



# Integrated Justice Exchange Document Methodology, Naming, and Design Rules (MNDR) Working Draft, April 20, 2006

## Document identifier:

wd-ijtc-MNDR-1.0.0.doc

## Location:

Persistent: TBD

This Version: <http://www.oasis-open.org/apps/org/workgroup/legalxml-intj-exmndr/>

Previous Version <http://www.oasis-open.org/apps/org/workgroup/legalxml-intj-exmndr/documents.php>

## Technical Committee:

OASIS LegalXML Integrated Justice TC

## Chairs:

Ellen Perry, MTG Management Consultants

John Ruegg, LA County Information Systems Advisory Body (ISAB)

## Editors:

John Ruegg, LA County Information Systems Advisory Body (ISAB)

<[jruegg@isab.co.la.ca.us](mailto:jruegg@isab.co.la.ca.us)>

Marcus Leon, LA County Information Systems Advisory Body (ISAB)

<[mleon@isab.co.la.ca.us](mailto:mleon@isab.co.la.ca.us)>

Marguerite Soto, LA County Internal Services Department

<[msoto@co.la.ca.us](mailto:msoto@co.la.ca.us)>

## Contributors:

Scott Came, Justice Integration Solutions, Inc.

Tom Carlson, National Center for State Courts

Ellen Perry, MTG Management Consultants, L.L.C.

John Ruegg, LA County Information Systems Advisory Body (ISAB)

Nancy Rutter, Maricopa County, Arizona

## Abstract:

The purpose of this document is to establish guidance on how to develop GJXDM Information Exchange Package Documentation.

## Status:

This document has been drafted by the OASIS LegalXML Integrated Justice MNDR subcommittee and is being submitted for review/revision to the OASIS LegalXML Integrated Justice TC. It will also be given to other groups in the justice community for review/revision. These groups include the GJXDM Training and Technical Assistance

41 Committee (GTTAC), the Global XML Structure Task Force (XSTF), and the IJIS Institute  
42 XML Advisory Committee. This document is a Working Group Draft NOT yet accepted by  
43 the Working Group as reflecting its consensus; however, it will serve as the basis for  
44 discussions. As a work in progress, it should NOT be considered authoritative or final.  
45 Other subsequently issued documents will supersede this document

46  
47 Committee members should send their comments on this specification to the  
48 workgroup\_mailer@lists.oasis-open.org list. Others should subscribe to and send  
49 comments to the mailto:legalxml-intjustice@lists.oasis-open.org list. To subscribe, send  
50 an email message to mailto:legalxml-intjustice@lists.oasis-open.org?subject=Subscribe  
51 with the word "subscribe" as the body of the message.

52

---

53 **Notices**

54 OASIS takes no position regarding the validity or scope of any intellectual property or other rights  
55 that might be claimed to pertain to the implementation or use of the technology described in this  
56 document or the extent to which any license under such rights might or might not be available;  
57 neither does it represent that it has made any effort to identify any such rights. Information on  
58 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS  
59 Website. Copies of claims of rights made available for publication and any assurances of  
60 licenses to be made available, or the result of an attempt made to obtain a general license or  
61 permission for the use of such proprietary rights by implementors or users of this specification,  
62 can be obtained from the OASIS President.

63 OASIS invites any interested party to bring to its attention any copyrights, patents or patent  
64 applications, or other proprietary rights which may cover technology that may be required to  
65 implement this specification. Please address the information to the OASIS President.

66 Copyright © OASIS Open 2005. *All Rights Reserved.*

67 This document and translations of it may be copied and furnished to others, and derivative works  
68 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,  
69 published and distributed, in whole or in part, without restriction of any kind, provided that the  
70 above copyright notice and this paragraph are included on all such copies and derivative works.  
71 However, this document itself does not be modified in any way, such as by removing the  
72 copyright notice or references to OASIS, except as needed for the purpose of developing OASIS  
73 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual  
74 Property Rights document must be followed, or as required to translate it into languages other  
75 than English.

76 The limited permissions granted above are perpetual and will not be revoked by OASIS or its  
77 successors or assigns.

78 This document and the information contained herein is provided on an "AS IS" basis and OASIS  
79 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO  
80 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE  
81 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A  
82 PARTICULAR PURPOSE.

