
600'. Turn to west 270° - proceed west 350'. Turn to north 0° - proceed north to point of
beginning." />
```

Child Elements

None

UNPLATTED_LAND Attributes

[_AbstractNumberIdentifier] Attribute

An Identifier, usually a number, for an abstract document which describes a parcel of land.

[_BaselIdentifier] Attribute

The Government Survey Base coordinate of the property that is the east-west line from which to measure Township lines (six [6] miles apart) north or south of the base line. For example: Township 2 North, Range 2 West measured from a Base and Meridian point.

[_LandGrantIdentifier] Attribute

Identifier of Land Grants issued by foreign governments to then citizens of the country during that countrys ownership or control of a territory now part of the United States, e.g. Gomez Grant.

[_MeridianIdentifier] Attribute

The Government survey Meridian of the property is the north-south line from which to measure Range lines east and west of the Meridian, or starting point.

[_MetesAndBoundsRemainingDescription] Attribute

The text of the remaining portion of a Metes and Bounds description after indexing information is populated into the other data fields.

For example: Commencing 96.8 rods North and 155 rods West from the Southeast corner of Section 11, Township 1 South, Range 1 West, Salt Lake Meridian, West 105.495 feet; thence South 98 feet; thence West 60 feet; thence South 34 feet; thence East 10.03 rods; thence North 8 rods to the point of beginning.

There are two other terms that often appear in metes and bounds descriptions: a chain, which is approximately 66 feet, and a rod which is approximately one-quarter of a chain or 16.5 feet.

[_QuarterSectionIdentifier] Attribute

The quarter section(s) in which the property is located based on Government Survey system. Up to four quarter or half sections may be used to describe or locate the property (e.g. NE of the SE).

- Note: It is recommended to include all the quarter section calls in one string.
 - Example: "The northeast quarter of the southeast quarter of section 31..."

```
<UNPLATTED_LAND>
  <_QuarterSectionIdentifier = NE,SE/>
  <PropertySectionIdentifier = 31/>
/<UNPLATTED_LAND>
```

[_SequencelIdentifier] Attribute

If more than one unplatted land legal description is used this indicates which order they should appear on the document.

[_DescriptionType] Attribute

The type of land description system used for unplatted lands, either Government Survey, or metes and bounds, or other (e.g. land grants).

[_DescriptionTypeOtherDescription] Attribute

A free-form text field used to describe the type of land description system used for unplatted lands if Other is selected as the Unplatted Land Description Type.

[PropertyRangeldIdentifier] Attribute

A piece of land measuring 6 miles by 6 miles. It's measured East and West in columns. Commonly used in a grid along with Township and Section to describe an area of land at the county level.

[PropertySectionIdentifier] Attribute

Legal Description Section number of property being financed. Numbered 1 through 36.

[PropertyTownshipIdentifier] Attribute

A piece of land measuring 6 miles by 6 miles. It is measured North and South in rows. Commonly used in a grid along with Range and Section to describe an area of land at the county level.

UNPLATTED_LAND Deprecations

None

[PARTIES] Element

The PARTIES element contains information about the return-to, the prepared-by, the bill-to, and the taxable parties.

PARTIES DTD Definition

```
<!ELEMENT PARTIES (_RETURN_TO_PARTY+, _PREPARED_BY_PARTY*, TAXABLE_PARTY*,  
BILL_TO_PARTY*, WITNESS*)>  
<!ATTLIST PARTIES  
  _ID ID #IMPLIED>
```

PARTIES Examples

Please see the following example listed below.

None

Child Elements:

[_RETURN_TO_PARTY]

[_PREPARED_BY_PARTY]

[TAXABLE_PARTY]

[BILL_TO_PARTY]

[WITNESS]

PARTIES Attributes

None

PARTIES Deprecations

None

[_RETURN_TO_PARTY] Element

The _RETURN_TO_PARTY element contains information regarding where the county recorder is to route the document once recording is completed.

_RETURN_TO_PARTY DTD Definition

```
<!ELEMENT _RETURN_TO_PARTY (PREFERRED_RESPONSE*,
NON_PERSON_ENTITY_DETAIL, CONTACT_DETAIL)>
<!ATTLIST _RETURN_TO_PARTY
  _ID ID #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _StreetAddress CDATA #IMPLIED
  _StreetAddress2 CDATA #IMPLIED
  _City CDATA #IMPLIED
  _State CDATA #IMPLIED
  _PostalCode CDATA #IMPLIED
  _County CDATA #IMPLIED
  _Country CDATA #IMPLIED
  _CountryCode CDATA #IMPLIED
  NonPersonEntityIndicator (Y | N) #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _TitleDescription CDATA #IMPLIED>
```

_RETURN_TO_PARTY Examples

Please see the following example listed below.

None

Child Elements:

[PREFERRED_RESPONSE]

[NON_PERSON_ENTITY_DETAIL]

[CONTACT_DETAIL]

_RETURN_TO_PARTY Attributes

[_UnparsedName] Attribute

The unparsed name of the party to whom the recorded document is to be returned.

[_StreetAddress] Attribute

The street address to which the recorded document should be returned.

[_StreetAddress2] Attribute

The second line of the street address to which the recorded document should be returned.

[_City] Attribute

The city in which the address to which the recorded document should be returned is located.

[_State] Attribute

The state in which the address to which the recorded document should be returned is located.

[_PostalCode] Attribute

The postal code (zip code in US) of the address to which the recorded document should be returned is located. Zip code may be either 5 or 9 digits.

[_County] Attribute

The county in which the address to which the recorded document should be returned is located.

[_Country] Attribute

The country in which the address to which the recorded document should be returned is located.

[_CountryCode] Attribute

The country code in which the address to which the recorded document should be returned is located.

[_NonPersonEntityIndicator] Attribute

When true, indicates that the party is a non person entity.

[_SequenceIdentifier] Attribute

When multiple Return To Parties are included this indicates the order in which they should appear on the document

[_TitleDescription] Attribute

The title of the party to whom the recorded document is to be returned.

RETURN TO PARTY Deprecations

None

[PREFERRED_RESPONSE] Element

The PREFERRED_RESPONSE element contains information regarding how the party desires to receive responses to its requests.

In the PRIA_DOCUMENT DTD it is used with _RETURN_TO_PARTY, _PREPARED_BY_PARTY, TAXABLE_PARTY and BILL_TO_PARTY.

PREFERRED_RESPONSE DTD Definition

```
<!ELEMENT PREFERRED_RESPONSE EMPTY>
<!ATTLIST PREFERRED_RESPONSE
  _ID ID #IMPLIED
  _Format (Other | PCL | PDF | Text | TIFF | XML) #IMPLIED
  _Method (Fax | File | FTP | HTTP | HTTPS | Mail | MessageQueue | Other | SMTP | VAN)
  #IMPLIED
  _Destination CDATA #IMPLIED
  _FormatOtherDescription CDATA #IMPLIED
  _MethodOther CDATA #IMPLIED
  _UseEmbeddedFileIndicator (Y | N) #IMPLIED
  MIMETYPE CDATA #IMPLIED
  _VersionIdentifier CDATA #IMPLIED>
```

PREFERRED_RESPONSE Examples

Please see the following example listed below.

None

Child Elements:

None

PREFERRED_RESPONSE Attributes

[_Format] Attribute

Designate the response format type for the service or product being requested.

The format options are Text, PCL, PDF, XML or Other. If Other is selected, enter the format in Response Format Other.

[_Method] Attribute

Specifies the method for delivering the credit report - Email, Fax, File, FTP, HTTP, etc.

The destination information is listed in the Response Destination element.

[Destination] Attribute

A text description of the destination to which the response should be delivered.

[_FormatOtherDescription] Attribute

Describes the Other format selected in the Response Format element.

[_MethodOther] Attribute

Specify a delivery method not listed in the Response Method element.

[_UseEmbeddedFileIndicator] Attribute

Set to Y if the specified Preferred Response is to be contained in the Embedded File element.

Set to N if it is to be provided as a separate file.

[MIMETYPE] Attribute

Indicates the Multipurpose Internet Mail Extensions type of the data in the DOCUMENT container.

A registered list of these types is available at <http://www.isi.edu/in-notes/iana/assignments/media-types/media-types>.

[_VersionIdentifier] Attribute

Designates the Version of the Preferred Response Format that the requester expects to receive in the response (i.e. MISMO 2.1 to indicate that the requester expects to receive a MISMO version 2.1 response file).

PREFERRED_RESPONSE Deprecations

None

[NON_PERSON_ENTITY_DETAIL] Element

The NON_PERSON_ENTITY_DETAIL element contains information regarding non-human parties such as businesses, LLC's, trusts, etc.

In the PRIA_DOCUMENT DTD it is used with _RETURN_TO_PARTY, _PREPARED_BY_PARTY, TAXABLE_PARTY and BILL_TO_PARTY.

NON_PERSON_ENTITY_DETAIL DTD Definition

```
<!ELEMENT NON_PERSON_ENTITY_DETAIL (AUTHORIZED_REPRESENTATIVE*)>
<!ATTLIST NON_PERSON_ENTITY_DETAIL
  _ID ID #IMPLIED
  _OrganizationType (Corporation | JointVenture | LimitedLiabilityCompany | LimitedPartnership |
  NonProfitCorporation | Other | Partnership | SoleProprietorship | Trust) #IMPLIED
  _OrganizationTypeOtherDescription CDATA #IMPLIED
  _OrganizedUnderTheLawsOfJurisdictionName CDATA #IMPLIED
  _SuccessorClauseTextDescription CDATA #IMPLIED
  _TaxIdentificationNumberIdentifier CDATA #IMPLIED>
```

NON_PERSON_ENTITY_DETAIL Examples

Please see the following example listed below.

Child Elements:

[\[AUTHORIZED_REPRESENTATIVE\]](#)

NON_PERSON_ENTITY_DETAIL Attributes

[_OrganizationType] Attribute

The description of the entity type of the party or organization.

[_OrganizationTypeOtherDescription] Attribute

The description of the Organization Type when Other is selected as the option from the enumerated list.

[_OrganizedUnderTheLawsOfJurisdictionName] Attribute

The name and type of jurisdiction under which the party as an entity is organized and under whose authority the party as an entity operates.

[_SuccessorClauseTextDescription] Attribute

The description used as the Successor Clause for the non person entity, i. e. Its successors and/or assigns.

[_TaxIdentificationNumberIdentifier] Attribute

The Tax Identification Number assigned to the non person entity.

- Examples include: FEIN number or Social Security number

NON_PERSON_ENTITY_DETAIL Deprecations

None

[AUTHORIZED_REPRESENTATIVE] Element

The AUTHORIZED_REPRESENTATIVE element contains information regarding the individual empowered to act on behalf of a non-person entity.

AUTHORIZED_REPRESENTATIVE DTD Definition

```
<!ELEMENT AUTHORIZED_REPRESENTATIVE (CONTACT_DETAIL?)>
<!ATTLIST AUTHORIZED_REPRESENTATIVE
  _ID ID #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _TitleDescription CDATA #IMPLIED
  AuthorizedToSignIndicator (Y | N) #IMPLIED>
```

AUTHORIZED_REPRESENTATIVE Examples

Please see the following example listed below.

None

Child Elements:

[\[CONTACT_DETAIL\]](#)

AUTHORIZED_REPRESENTATIVE Attributes

[_UnparsedName] Attribute

The name of the person or entity designated as the authorized representative for the party.

[_TitleDescription] Attribute

The title of the authorized representative.

[AuthorizedToSignIndicator] Attribute

This indicator is set to be true when the authorized representative can sign for the party.

AUTHORIZED_REPRESENTATIVE Deprecations

None

[CONTACT_DETAIL] Element

The CONTACT_DETAIL element contains the name of the individual acting as a contact for a party.

In the PRIA_DOCUMENT DTD it is used with _RETURN_TO_PARTY, _PREPARED_BY_PARTY, TAXABLE_PARTY and BILL_TO_PARTY.

CONTACT_DETAIL DTD Definition

```
<!ELEMENT CONTACT_DETAIL (CONTACT_POINT*)>
<!ATTLIST CONTACT_DETAIL
    _ID ID #IMPLIED
    _Name CDATA #IMPLIED>
```

CONTACT_DETAIL Examples

Please see the following example listed below.

None

Child Elements:

[CONTACT_POINT]

CONTACT_DETAIL Attributes

[_Name] Attribute

The name of an individual acting as a contact for a party to the loan.

CONTACT_DETAIL Deprecations

None

[CONTACT_POINT] Element

The CONTACT_POINT element contains information regarding how to contact the individual named in CONTACT_DETAIL.

CONTACT_POINT DTD Definition