# Table of Contents

84	<b>1 INTRODUCTION .....</b>	<b>6</b>
85	1.1 AUDIENCE .....	6
86	1.2 SCOPE .....	6
87	1.2.1 <i>What this MNDR contains</i> .....	7
88	1.2.2 <i>What this MNDR does not address</i> .....	7
89	1.3 PURPOSE .....	7
90	1.4 TERMINOLOGY AND NOTATION .....	7
91	1.5 PRINCIPLES .....	8
92	1.6 TERMS AND DEFINITIONS .....	9
93	1.6.1 <i>Software Licensing Definitions</i> .....	9
94	1.7 SYMBOLS AND ABBREVIATIONS .....	9
95	1.8 RELATIONSHIP TO OTHER XML SPECIFICATIONS .....	11
96	1.8.1 <i>Global Justice XML Data Model (GJXDM)</i> .....	11
97	1.8.2 <i>OASIS Universal Business Language</i> .....	11
98	1.9 NORMATIVE REFERENCES .....	11
99	1.10 NON-NORMATIVE REFERENCES .....	12
100	<b>2 INFORMATION EXCHANGE PACKAGE DOCUMENTATION OVERVIEW .....</b>	<b>14</b>
101	2.1 BACKGROUND .....	14
102	2.1.1 <i>Justice Information Sharing Initiative (Global)</i> .....	14
103	2.1.2 <i>Global Justice XML Data Model (GJXDM)</i> .....	14
104	2.1.3 <i>GJXDM Information Exchange Package (GIEP)</i> .....	14
105	2.1.4 <i>GJXDM Information Exchange Package Documentation (GIEPD)</i> .....	14
106	2.2 GIEPD FUNCTIONAL USES .....	15
107	2.2.1 <i>Samples / Project starting points</i> .....	15
108	2.2.2 <i>Jurisdictional Standards</i> .....	15
109	2.2.3 <i>Geopolitical Standards</i> .....	15
110	2.2.4 <i>Exchange Documents, Reference Documents</i> .....	15
111	2.3 GIEPD DEVELOPMENT PROCESS GUIDANCE .....	15
112	2.3.1 <i>Workgroup Composition</i> .....	15
113	2.3.2 <i>Development Process Steps</i> .....	16
114	<b>3 INFORMATION EXCHANGE PACKAGE DOCUMENTATION (GIEPD) .....</b>	<b>18</b>
115	3.1 GIEPD DEVELOPMENT METHODOLOGY DESCRIPTION .....	18
116	3.2 GIEPD INFORMATION EXCHANGE DOMAIN MODEL DESCRIPTION .....	18
117	3.2.1 <i>Graphical Domain Model Documentation Rules</i> .....	22
118	3.2.2 <i>Textual Domain Model</i> .....	23
119	3.2.3 <i>Textual Domain Model Documentation Rules</i> .....	24
120	3.3 GIEPD SCHEMAS .....	26
121	3.3.1 <i>Document Schema</i> .....	27
122	3.3.2 <i>Subset Schema</i> .....	27
123	3.3.3 <i>Constraint Schema</i> .....	27
124	3.3.4 <i>Extension Schema</i> .....	27
125	3.4 GIEPD INSTANCE DOCUMENTS .....	28
126	3.4.1 <i>Sample Source Documents &amp; XML Instance Documents</i> .....	28
127	3.5 OTHER SUPPORTING DOCUMENTATION .....	29
128	3.6 PACKAGING OF GIEPD ARTIFACTS (I.E., ZIP) AND NAMING RULES .....	29
129	<b>4 SCHEMA SET AND INSTANCE DOCUMENT NAMING AND DESIGN RULES .....</b>	<b>32</b>
130	4.1 GENERAL SCHEMA SET NAMING AND DESIGN RULES .....	32
131	4.1.1 <i>General Naming Rules</i> .....	32
132	4.1.2 <i>Namespaces and Schema Locations</i> .....	36

133	4.1.3	<i>External Code List Rules</i> .....	37
134	4.1.4	<i>General Type Definitions</i> .....	38
135	4.1.5	<i>Complex Type Definitions</i> .....	38
136	4.1.6	<i>Complex Type Naming Rules</i> .....	40
137	4.1.7	<i>Attribute Declarations</i> .....	40
138	4.1.8	<i>Element Declarations and Naming Rules</i> .....	44
139	4.1.9	<i>Schema Documentation and Annotations</i> .....	46
140	4.1.10	<i>Schema Version Numbering Rules</i> .....	48
141	4.1.11	<i>Import versus Include</i> .....	52
142	4.1.12	<i>Character encoding</i> .....	54
143	4.1.13	<i>XSD:notation</i> .....	55
144	4.1.14	<i>XSD:all</i> .....	55
145	4.1.15	<i>XSD:choice</i> .....	55
146	4.2	SUBSET SCHEMA NAMING AND DESIGN RULES .....	55
147	4.2.1	<i>Rules for Conformant Subset Schemas</i> .....	56
148	4.2.2	<i>Subset Namespace and Filename Rules</i> .....	56
149	4.2.3	<i>Subset Schema File Layout</i> .....	57
150	4.2.4	<i>Subset Schema Generation Tool (SSGT)</i> .....	58
151	4.3	CONSTRAINT SCHEMA NAMING AND DESIGN RULES .....	58
152	4.3.1	<i>Rules for Conformant Constraint Schemas</i> .....	58
153	4.3.2	<i>Constraint Namespace and Filename Rules</i> .....	59
154	4.4	EXTENSION SCHEMA NAMING AND DESIGN RULES .....	60
155	4.4.1	<i>Extension Schema File Layout</i> .....	61
156	4.4.2	<i>Extension Patterns</i> .....	62
157	4.4.3	<i>Controlling Type extension/restriction (final)</i> .....	69
158	4.4.4	<i>Controlling Type and Element Substitution (block)</i> .....	70
159	4.5	DOCUMENT SCHEMA NAMING AND DESIGN RULES .....	71
160	4.5.1	<i>Document Schema File Layout</i> .....	72
161	4.6	INSTANCE NAMING AND DESIGN RULES .....	74
162	4.6.1	<i>Root Element</i> .....	74
163	4.6.2	<i>XML Instance validation</i> .....	74
164	4.6.3	<i>Character encoding</i> .....	75
165	4.6.4	<i>Empty content</i> .....	75
166		<b>APPENDIX A. GJXDM MNDR CHECKLIST</b> .....	<b>77</b>
167		<b>APPENDIX B. APPROVED ACRONYMS AND ABBREVIATIONS</b> .....	<b>98</b>
168		<b>APPENDIX C. (INFORMATIVE) TECHNICAL TERMINOLOGY</b> .....	<b>100</b>
169		<b>APPENDIX D. (INFORMATIVE) EXAMPLE GIEPD(S)</b> .....	<b>104</b>
170		<b>APPENDIX E. (INFORMATIVE) ONGOING WORK ITEMS</b> .....	<b>105</b>
171		<b>APPENDIX F. (INFORMATIVE) ACKNOWLEDGMENTS</b> .....	<b>106</b>
172		<b>APPENDIX G. (INFORMATIVE) REVISION HISTORY</b> .....	<b>107</b>