```
<!ELEMENT CONTACT_POINT EMPTY>
<!ATTLIST CONTACT_POINT
  _ID ID #IMPLIED
  _PreferenceIndicator (Y | N) #IMPLIED
  _RoleType (Home | Mobile | Other | Work) #IMPLIED
  _Type (Email | Fax | Other | Phone) #IMPLIED
  _TypeOtherDescription CDATA #IMPLIED
  _Value CDATA #IMPLIED>
```

CONTACT_POINT Examples

Please see the following example listed below.

None

Child Elements:

None

CONTACT_POINT Attributes

[_PreferenceIndicator] Attribute

This indicator is set to be true when the particular contacts Contact Point (i.e. phone number) is preferred over any others.

[_RoleType] Attribute

This element sets the type of role (i.e. Home, Work or Mobile) used for the Contact Point Type (Phone, Fax, Email).

[_Type] Attribute

This datum sets the Type (Phone, Fax, E-Mail, Other) used for the Contact Point Value.

[_TypeOtherDescription] Attribute

A free-form text field used to describe the Contact Point if Other is selected as the Contact Point Type.

[_Value] Attribute

This is the actual value (Phone, Fax, E-Mail, Other) of the Contact Point Type.

CONTACT_POINT DTD Deprecations

None

[_PREPARED_BY_PARTY] Element

The `_PREPARED_BY_PARTY` element contains the name of the individual who drafted the document.

PREPARED_BY_PARTY Definition

```
<!ELEMENT _PREPARED_BY_PARTY (PREFERRED_RESPONSE*,
NON_PERSON_ENTITY_DETAIL, CONTACT_DETAIL)>
<!ATTLIST _PREPARED_BY_PARTY
  _ID ID #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _StreetAddress CDATA #IMPLIED
  _StreetAddress2 CDATA #IMPLIED
  _City CDATA #IMPLIED
  _State CDATA #IMPLIED
  _PostalCode CDATA #IMPLIED
  _County CDATA #IMPLIED
  _Country CDATA #IMPLIED
  _CountryCode CDATA #IMPLIED
  NonPersonEntityIndicator (Y | N) #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _TitleDescription CDATA #IMPLIED>
```

PREPARED_BY_PARTY Examples

Please see the following example listed below.

None

Child Elements:

[\[PREFERRED_RESPONSE\]](#)

[\[NON_PERSON_ENTITY_DETAIL\]](#)

[\[CONTACT_DETAIL\]](#)

PREPARED_BY_PARTY Attributes

[_UnparsedName] Attribute

The unparsed name of the individual who is the preparer of the recordable document.

[_StreetAddress] Attribute

The street address of the preparer of the recordable document.

[_StreetAddress2] Attribute

The second line of the street address of the preparer of the recordable document (if needed).

[_City] Attribute

The city in which the address of the preparer of the recordable document is located.

[_State] Attribute

The state in which the address of the preparer of the recordable document is located.

[_PostalCode] Attribute

The postal code (zip code in US) of the address of the preparer of the recorded document. Zip code may be either 5 or 9 digits.

[_County] Attribute

The county in which the address of the preparer of the recordable document is located.

[_Country] Attribute

The country in which the address of the preparer of the recordable document is located.

[_CountryCode] Attribute

The country code in which the address of the preparer of the recordable document is located.

[NonPersonEntityIndicator] Attribute

When true, indicates that the party is a non person entity.

[_SequenceIdentifier] Attribute

When more than one name appears, this indicates the order in which they should appear on the document.

[_TitleDescription] Attribute

The title of the party designated as the preparer of the recordable document.

PREPARED_BY_PARTY Deprecations

None

[TAXABLE_PARTY] Element

The TAXABLE_PARTY element contains information regarding the party to whom the property tax bill will be sent to.

TAXABLE_PARTY DTD Definition

```
<!ELEMENT TAXABLE_PARTY (PREFERRED_RESPONSE*, NON_PERSON_ENTITY_DETAIL,
CONTACT_DETAIL)>
<!ATTLIST TAXABLE_PARTY
  _ID ID #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _StreetAddress CDATA #IMPLIED
  _StreetAddress2 CDATA #IMPLIED
  _City CDATA #IMPLIED
  _State CDATA #IMPLIED
  _PostalCode CDATA #IMPLIED
  _County CDATA #IMPLIED
  _Country CDATA #IMPLIED
  _CountryCode CDATA #IMPLIED
  NonPersonEntityIndicator (Y | N) #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _TitleDescription CDATA #IMPLIED>
```

TAXABLE_PARTY Examples

Please see the following example listed below.

None

Child Elements:

[PREFERRED_RESPONSE]

[NON_PERSON_ENTITY_DETAIL]

[CONTACT_DETAIL]

TAXABLE_PARTY Attributes

[_UnparsedName] Attribute

The unparsed name of the party to whom the tax bill is to be sent.

[_StreetAddress] Attribute

The street address to which the tax bill should be sent.

[_StreetAddress2] Attribute

The second line of the street address to which the tax bill should be sent.

[_City] Attribute

The city in which the address to which the tax bill is to be sent is located.

[_State] Attribute

The state in which the address to which the tax bill is to be sent is located.

[_PostalCode] Attribute

The postal code (zip code in US) of the address to which the tax bill is to be sent is located. Zip code may be either 5 or 9 digits.

[_County] Attribute

The county in which the address to which the tax bill is to be sent is located.

[_Country] Attribute

The country in which the address to which the tax bill is to be sent is located.

[_CountryCode] Attribute

The country code in which the address to which the tax bill is to be sent is located.

[_NonPersonEntityIndicator] Attribute

When true, indicates that the party is a non person entity.

[_SequencelIdentifier] Attribute

When multiple Taxable_Parties are included this indicates the order in which they should appear on the document

[_TitleDescription] Attribute

The title of the party to whom the tax bill is to be sent.

TAXABLE_PARTY Deprecations

None

[BILL_TO_PARTY] Element

The BILL_TO_PARTY Element contains information regarding the party to whom a recording transaction is to be invoiced to after the recording is completed. This is usually restricted to state and federal agencies.

BILL_TO_PARTY DTD Definition

```
<!ELEMENT BILL_TO_PARTY (PREFERRED_RESPONSE*, NON_PERSON_ENTITY_DETAIL,
CONTACT_DETAIL)>
<!ATTLIST BILL_TO_PARTY
  _ID ID #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _StreetAddress CDATA #IMPLIED
  _StreetAddress2 CDATA #IMPLIED
  _City CDATA #IMPLIED
  _State CDATA #IMPLIED
  _PostalCode CDATA #IMPLIED
  _County CDATA #IMPLIED
  _Country CDATA #IMPLIED
  _CountryCode CDATA #IMPLIED
  NonPersonEntityIndicator (Y | N) #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _TitleDescription CDATA #IMPLIED>
```

BILL_TO_PARTY Examples

Please see the following example listed below.

None

Child Elements:

[\[PREFERRED_RESPONSE\]](#)

[\[NON_PERSON_ENTITY_DETAIL\]](#)

[\[CONTACT_DETAIL\]](#)

BILL_TO_PARTY Attributes

[_UnparsedName] Attribute

The unparsed name of the party to whom fees are billed

[_StreetAddress] Attribute

The street address of the party to whom fees are billed.

[_StreetAddress2] Attribute

The second line of the street address of the party to whom fees are billed is located.

[_City] Attribute

The city in which the address of the party to whom fees are billed is located.

[_State] Attribute

The state in which the address of the party to whom fees are billed is located.

[_PostalCode] Attribute

The postal code (zip code in US) of the address of the party to whom fees are billed is located. Zip code may be either 5 or 9 digits.

[_County] Attribute

The county in which the address of the party to whom fees are billed is located.

[_Country] Attribute

The country in which the address of the party to whom fees are billed is located.

[_CountryCode] Attribute

The country code in which the address of the party to whom fees are billed is located.

[_NonPersonEntityIndicator] Attribute

When true, indicates that the party is a non person entity.

[_SequencelIdentifier] Attribute

When multiple Bill_To_Parties are included this indicates the order in which they should appear on the document

[_TitleDescription] Attribute

The title of the party to whom fees are billed.

BILL_TO_PARTY Deprecations

None

[WITNESS] Element

The WITNESS element contains the names of the witness(es) to the signature(s) applied to the document.

WITNESS DTD Definition

```
<!ELEMENT WITNESS EMPTY>
<!ATTLIST WITNESS
  _ID ID #IMPLIED
  _UnparsedName CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED>
```

WITNESS Examples

Please see the following example listed below.

None

Child Elements:

None

WITNESS Attributes

[_UnparsedName] Attribute

The unparsed name of the witness to the signing, by one or more borrowers, of the recordable document.

[_SequenceIdentifier] Attribute

An identifier that designates the order or place assigned to a specific witness in relation to other witnesses. Used to assign a specific position to a witness within a sequence of witnesses.

WITNESS Deprecations

None

[EXECUTION] Element

The EXECUTION element contains information regarding where the document was signed.

EXECUTION DTD Definition

```
<!ELEMENT EXECUTION EMPTY>
<!ATTLIST EXECUTION
  _ID ID #IMPLIED
  _Date CDATA #IMPLIED
  _City CDATA #IMPLIED
  _County CDATA #IMPLIED
  _State CDATA #IMPLIED>
```

EXECUTION Examples

Please see the following example listed below.

None

Child Elements:

None

EXECUTION Attributes

[_Date] Attribute

The date documents were signed (executed).

[_City] Attribute

The name of the city where documents were signed (executed).

[_County] Attribute

The name of the county where documents were signed (executed).

[_State] Attribute

The name of the state where documents were signed (executed).

EXECUTION Deprecations

None

[MORTGAGE_CONSIDERATION] Element

The MORTGAGE_CONSIDERATION element contains information regarding the amount of the mortgage.

MORTGAGE_CONSIDERATION DTD Definition

```
<!ELEMENT MORTGAGE_CONSIDERATION EMPTY>
<!ATTLIST MORTGAGE_CONSIDERATION
  _ID ID #IMPLIED
  HELOCInitialAdvanceAmount CDATA #IMPLIED
  OriginalLoanAmount CDATA #IMPLIED>
```

MORTGAGE_CONSIDERATION Examples

Please see the following example listed below.

None

Child Elements:

None

MORTGAGE_CONSIDERATION Attributes

[HELOCInitialAdvanceAmount] Attribute

The amount of money that a borrower receives at closing in a HELOC transaction.

[_OriginalLoanAmount] Attribute

Amount of the Mortgage as stated on the original note.

MORTGAGE_CONSIDERATION Deprecations

None

[CONSIDERATION] Element

The CONSIDERATION element contains information regarding what a property sold for.

CONSIDERATION DTD Definition

```
<!ELEMENT CONSIDERATION EMPTY>
<!ATTLIST CONSIDERATION
  _ID ID #IMPLIED
  _Amount CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _Type (SalePrice | Judgment | Lien | AttorneysFee | Other) #IMPLIED
  _TypeOtherDescription CDATA #IMPLIED>
```

CONSIDERATION Examples

Please see the following example listed below.

None

Child Elements:

None

CONSIDERATION Attributes

[_Amount] Attribute

Amount of the consideration.

[_SequenceIdentifier] Attribute

When multiple forms of consideration are included, this indicates the order in which they should be listed

[_Type] Attribute

Description of the consideration or money amount of the document. (Not necessarily the mortgage amount)

[_TypeOtherDescription] Attribute

A description of a CONSIDERATION type not included in the enumerated list

CONSIDERATION Deprecations

None

[RECORDABLE_DOCUMENT] Element

The RECORDABLE_DOCUMENT element contains information about previously recorded documents that are referenced in a new document.

RECORDABLE_DOCUMENT DTD Definition

```
<!ELEMENT RECORDABLE_DOCUMENT (_ASSOCIATED_DOCUMENT*)>  
<!ATTLIST RECORDABLE_DOCUMENT  
  _ID ID #IMPLIED>
```

RECORDABLE_DOCUMENT Examples

Please see the following example listed below.

None

Child Elements:

[\[ASSOCIATED_DOCUMENT\]](#)

RECORDABLE_DOCUMENT Attributes

None

RECORDABLE_DOCUMENT Deprecations

None

[_ASSOCIATED_DOCUMENT] Element

The _ASSOCIATED_DOCUMENT Element contains information regarding a document that relates to or is dependent upon another recorded document.

_ASSOCIATED_DOCUMENT DTD Definition