173

---

# 1 Introduction

There is a need to develop standards and best practices for the development of GJXDM conformant information exchange packages. The purpose of this document is to provide:

- A methodology for the construction of GJXDM-conformant information exchange package documentation
- Naming and design rules for the artifacts called for by the methodology
- Guidelines for the customization of GJXDM schema structures

This work will benefit the justice community in the following ways:

- It will improve interoperability by promoting consistent use of GJXDM in constructing schemas
- It will lower risk and improve efficiency of information exchange projects between justice partners by defining, consolidating, and disseminating best practices
- It will provide a formal, normative standard that exchange partners can use to establish quality assurance criteria.

## 1.1 Audience

This document is intended to provide consistent, practical guidance for technical and business users seeking to create GJXDM-conformant information exchange packages.

Technical Users – to provide technical guidelines, methods, and rules to analysts and developers

Business Users – to provide a concrete framework for business (subject matter) experts to ensure that development work is completed accurately and completely, with appropriate quality assurance features.

## 1.2 Scope

This document covers the following sections:

Information Exchange Package Documentation Overview – Provides the background and functional uses for the Global Information Exchange Package Documentation (GIEPD); leverages the GJXDM Information Exchange Package Documentation Guidelines [**GJXDM IEPD**] published by the GJXDM XML Structured Task Force.

GIEPD Development Process Guidance provides information about the composition and responsibilities of the development team, including the steps to follow to define an exchange, along with the required artifacts.

Information Exchange Package Documentation (GIEPD) - This section describes the required and optional components which make up a GIEPD including Domain Modeling documentation, Schema(s), XML Instance Documentation, other supporting documents and MNDR standards for Naming and Packaging the GIEPD into a MNDR conformant ZIP file .

216 Schema Set & Instance Document Naming and Design Rules – Provides detailed instructions  
217 about how to code GJXDM conformant XML schemas, including information about XML types,  
218 elements, attributes, and documentation, as well as other schema rules and guidelines. Also,  
219 describes the rules for constructing instance documents, including requirements for root elements  
220 and validation methods.

221

## 222 **1.2.1 What this MNDR contains**

223 This document provides step by step guidelines for the analysis, high level object modeling,  
224 GJXDM mapping, construction, and customization of all components of GJXDM conformant  
225 information exchange packages, including instance document examples.

226

## 227 **1.2.2 What this MNDR does not address**

228 This document does not address instructions and requirements regarding messaging and other  
229 end-to-end transaction requirements.

230

## 231 **1.3 Purpose**

232 This document seeks to achieve and support interoperability beyond what is provided by XML  
233 Schema and the GJXDM by defining standard development methodologies, best practices and  
234 exchange artifacts that meet the needs of technical and business users. It leverages and extends  
235 standards defined elsewhere in the technical and justice communities, including rules defined and  
236 endorsed by GTRI, the XSTF, OASIS, and W3C.

237

## 238 **1.4 Terminology and Notation**

239 The key words MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD,  
240 SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL in this document are to  
241 be interpreted as described in Internet Engineering Task Force (IETF) Request for  
242 Comments (RFC) 2119. Non-capitalized forms of these words are used in the regular  
243 English sense.

244

245 [Definition] – A formal definition of a term. Definitions are normative.

246

247 [Example] – A representation of a definition or a rule. Examples are informative.

248

249 [Note] – Explanatory information. Notes are informative.