```
<!ELEMENT _ASSOCIATED_DOCUMENT EMPTY>
<!-- ATTLIST _ASSOCIATED_DOCUMENT
  _ID ID #IMPLIED
  _BookNumber CDATA #IMPLIED
  _BookType (Plat | Deed | Mortgage | Maps | Other) #IMPLIED
  _BookTypeOtherDescription CDATA #IMPLIED
  _Code CDATA #IMPLIED
  _CountyOfRecordationName CDATA #IMPLIED
  _Title CDATA #IMPLIED
  _InstrumentNumber CDATA #IMPLIED
  _Number CDATA #IMPLIED
  _OfficeOfRecordationName CDATA #IMPLIED
  _PageNumber CDATA #IMPLIED
  _RecordingDate CDATA #IMPLIED
  _RecordingJurisdictionName CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _StateOfRecordationName CDATA #IMPLIED
  _Type (AbstractofJudgment | AffidavitofDeath | Assignment | AssignmentofDeedofTrust |
AssignmentOfMortgage | BargainAndSaleDeed | BlanketAssignment | Deed | DeedOfTrust |
FederalTaxLien | Judgment | ModificationAgreementOrConsolidationAgreements | Mortgage |
Other | PartialSatisfactionOfLien | PowerofAttorney | QuitClaimDeed | Reconveyance |
ReleaseofFederalTaxLien | ReleaseOfLien | ReleaseofStateTaxLien | SatisfactionOfLien |
SatisfactionofMortgage | StateTaxLien | SecurityInstrument | SignatureAffidavit |
SubordinateLienAgreement | SubstitutionofTrustee | TreasurersTaxLien | WarrantyDeed)
#IMPLIED
  _TypeOtherDescription CDATA #IMPLIED
  _VolumeNumber CDATA #IMPLIED-->
```

_ASSOCIATED_DOCUMENT Examples

Please see the following example listed below.

None

Child Elements:

None

_ASSOCIATED_DOCUMENT Attributes

[_BookNumber] Attribute

The book number identifier of the associated document referenced in the subject recordable document.

[_BookType] Attribute

The book type of the associated document referenced in the subject recordable document.

[_BookTypeOtherDescription] Attribute

The description of the book type of the associated document referenced in the subject recordable document when Other is selected as the enumerated value.

[_Code] Attribute

Document Code of the document referenced in the subject document

[_CountyOfRecordationName] Attribute

The county of recordation name of the associated document referenced in the subject recordable document.

[_Title] Attribute

The document title of the associated document referenced in the subject recordable document.

[_InstrumentNumber] Attribute

The instrument number identifier of the associated document referenced in the subject recordable document.

[_Number] Attribute

The document number identifier of the associated document referenced in the subject recordable document.

[_OfficeOfRecordationName] Attribute

The Office of Recordation of the associated document referenced in the subject recordable document.

[_PageNumber] Attribute

The page number identifier of the associated document referenced in the subject recordable document.

[_RecordingDate] Attribute

The recordation date of the associated document referenced in the subject recordable document.

[_RecordingJurisdictionName] Attribute

The recording jurisdiction name of the associated document referenced in the subject recordable document.

[_SequencIdentifier] Attribute

When multiple associated documents are referenced, this indicates the order in which they should appear in the document

[_StateOfRecordationName] Attribute

The state of recordation of the associated document referenced in the subject recordable document.

[_Type] Attribute

The document type of the associated document referenced in the subject recordable document.

[_TypeOtherDescription] Attribute

The description of the document type of the associated document referenced in the subject recordable document when Other is selected as the enumerated value.

[_VolumeNumber] Attribute

The volume number identifier of the associated document referenced in the subject recordable document.

ASSOCIATED_DOCUMENT Deprecations

None

[SIGNATORY] Element

The SIGNATORY element contains information regarding the person(s) signing a document.

In the PRIA_DOCUMENT DTD it is used here, for the document signers and also as a child element of NOTARY and RECORDING ENDORSEMENT.

SIGNATORY DTD Definition

```
<!ELEMENT SIGNATORY (ELECTRONIC_SIGNATURE)>
<!ATTLIST SIGNATORY
  ID ID #IMPLIED
  FirstName CDATA #IMPLIED
  MiddleName CDATA #IMPLIED
  LastName CDATA #IMPLIED
  NameSuffix CDATA #IMPLIED
  UnparsedName CDATA #IMPLIED
  SigningCapacityDescription CDATA #IMPLIED
  OrganizationName CDATA #IMPLIED
  OrganizationPositionDescription CDATA #IMPLIED
  SignatureDate CDATA #IMPLIED
  SequenceIdentifier CDATA #IMPLIED
  SignatureTimeStamp CDATA #IMPLIED>
```

SIGNATORY Examples

Please see the following example listed below.

None

Child Elements:

[ELECTRONIC_SIGNATURE]

SIGNATORY Attributes

[FirstName] Attribute

First Name of Party signing the document.

[MiddleName] Attribute

Middle Name (or initial) of Party signing the document.

[LastName] Attribute

Last Name of Party signing the document.

[NameSuffix] Attribute

The suffix or lineage of the Party signing the document (ie: Jr, Sr, III)

[UnparsedName] Attribute

Unparsed name of Party signing the document.

[SigningCapacityDescription] Attribute

Capacity of Individual signing the document. (i.e: executor, attorney-in-fact, etc.)

[OrganizationName] Attribute

Name of Organization or Corporation signing the document (other than party)

[OrganizationPositionDescription] Attribute

Title of Person or Agent signing the document (Corporate Officer) on behalf of Organization or Corporation

[SignatureDate] Attribute

Date Signature affixed to Document.

[_SequenceIdentifier] Attribute

When multiple signatures are applied, this indicates the order in which they should appear in the document

[SignatureTimeStamp] Attribute

When the signature is produced by some system the time stamp instant in time when the signature is produced.

SIGNATORY Deprecations

None

[ELECTRONIC_SIGNATURE] Element

The ELECTRONIC_SIGNATURE element contains the structure necessary to hold whatever type of electronic signature is used to sign a document.

ELECTRONIC_SIGNATURE DTD Definition

```
<!ELEMENT ELECTRONIC_SIGNATURE (TEXT_SIGNATURE?, EMBEDDED_SIGNATURE_FILE?,  
EXTERNAL_SIGNATURE_FILE?, Signature*, OTHER_SIGNATURE*)>  
<!ATTLIST ELECTRONIC_SIGNATURE  
  ID ID #IMPLIED  
  ElectronicSignatureType (Digital | EncodedEmbedded | EncodedExternal | Other) #IMPLIED  
  ElectronicSignatureTypeOtherDescription CDATA #IMPLIED>
```

ELECTRONIC_SIGNATURE Examples

Please see the following example listed below.

None

Child Elements:

[TEXT_SIGNATURE]

[EMBEDED_SIGNATURE_FILE]

[EXTERNAL_SIGNATURE_FILE]

[Signature]

[OTHER_SIGNATURE]

ELECTRONIC_SIGNATURE Attributes

[ElectronicSignatureType] Attribute

The type of electronic signature being used.

[ElectronicSignatureTypeOtherDescription] Attribute

The description of type of electronic signature when the Electronic Signature enumerated value selected is Other.

ELECTRONIC_SIGNATURE Deprecations

None

[TEXT_SIGNATURE] Element

The TEXT_SIGNATURE element contains a typed name used as a signature with the intent to sign the document.

TEXT_SIGNATURE DTD Definition

```
<!ELEMENT TEXT_SIGNATURE EMPTY>  
<!ATTLIST TEXT_SIGNATURE  
  ID ID #IMPLIED>
```

TEXT_SIGNATURE Examples

Please see the following example listed below.

None

Child Elements:

None

TEXT_SIGNATURE Attributes

None

TEXT_SIGNATURE Deprecations

None

[EMBEDDED_SIGNATURE_FILE] Element

The EMBEDDED_SIGNATURE_FILE element contains information about an embedded file that is being used as a signature with the intent to sign the document.

EMBEDDED_SIGNATURE_FILE DTD Definition

```
<!ELEMENT EMBEDDED_SIGNATURE_FILE (DOCUMENT?)>
<!ATTLIST EMBEDDED_SIGNATURE_FILE
  ID CDATA #IMPLIED
  EmbeddedFileType CDATA #IMPLIED
  EmbeddedFileVersion CDATA #IMPLIED
  EmbeddedFileName CDATA #IMPLIED
  FileEncodingType CDATA #IMPLIED
  FileDescription CDATA #IMPLIED
  MIMETYPE CDATA #IMPLIED
  PageCount CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED>
```

EMBEDDED_SIGNATURE_FILE Examples

Please see the following example listed below.

None

Child Elements:

[DOCUMENT]

EMBEDDED_SIGNATURE_FILE Attributes

[EmbeddedFileType] Attribute

Type of file embedded in signature

[EmbeddedFileVersion] Attribute

The version number for the embedded file

[EmbeddedFileName] Attribute

The name of the embedded file

[FileEncodingType] Attribute

Specifies the type of encoding used on the embedded file (i.e. Base64).

Transfer Encoding names are registered by the Internet Assigned Numbers Authority, and the currently registered names can be viewed at: <http://www.iana.org/assignments/transfer-encodings><http://www.iana.org/assignments/media-types/index.html> which may be useful to standardize the values in this attribute.

[FileDescription] Attribute

Provides additional information about the embedded file.

[MIMEType] Attribute

Indicates the Multipurpose Internet Mail Extensions type of the data in the DOCUMENT container. A registered list of these types is available at <http://www.isi.edu/in-notes/iana/assignments/media-types/media-types>.

[PageCount] Attribute

The number of pages in the embedded file.

[_SequenceIdentifier] Attribute

When more than one embedded signature file is included, this indicates the order in which they should appear.

[DOCUMENT] Element

The DOCUMENT element contains the actual file defined in its parent element.

In the PRIA_DOCUMENT DTD it is used with EMBEDDED_SIGNATURE_FILE and EMBEDDED_FILE.

DOCUMENT ELEMENT DTD Definition

<ELEMENT DOCUMENT (ANY)>

DOCUMENT ELEMENT Examples

Please see the following example listed below.

None

Child Elements:

None

DOCUMENT ELEMENT DTD Attributes

None

DOCUMENT ELEMENT Deprecations

None

[EXTERNAL_SIGNATURE_FILE] Element

The EXTERNAL_SIGNATURE_FILE element contains information regarding an electronic signature that is located outside the electronic document itself.

EXTERNAL_SIGNATURE_FILE DTD Definition

```
<!ELEMENT EXTERNAL_SIGNATURE_FILE EMPTY>
<!ATTLIST EXTERNAL_SIGNATURE_FILE
  ID CDATA #IMPLIED
  FileEncodingType CDATA #IMPLIED
  FileDescription CDATA #IMPLIED
  MIMETYPE CDATA #IMPLIED
  URI CDATA #IMPLIED
  PageCount CDATA #IMPLIED
  SignatureSequencelIdentifier CDATA #IMPLIED>
```

EXTERNAL_SIGNATURE_FILE Examples

Please see the following example listed below.

None

Child Elements:

None

EXTERNAL_SIGNATURE_FILE Attributes

[FileEncodingType] Attribute

Specifies the type of encoding used on the external file (i.e. Base64).

Transfer Encoding names are registered by the Internet Assigned Numbers Authority, and the currently registered names can be viewed at: <http://www.iana.org/assignments/transfer-encodings><http://www.iana.org/assignments/media-types/index.html> which may be useful to standardize the values in this attribute.

[FileDescription] Attribute

Provides additional information about the external file.

[MIMETYPE] Attribute

Indicates the Multipurpose Internet Mail Extensions type of the data in the ELECTRONIC_SIGNATURE container. A registered list of these types is available at <http://www.isi.edu/in-notes/iana/assignments/media-types/media-types>.

[URI] Attribute

Specifies the location of the external file.

[PageCount] Attribute

The number of pages in the external file.

[SignatureSequencelIdentifier] Attribute

When more than one external signature file is included, this indicates the order in which they should appear.

EXTERNAL_SIGNATURE_FILE Deprecations

None

[OTHER_SIGNATURE] Element

The OTHER_SIGNATURE element contains the electronic signature used when the ElectronicSignatureType selected is “Other”.

OTHER_SIGNATURE DTD Definition

```
<!ELEMENT OTHER_SIGNATURE ANY>  
<!ATTLIST OTHER_SIGNATURE  
  ID CDATA #IMPLIED>
```

OTHER_SIGNATURE Examples

Please see the following example listed below.

None

Child Elements:

None

OTHER_SIGNATURE Attributes

None

OTHER_SIGNATURE Deprecations

None

[NOTARY] Element

The NOTARY element contains identifying information for the notary public.

NOTARY DTD Definition

```
<!ELEMENT NOTARY (_CERTIFICATE*, SIGNATORY+)>
<!ATTLIST NOTARY
  _ID ID #IMPLIED
  _CommissionBondNumberIdentifier CDATA #IMPLIED
  _CommissionNumberIdentifier CDATA #IMPLIED
  _CommissionExpirationDate CDATA #IMPLIED
  _CommissionState CDATA #IMPLIED
  _CommissionCounty CDATA #IMPLIED
  _CommissionCity CDATA #IMPLIED
  _PRIAVersionIdentifier CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED
  _TitleDescription CDATA #IMPLIED
  _UnparsedName CDATA #IMPLIED>
```

NOTARY Examples

Please see the following example listed below.

None

Child Elements:

[\[_CERTIFICATE\]](#)

[\[SIGNATORY\]](#)

NOTARY Attributes

[_CommissionBondNumberIdentifier] Attribute

The identifying commission bond number of the notary.

[_CommissionNumberIdentifier] Attribute

The identifying commission number assigned to the notary.

[_CommissionExpirationDate] Attribute

The expiration date of the notary's commission.

[_CommissionState] Attribute

The state in which notary is commissioned.

[_CommissionCounty] Attribute

The county in which the notary is commissioned.

[_CommissionCity] Attribute

The city where the notary is commissioned.

[_PRIAVersion] Attribute

The version of the PRIA NOTARY DTD used

[_SequencelIdentifier] Attribute

When multiple notaries are utilized, this indicates the order in which they should appear in the document.

[_TitleDescription] Attribute

The title or position of the notary.

[_UnparsedName] Attribute

The unparsed name of the notary.

NOTARY Deprecations

None

[_CERTIFICATE] Element

The _CERTIFICATE element contains data points necessary to complete a notarial certificate. It does not contain the certificate language at this time.

__CERTIFICATE DTD Definition