250

251 [RRRn] – Identification of a rule that requires conformance to ensure that an XML Schema is  
252 conformant to OASIS Integrated Justice Exchange Document Methodology, Naming, and Design  
253 Rules (MNDR). The value RRR is a prefix to categorize the type of rule where the value of RRR  
254 is as defined in Figure 1 and n (1..n) indicates the sequential number of the rule within its  
255 category. In order to ensure continuity across versions of the specification, rule numbers that are  
256 deleted in future versions will not be re-issued, and any new rules will be assigned the next higher  
257 number – regardless of location in the text. Future versions will contain an appendix that lists  
258 deleted rules and the reason for their deletion. Only rules and definitions are normative; all other  
259 text is explanatory.

260

261 **Figure 1 – Rule Prefix Token Value**

262

Attribute Declaration Rules	(ATD)
Code List Rules	(CDL)
ComplexType Definition Rules	(CTD)
ComplexType Naming Rules	(CTN)
Documentation Rules	(DOC)
Element Declaration Rules	(ELD)
General Naming Rules	(GNR)
General Type Definition Rules	(GTD)
General XML Schema Rules	(GXS)
Instance Document Rules	(IND)
Mapping Documentation Rules	(MAP)
Modeling Constraints Rules	(MDC)
Namespace Rules	(NMS)
Root Element Declaration Rules	(RED)
Schema Structure Modularity Rules	(SSM)
Standards Adherence Rules	(STA)
Versioning Rules	(VER)

263

## 264 **1.5 Principles**

265 Open Standards – All artifacts will rely on open standards to represent required content.

266

267 Reuse versus Reinvent – This work incorporates proven work developed by other groups,  
268 including GJXDM methods and standards and OASIS UBL. For example, this work is patterned  
269 after the successful development of similar methodology, naming, and design rules in the OASIS  
270 UBL Technical Committee

271

272 Collaboration - In developing these proposed rules and guidelines, the subcommittee will consult  
273 and collaborate with other groups in the justice community (including but not limited to the  
274 GJXDM Training and Technical Assistance Committee [GTTAC], the Global XML Structure Task  
275 Force [XSTF], and the IJIS Institute XML Advisory Committee.) In consulting with these groups,



276 the subcommittee will seek to incorporate or reference, as appropriate, any existing or emerging  
277 work product that they may have

278

## 279 **1.6 Terms and Definitions**

280 This document assumes a basic knowledge of XML which allows designers to create their own  
281 customized tags, enabling the definition, transmission, validation, and interpretation of data  
282 between applications and between organizations. A select set of Technical Terms and Definitions  
283 can be found in the Appendix C - Technical Terminology.

284

### 285 **1.6.1 Software Licensing Definitions**

#### 286 **Proprietary**

287 Proprietary refers to software or formats in which an entity retains distribution and modification  
288 rights. Proprietary software, and particularly products produced with the software, can be difficult  
289 to use without owning a license to the producing software.

290 In some cases, a proprietary format becomes ubiquitous to the point that it is a de facto standard.

291 Proprietary may refer to a software application itself, or to a product of a software application. For  
292 example, the underlying format of a document produced with a word processor may be  
293 proprietary, independent of whether the word processor is proprietary.

#### 294 **Open Source**

295 Open Source refers to a group of licenses that allow entities to distribute software while allowing  
296 others to copy and modify it. Typically, Open Source licenses require that, if an entity modifies  
297 and distributes their modifications, they must also grant the same redistribution rights. This  
298 ensures that improvements to Open Source products are returned to the development  
299 community.

#### 300 **Freely Available**

301 Freely Available refers to software or services that are free to use, but are not distributed under  
302 an Open Source license.

303

## 304 **1.7 Symbols and Abbreviations**

### 305 **CCTS**

306 Core Components Technical Specifications

### 307 **DOJ**

308 Department of Justice

### 309 **ebXML**

310 Electronic Business XML

### 311 **GIEP**

312 GLOBAL Information Exchange Package

### 313 **GIEPD**

314 GLOBAL Information Exchange Package Documentation

### 315 **GJXDM**

316 Global Justice XML Data Model

### 317 **GTRI**

318 Georgia Technical Research Institute

319 **HTML**  
320     HyperText Markup Language  
321 **IETF**  
322     Internet Engineering Task Force  
323 **IEP**  
324     Information Exchange Package  
325 **ISO**  
326     International Standards Organization  
327 **JXDM**  
328     justice xml data model  
329 **LCC**  
330     lowerCamelCase  
331 **MNDR**  
332     Methodology, Naming and Design Rules  
333 **NDR**  
334     Naming and Design Rules  
335 **NS**  
336     NameSpace  
337 **OJP**  
338     Office of Justice Planning within the Federal Department of Justice  
339 **OASIS**  
340     Organization for the Advancement of Structured Information Standards  
341 **PDF**  
342     Postscript Data Format  
343 **RFC**  
344     Request For Comment  
345 **SSGT**  
346     Subset Schema Generation Tool  
347 **UBL**  
348     Universal Business Language  
349 **UCC**  
350     UpperCamelCase  
351 **UML**  
352     Universal Modeling Language  
353 **URI**  
354     Universal Resource Identifier  
355 **URL**  
356     Universal Resource Locator  
357 **W3C**  
358     World Wide Web Consortium  
359 **XMI**  
360     XML Metadata Interchange