```
<!ELEMENT _CERTIFICATE (_SIGNER_IDENTIFICATION+)>
<!-- ATTLIST _CERTIFICATE
    _ID ID #IMPLIED
    _SequenceIdentifier CDATA #IMPLIED
    _SignerFirstName CDATA #IMPLIED
    _SignerMiddleName CDATA #IMPLIED
    _SignerLastName CDATA #IMPLIED
    _SignerNameSuffix CDATA #IMPLIED
    _SignerUnparsedName CDATA #IMPLIED
    _SignerCompanyName CDATA #IMPLIED
    _SignerTitleDescription CDATA #IMPLIED
    _SigningDate CDATA #IMPLIED
    _SigningCounty CDATA #IMPLIED
    _SigningState CDATA #IMPLIED-->
```

__CERTIFICATE Examples

Please see the following example listed below.

None

Child Elements:

[_SIGNER_IDENTIFICATION]

__CERTIFICATE Attributes

[_SequenceIdentifier] Attribute

If more than one certificate is used, this indicates the order in which they should appear on the document.

[_SignerFirstName] Attribute

The first name of the party signing before the notary.

[_SignerMiddleName] Attribute

The middle name of the party signing before the notary.

[_SignerLastName] Attribute

The last name of the party signing before the notary.

[_SignerNameSuffix] Attribute

The name suffix of the party signing before the notary.

[_SignerUnparsedName] Attribute

The unparsed name of the party signing before the notary.

[_SignerCompanyName] Attribute

The company name of the party signing before the notary.

[_SignerTitleDescription] Attribute

The title of the party signing before the notary.

[_SigningDate] Attribute

The date the notary performs the notarial act and signs the document.

[_SigningCounty] Attribute

The county in which the notary performs the notarial act.

[_SigningState] Attribute

The state in which the notary performs the notarial act.

CERTIFICATE Deprecations

None

[_SIGNER_IDENTIFICATION] Element

The _SIGNER_IDENTIFICATION element contains information regarding what type of identification was presented to the notary at the time of signing.

__SIGNER_IDENTIFICATION DTD Definition

```
<!ELEMENT _SIGNER_IDENTIFICATION EMPTY>
  <!ATTLIST _SIGNER_IDENTIFICATION
    _ID ID #IMPLIED
    _Description CDATA #IMPLIED
    _SequenceIdentifier CDATA #IMPLIED
    _Type (ProvidedIdentification | PersonallyKnown) #IMPLIED>
```

__SIGNER_IDENTIFICATION Examples

Please see the following example listed below.

None

Child Elements:

None

__SIGNER_IDENTIFICATION Attributes

[_Description] Attribute

The description of the type of identification provided by the party signing before the notary, e.g. drivers license and state of issuance.

[_SequenceIdentifier] Attribute

When multiple forms of identification are presented, this indicates the order in which they should appear in the document.

[_Type] Attribute

The type of identification provided by the party signing before the notary.

__SIGNER_IDENTIFICATION Deprecations

None

[RECORDING_ENDORSEMENT] Element

The RECORDING_ENDORSEMENT element contains the data that makes up the “time/date/instrument number” endorsement affixed to a document by the county recorder.

RECORDING_ENDORSEMENT DTD Definition

```
<!ELEMENT RECORDING_ENDORSEMENT (_VOLUME_PAGE*, _FEES?, _EXEMPTIONS*,
SIGNATORY?)>
<!ATTLIST RECORDING_ENDORSEMENT
  _ID ID #IMPLIED
  _Identifier CDATA #IMPLIED
  _OfficersName CDATA #IMPLIED
  _InstrumentNumberIdentifier CDATA #IMPLIED
  _PagesCount CDATA #IMPLIED
  _Volume CDATA #IMPLIED
  _VolumeType (Plat | Deed | Mortgage | Maps | Other) #IMPLIED
  _VolumeTypeOtherDescription CDATA #IMPLIED
  _RecordedDateTime CDATA #IMPLIED
  _RecordedCounty CDATA #IMPLIED
  _RecordedState CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED>
```

RECORDING_ENDORSEMENT Examples

Please see the following example listed below.

None

Child Elements:

[_VOLUME_PAGE]
 [_FEES]
 [_EXEMPTIONS]
 [SIGNATORY]

RECORDING_ENDORSEMENT Attributes

[Identifier] Attribute

Endorsement identification number assigned to each recording transaction, such as an audit number. This differs from the Recording Endorsement Instrument Number.

[OfficersName] Attribute

Name of the County Recording Official

[InstrumentNumberIdentifier] Attribute

Sequential number assigned to each recorded document by the a County Recorder

[_PagesCount] Attribute

Number of pages in document being recorded

[_Volume] Attribute

Volume or Book number assigned to recorder document by the recorder

[_VolumeType] Attribute

Volume type of the document referenced in the subject document

[_VolumeTypeOtherDescription] Attribute

A description of the volume type when Other is selected from the enumerated list

[_RecordedDateTime] Attribute

Date and time the document was recorded by recorder

[_RecordedCounty] Attribute

County in which the document is recorded

[_RecordedState] Attribute

State in which the document is recorded

[_SequenceIdentifier] Attribute

When multiple endorsements are applied to a document, this indicates the order in which they should appear.

RECORDING_ENDORSEMENT Deprecations

None

[_VOLUME_PAGE] Element

The VOLUME_PAGE element contains numeric references to the pages of a recorded document. Often, the recording endorsement only references the first page number of a document.

_VOLUME_PAGE DTD Definition

```
<!ELEMENT _VOLUME_PAGE EMPTY>
<!ATTLIST _VOLUME_PAGE
  _ID ID #IMPLIED
  _NumberIdentifier CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED>
```

_VOLUME_PAGE Examples

Please see the following example listed below.

None

Child Elements:

None

_VOLUME_PAGE Attributes

[_Number] Attribute

The page(s) of the recorded document

[_SequenceIdentifier] Attribute

The sequencing of multiple page references.

_VOLUME_PAGE Deprecations

None

[_FEES] Element

The _FEES element contains the total amount of all fees collected to record a document.

_FEES DTD Definition

```
<!ELEMENT _FEES (_RECORDING_FEE*)>
<!ATTLIST _FEES
  _ID ID #IMPLIED
  _TotalAmount CDATA #IMPLIED>
```

_FEES Examples

Please see the following example listed below.

None

Child Elements:

[\[_RECORDING_FEE\]](#)

_FEES Attributes

[_TotalAmount] Attribute

The total amount of all fees required to be remitted to the county recorder for recording a single document. (Depending on state laws and local ordinances there may not be a direct relationship between Recording Endorsement Fees and RESPA Fees.)

_FEES Deprecations

None

[_RECORDING_FEE] Element

The _RECORDING_FEE element contains the various fees that may be assessed when a document is recorded.

_RECORDING_FEE DTD Definition