361 **XML**  
362 eXtensible Markup Language  
363 **XSD**  
364 XML Schema Definition  
365 **XSTF**  
366 XML Structure Task Force Committee sponsored the Federal Department of Justice  
367

## 368 **1.8 Relationship to other XML Specifications**

369 The MNDR specification leverages other existing, non-proprietary XML specifications wherever  
370 possible. In particular, the specification has dependencies on the **[GJXDM]** and the **[UBL NDR]**  
371 specifications.

### 372 **1.8.1 Global Justice XML Data Model (GJXDM)**

373 **[GJXDM]** conformance, as defined by the GJXDM Implementation Guidelines (**[GJXDM**  
374 **IMPLEMENT]**), is a core objective of this specification. The **[GJXDM]** is an XML standard  
375 designed specifically for justice information exchanges, providing law enforcement, public safety  
376 agencies, prosecutors, public defenders, and the judicial branch with a tool to effectively share  
377 data and information in a timely manner. The **[GJXDM]** provides a library of reusable  
378 components that can be combined to automate justice information exchanges. The **[GJXDM]**  
379 removes the burden from agencies to independently create exchange standards. Because of its  
380 extensibility, there is more flexibility to deal with unique agency requirements and changes.  
381 Through the use of a common vocabulary that is understood system to system, **[GJXDM]**  
382 enables access from multiple sources and reuse in multiple applications.

383

384 The **[GJXDM]** is most useful for describing common objects such as persons and locations and  
385 criminal justice-specific processes such as arrest, booking, jail and prosecution. The **[GJXDM]** is  
386 not as well developed for describing non-criminal information exchanges and processes. The  
387 MNDR assumes the **[GJXDM]** version 3.0.3 structures and definitions. A separate version of the  
388 MNDR will be released to support newer version(s) of **[GJXDM]** and **[NIEM]**.

389

### 390 **1.8.2 OASIS Universal Business Language**

391 **[UBL]** is an OASIS Standard that provides a single ubiquitous language for business  
392 communication that takes into account the requirements common to all enterprises. **[UBL]**  
393 provides a library of reusable components that can be combined to create electronic business  
394 schemas. This shared library is essential to interoperability; without a common set of base  
395 components, each document format would risk redefining addresses, locations, and other basic  
396 information in similar but incompatible ways. Many of the Naming & Design Rules from the **[UBL**  
397 **NDR]** have been incorporated into this **[MNDR]** specification.

398

## 399 **1.9 Normative References**

400

401 **[GJXDM]** Global Justice XML Data Model 3.0.3, <http://www.it.ojp.gov/gjxdm>, US  
402 DOJ OJP, 2005.

403

- 404 **[GJXDM IMPLEMENT]** *Global JXDM Implementation Guidelines*,  
405 [http://it.ojp.gov/topic.jsp?topic\\_id=138](http://it.ojp.gov/topic.jsp?topic_id=138), US DOJ OJP, 2005.  
406
- 407 **[GJXDM USER]** *Catherine Plummer, GJXDM Users Guide*,  
408 <http://it.ojp.gov/documents/GJXDMUserGuide.pdf> ,  
409 SEARCH, 2005.  
410
- 411 **[Namespaces]** T. Bray, Namespaces in XML, [http://www.w3.org/TR/1999/REC-xml-](http://www.w3.org/TR/1999/REC-xml-names-19990114)  
412 [names-19990114](http://www.w3.org/TR/1999/REC-xml-names-19990114) , January 14, 1999.  
413
- 414 **[UBL]** B. Meadows, L. Seaburg (editors), Universal Business Language 1.0,  
415 <http://docs.oasis-open.org/ubl/cd-UBL-1.0/> , OASIS Standard,  
416 September 15 2004  
417
- 418 **[UBL NDR]** M. Crawford (editor), Universal Business Language (UBL) Naming and  
419 Design Rules, [http://www.oasis-](http://www.oasis-open.org/committees/download.php/10323/cd-UBL-NDR-1.0Rev1c.pdf)  
420 [open.org/committees/download.php/10323/cd-UBL-NDR-1.0Rev1c.pdf](http://www.oasis-open.org/committees/download.php/10323/cd-UBL-NDR-1.0Rev1c.pdf) ,  
421 OASIS Standard, November 15, 2004  
422
- 423 **[RFC2119]** S. Bradner, Key words for use in RFCs to Indicate Requirement Levels,  
424 <http://www.ietf.org/rfc/rfc2119.txt>, IETF RFC 2110, March 1997.  
425
- 426 **[Schema Part 1]** H.S. Thompson, D. Beech, M. Maloney, N. Mendelsohn, XML Schema  
427 Part 1: Structures Second Edition, [http://www.w3.org/TR/2004/REC-](http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/)  
428 [xmlschema-1-20041028/](http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/) , W3C Recommendation, October 28, 2004  
429
- 430 **[Schema Part 2]** H.P. Biron, A. Malhotra, XML Schema Part 2:Data Types Second  
431 Edition, <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/> ,  
432 W3C Recommendation, October 28, 2004  
433
- 434 **[UML]** Universal Modeling Language,  
435 <http://www.omg.org/technology/documents/formal/uml.htm> , Object  
436 Management Group (OMG) UML specifications.  
437
- 438 **[XML 1.0]** T. Bray, Extensible Markup LanguageNamespaces in (XML) 1.0 (Third  
439 Edition), <http://www.w3.org/TR/REC-xml/REC-XML-20040204>, W3C  
440 Recommendation, December 2002.  
441  
442

## 443 **1.10 Non-Normative References**

- 444
- 445 **[CONCEPT Maps]** Joseph D. Novak, The Theory Underlying Concept Maps and How to  
446 Construct Them, <http://cmap.coginst.uwf.edu/info/printer.html> , Cornell  
447 University  
448

449	<b>[DON NDR]</b>	CIO Office of the Department of Navy (DON), Chief Information Officer, Naming and Design Rules, <a href="http://xml.coverpages.org/DON-XML-NDR20050127-33942.pdf">http://xml.coverpages.org/DON-XML-NDR20050127-33942.pdf</a> , Final Version 2.0. , January 2005. 168 pages
450		
451		
452		
453	<b>[GJXDM HELP]</b>	GJXDM Virtual Help Desk, <a href="http://it.ojp.gov/gjxdm/helpdesk/">http://it.ojp.gov/gjxdm/helpdesk/</a>
454		
455	<b>[GJXDM IEPD]</b>	M. Hulme, GJXDM Information Exchange Package Documentation Guidelines, <a href="http://it.ojp.gov/process_links.jsp?link_id=4581">http://it.ojp.gov/process_links.jsp?link_id=4581</a> , March 2, 2005.
456		
457		
458		
459	<b>[GLOBAL]</b>	Informational Resource on Federal Department of Justice GLOBAL Organization with links to numerous Justice Information Sharing Standards & Specifications.
460		
461		
462		<a href="http://it.ojp.gov/topic.jsp?topic_id=8">http://it.ojp.gov/topic.jsp?topic_id=8</a>
463		
464	<b>[ISO 11179]</b>	Links to general information and specifications covering the International Standards Organization (ISO) 11179 specification.
465		
466		<a href="http://en.wikipedia.org/wiki/ISO/IEC_11179">http://en.wikipedia.org/wiki/ISO/IEC_11179</a>
467		
468	<b>[LegalXML ECF]</b>	R. Winters (editor), LegalXML Electronic Court Filing 3.0,
469		<a href="http://docs.oasis-open.org/legalxml-courtfilling/specs/ecf/v3.0/ecf-v3.0-spec-cd01.zip">http://docs.oasis-open.org/legalxml-courtfilling/specs/ecf/v3.0/ecf-v3.0-spec-cd01.zip</a> , November 15, 2005
470		
471		
472	<b>[MNDR Charter]</b>	OASIS Integrated Justice Technical Committee, MNDR Charter,
473		<a href="http://www.oasis-open.org/apps/org/workgroup/legalxml-intj-exmndr/description.php">http://www.oasis-open.org/apps/org/workgroup/legalxml-intj-exmndr/description.php</a>
474		
475	<b>[NIEM]</b>	NIEM Concept of Operations, <a href="http://www.niem.gov">http://www.niem.gov</a> , DOJ/DHS, October 7, 2005.
476		
477		
478	<b>[SSGT Introduction]</b>	Document describing the Schema Subset Generation Tool and its application in developing Subset Schemas for GJXDM Information Exchange Packages.
479		
480		
481		<a href="http://justicexml.gtri.gatech.edu/subset_tool.html">http://justicexml.gtri.gatech.edu/subset_tool.html</a>
482		
483	<b>[UBL Customization]</b>	Eduardo Gutenberg (editor), Guidelines For The Customization of UBL v1.0 Schemas, <a href="http://docs.oasis-open.org/ubl/cd-UBL-1.0/doc/cm/wd-ubl-cmsc-cmguidelines-1.0.html">http://docs.oasis-open.org/ubl/cd-UBL-1.0/doc/cm/wd-ubl-cmsc-cmguidelines-1.0.html</a> , OASIS Working Draft 1.0, April 22, 2004
484		
485		
486		
487	<b>[XML TOPICS]</b>	XML Coverpages hosted by OASIS provides numerous links on XML topics and standards, <a href="http://xml.coverpages.org/xml.html">http://xml.coverpages.org/xml.html</a>
488		
489		

---

## 490 2 Information Exchange Package Documentation 491 Overview

492

### 493 2.1 Background

494

495 This section provides definitions of key terms and organizations, a delineation of GIEPD  
496 functional uses, and guidance on the GIEPD development process.