```
<!ELEMENT _RECORDING_FEE EMPTY>
<!ATTLIST _RECORDING_FEE
  _ID ID #IMPLIED
  RecordingEndorsementFeeAmount CDATA #IMPLIED
  RecordingEndorsementFeeDescription CDATA #IMPLIED
  RecordingEndorsementFeeSequenceIdentifier CDATA #IMPLIED
  >
```

_RECORDING_FEE Examples

Please see the following example listed below.

None

Child Elements:

None

_RECORDING_FEE Attributes

[RecordingEndorsementFeeAmount] Attribute

The amount of a fee required for recording a document. Multiple fees could be required for a single document. (Depending on state laws and local ordinances there may not be a direct relationship between Recording Fee and RESPA Fees.)

[RecordingEndorsementFeeDescription] Attribute

Description of an individual fee required for recording a document. This could include document recording fees, mortgage taxes, various conveyance taxes, etc. (Depending on state laws and local ordinances there may not be a direct relationship between Recording Fee and RESPA Fees.)

[RecordingEndorsementFeeSequenceIdentifier] Attribute

When multiple fees are assessed, this indicates the order in which they should appear.

_RECORDING_FEE Deprecations

None

[_EXEMPTIONS] Element

The _EXEMPTIONS element contains information regarding recording fee exemptions and the amounts of those exemptions.

__EXEMPTIONS DTD Definition

```
<!ELEMENT _EXEMPTIONS EMPTY>
<!ATTLIST _EXEMPTIONS
  _ID ID #IMPLIED
  _Amount CDATA #IMPLIED
  _Description CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED>
```

__EXEMPTIONS Examples

Please see the following example listed below.

None

Child Elements:

None

__EXEMPTIONS Attributes

[_Amount] Attribute

Amount of Exemption from recording fees.

[_Description] Attribute

Description of Exemptions for Document Recording.

[_SequenceIdentifier] Attribute

When multiple exemptions exist, this indicates the order in which they should appear.

__EXEMPTIONS Deprecations

None

[EMBEDDED_FILE] Element

The EMBEDDED_FILE Element has been reused in its entirety from the MISMO Architecture Workgroup definition at <http://www.mismo.org>.

There is often a business need to pass a print image or other non-XML file format within a defined MISMO standard. When this is the case, MISMO recommends the use of a generic Embedded File Element. The Embedded File Element allows for maximum flexibility to identify and carry a variety of file types.

The Embedded File Container shall be defined as follows:

EMBEDDED_FILE – A common structure used to pass an image or file within an XML structure.

EMBEDDED_FILE DTD Definition

```
<!ELEMENT EMBEDDED_FILE (DOCUMENT?)>
<!ATTLIST EMBEDDED_FILE
  ID CDATA #IMPLIED
  EmbeddedFileType CDATA #IMPLIED
  EmbeddedFileVersion CDATA #IMPLIED
  EmbeddedFileName CDATA #IMPLIED
  FileEncodingType CDATA #IMPLIED
  FileDescription CDATA #IMPLIED
  MIMETYPE CDATA #IMPLIED
  PageCount CDATA #IMPLIED
  _SequenceIdentifier CDATA #IMPLIED>
```

EMBEDDED_FILE Example

Please see the following example listed below.

```
<EMBEDDED_FILE id="A12345"
  EmbeddedFileType="PDF"
  EmbeddedFileVersion="1.2"
  EmbeddedFileName="Brick.pdf"
  FileEncodingType="base64"
  FileDescription="Deed of Trust"
  MIMETYPE="application/pdf">
  PageCount="12"
  _SequenceIdentifier="1"
</DOCUMENT>SUKqAAgAAAATAP4ABAABAA////////wAEAE=
</DOCUMENT
</EMBEDDED_FILE>
```

Child Elements:

[\[DOCUMENT\]](#)

EMBEDDED_FILE Attributes

[EmbeddedFileType] Attribute

Identifies the program used to create the embedded file (i.e. Adobe Acrobat).

[EmbeddedFileVersion] Attribute

Identifies the version of the program used to create the embedded file (i.e. 4.0).

[EmbeddedFileName] Attribute

Identifies the file name of the embedded file (i.e. WarrantyDeed0001.pdf).

It has been pointed out that implementers may wish to include file PATH information in this variable. If PATH information is included, MISMO makes the recommendation to use URI syntax to express the PATH.

[FileEncodingType] Attribute

Specifies the type of encoding used on the embedded file (i.e. Base64).

Transfer Encoding names are registered by the Internet Assigned Numbers Authority, and the currently registered names can be viewed at:

<http://www.iana.org/assignments/transfer-encodings><http://www.iana.org/assignments/media-types/index.html> which may be useful to standardize the values in this attribute.

[FileDescription] Attribute

Provides additional information about the embedded file.

[MIMEType] Attribute

Indicates the Multipurpose Internet Mail Extensions type of the data in the DOCUMENT container.

A registered list of these types may be downloaded via ftp from: <ftp://ftp.isi.edu/in-notes/iana/assignments/mediatypes/media-types>. As a last resort it is recommended that the octet MIME type should be used.

MIME Media Types are registered by the Internet Assigned Numbers Authority, and the currently registered Types can be viewed at:

<http://www.iana.org/assignments/media-types/index.html>

A MIME type should be specified for all embedded files. For text formats that do not have a specific MIME type defined, the generic text MIME type of "text/plain" type may be specified. Similarly, the "application/octet-stream" MIME type may be specified for otherwise undefined binary formats. More information on MIME types may be found in the IETF RFC 2046 at

<http://www.ietf.org/rfc/rfc2046.txt?number=2046>.

[PageCount] Attribute

Indicates the number of pages encoded in the DOCUMENT element, if multiple EMBEDDED_FILE elements each contain a single page of a multi-page document, PageCount in each EMBEDDED_FILE would be "1".

[_SequenceIdentifier] Attribute

Indicates the page number if multiple EMBEDDED_FILE elements each contain a single page of a multi-page document.

EMBEDDED_FILE Deprecations

None

[DOCUMENT] Element

The DOCUMENT element contains the actual file defined in EMBEDDED_FILE.

DOCUMENT DTD Definition

```
<!ELEMENT DOCUMENT (ANY?)>
```

DOCUMENT Example

```
<DOCUMENT>SUkqAAgAAAATAP4ABAABAA////////wAEAE=  
</DOCUMENT
```

Child Elements:

None

DOCUMENT Attributes

None

DOCUMENT Deprecations

None

*Source*PRIA Logical Data Dictionary (LDD)

Version 2.4

PRIA Copyright Notice, Disclaimer and End-User License

Version 1.2 September 2003 (the "PRIA License" or the "License")

"This document or software (the ""Work"") is published by the Property Records Industry Association (""PRIA""). Copyright © 2002-2003 - writers referenced at www.pria.us (collectively or individually, a ""Licensor""). All rights reserved.