497

#### 498 2.1.1 Justice Information Sharing Initiative (Global)

499 The efforts of the Global Justice Information Sharing Initiative (Global) Advisory Committee  
500 (GAC) have direct impact on the work of more than 1.2 million justice professionals. The  
501 importance of the organization's mission, however, positions Global to impact citizens of the U.S.,  
502 Canada, and beyond. Global's mission — the efficient sharing of data among justice entities — is  
503 at the very heart of modern public safety and law enforcement.

504 Global is a “group of groups,” representing more than thirty independent organizations spanning  
505 the spectrum of law enforcement, judicial, correctional, and related bodies. Member organizations  
506 participate in Global out of shared responsibility and shared belief that, together, they can bring  
507 about positive change in inter-organizational communication and data sharing.

508

#### 509 2.1.2 Global Justice XML Data Model (GJXDM)

510 The Global JXDM is an XML standard designed specifically for criminal justice information  
511 exchanges, providing law enforcement, prosecution, defense, probation, and the judicial branch  
512 with a tool to effectively share data and information in a timely manner. The Global JXDM  
513 removes the burden from agencies to independently create exchange standards, and because of  
514 its extensibility, there is more flexibility to deal with unique agency requirements and changes.  
515 Through the use of a common vocabulary that is understood system to system, Global JXDM  
516 enables access from multiple sources and reuse in multiple applications.

517

#### 518 2.1.3 GJXDM Information Exchange Package (GIEP)

519 An “Information Exchange Package” represents a set of data that is transmitted for a specific  
520 business purpose. It is the actual XML instance that delivers the payload or information. (The  
521 word “package” as used herein refers to a package of the actual data, not a package of artifacts  
522 documenting the structure and content of the data.) An Information Exchange Package can be  
523 prefixed with “GJXDM” to indicate or highlight that the IEP is GJXDM-conformant, as in “GJXDM  
524 Information Exchange Package.” The fact that an IEP is GJXDM-conformant may be readily  
525 apparent from the context, so it is not absolutely necessary to use the word “GJXDM” even if the  
526 IEP is GJXDM-conformant.

527

#### 528 2.1.4 GJXDM Information Exchange Package Documentation (GIEPD)

529 “Information Exchange Package Documentation” is a collection of artifacts that describe the  
530 structure and content of an Information Exchange Package. It does not specify other interface

531 layers (such as web services). It can optionally be prefixed with “GJXDM” to indicate or highlight  
532 that a resulting IEP is GJXDM-conformant.

533

## 534 **2.2 GIEPD Functional Uses**

535 Global Information Exchange Package Documentations are not technical specifications, in the  
536 usual sense. Rather, they fulfill a number of higher level functions:

### 537 **2.2.1 Samples / Project starting points**

538 Global Information Exchange Package Documentations may act as starting points for further  
539 development. They may be used in part or in whole. The substantive information within a GIEPD  
540 may be used apart from other artifacts. Alternately, information about the development process  
541 itself may be used as a base for creating additional GIEPDs.

### 542 **2.2.2 Jurisdictional Standards**

543 Global Information Exchange Package Documentations may define jurisdictional standards. A  
544 high-level GIEPD may be further refined for sub-areas within a jurisdiction. Jurisdictions may  
545 make a GIEPD a mandatory standard.

### 546 **2.2.3 Geopolitical Standards**

547 Global Information Exchange Package Documentations may define national or state standards.  
548 The GIEPD may be further refined, constrained, or expanded at the state or local level.

### 549 **2.2.4 Exchange Documents, Reference Documents**

550 Global Information Exchange Package Documentations may be used to define message level  
551 and document level exchanges.

552

## 553 **2.3 GIEPD Development Process Guidance**

554 The purpose of this section is to provide guidance on how to develop GJXDM information  
555 exchange package documentation

### 556 **2.3.1 Workgroup Composition**

557 GJXDM Exchange Documents are created by a team of people called a *workgroup*. A workgroup  
558 is a group of people who are responsible for building the GIEPD. It is important to note that an  
559 individual can play more than one role in the workgroup. To be effective, the workgroup needs to  
560 contain both technical members and members with expertise in the business area of the  
561 exchange. In particular:

562 The workgroup **MUST** contain members with business expertise in the business area to be  
563 addressed by the exchange.

564 The workgroup **MUST** contain members with technical expertise in GJXDM, W3C XML Schema  
565 development, and domain modeling.

566 The workgroup **MUST** contain a facilitator whose role is:

- 567 • to coordinate the efforts of the workgroup
- 568 • to ensure that the development process documented here is followed by the workgroup
- 569 • to ensure that the other methodology, naming, and design rules documented in this  
570 specification are followed by the workgroup

## 571 **2.3.2 Development Process Steps**

572 The overall goal of the exchange document development process is to produce a package of  
573 artifacts that adhere to the methodology, naming and design rules in this specification. The rules  
574 in this section are intended to facilitate this goal.

575

### 576 **2.3.2.1 Modeling and Mapping Meeting(s)**

577 The workgroup SHOULD meet in person to start the development process.

578 The agenda for these meetings MUST include the following:

- 579 • Appropriate education of workgroup members about the development process to be  
580 followed
- 581 • Appropriate education of workgroup members about domain modeling and GJXDM  
582 mapping techniques and notations to be used
- 583 • Agreement among the workgroup as to how decisions will be made (e.g., by vote or  
584 consensus)
- 585 • Assigning the responsibility for each task to workgroup members
- 586 • Domain modeling of the exchange document structure
- 587 • Mapping of the domain model to GJXDM

588 The workgroup facilitator and/or technical workgroup members SHOULD create a candidate  
589 domain model prior to the first meeting. The candidate domain model SHOULD be based on  
590 existing paper document forms (if available) or input from business experts, to the extent feasible.

591 If it is not possible to complete the GJXDM mapping in the initial meeting, then the facilitator  
592 SHOULD ensure that the workgroup's attention be focused on the most significant domain model  
593 elements, as well as any elements that are potentially controversial.

594

### 595 **2.3.2.2 Completion of Mapping**

596 If the workgroup is unable to complete the domain-GJXDM mapping in the initial meeting, the  
597 facilitator MUST coordinate the efforts of the workgroup's business and technical/GJXDM experts  
598 to complete the mapping subsequent to the initial meeting.

599 The facilitator MUST present the mapping artifact to the workgroup for review, using whatever  
600 means is appropriate for the workgroup (e.g., in-person meeting, email, posting to a website).

601

### 602 **2.3.2.3 Schema and GIEPD Creation**

603

604 Following workgroup approval of the domain-GJXDM mapping, the facilitator MUST coordinate  
605 the efforts of the workgroup's technical/GJXDM experts to create XML Schemas based on the  
606 mapping. These schemas MUST adhere to the naming and design rules documented elsewhere  
607 in this specification.

608 All artifacts developed by the workgroup MUST be packaged according to the guidelines  
609 established in Section 3 – Information Exchange Package Documentation (GIEPD).

610

### 611 **2.3.2.4 Iterative Process**

612

613 The process outlined here is designed to be used in an iterative fashion, in which the steps in this  
614 section are repeated as often as needed. For example, a jurisdiction may develop an initial



615 version of an exchange document package, and at some point in the future may discover new  
616 requirements (or changes to previous requirements) that compel creation of a new version of the  
617 package. In this case, a workgroup would be formed, the workgroup would use the previous  
618 domain model and make any necessary changes, which would compel changes to the mapping  
619 and then the schema set. Similarly, a national association may develop a reference exchange  
620 document to serve as a model for local jurisdictions to use for a particular exchange. When  
621 modifying a reference exchange document, it is expected that the local jurisdiction would begin by  
622 forming a workgroup, modifying the domain model as necessary, then modifying the mapping and  
623 schemas accordingly.

624

### 625 **2.3.2.5 Openess of Artifacts**

626 An important goal of the development process is to produce artifacts that are usable by the widest  
627 possible audience. Consumers of the artifacts cannot be required to purchase expensive  
628 modeling or development tools in order to use the artifacts.

629 Also, the iterative nature of the development process requires artifacts that can be easily revised.  
630 The degree to which artifacts are reusable in new contexts is directly proportional to the  
631 openness of the artifacts and the tools used to create them.

632 See Section 3 for a delineation of artifact documentation requirements. The rules and guidelines  
633 included in Section 3 provide for openness of artifacts.

634

---

## 635 3 Information Exchange Package Documentation 636 (GIEPD)

637 Mandatory and optional components of a GJXDM information exchange package document are  
638 as described in this section of the MNDR. The GIEPD is composed of the following components:

639

- 640 • Description of Development Methodology & Participants
- 641 • Domain Model Requirements (Graphical and Textual)
- 642 • Schema Set Requirements
- 643 • Supporting Documentation Requirements
- 644 • GIEPD Packaging Standard

645

### 646 3.1 GIEPD Development Methodology Description

647 A GIEPD Development Methodology description delineates the process used in creating a  
648 particular GIEPD. This documentation MUST include:

649

- 650 • **A Description of the GIEPD Development Process**

651

652 GIEPDs MUST document the steps involved in their development. This helps ensure that  
653 best practices were followed in the development of the GIEPD. The process used to  
654 create the exchange document package, SHOULD BE recorded in Microsoft Word  
655 document(s).

656

- 657 • **Participants in the GIEPD Development Process**

658

659 GIEPDs Development Process documentation MUST list participants in the process.  
660 Listing participants, including their affiliation and role (business, technical etc) helps  
661 ensure that a broad spectrum of participants was involved. This, in turn, ensures that the  
662 resulting GIEPD is applicable to most uses within the realm for which it was developed.

663

### 664 3.2 GIEPD Information Exchange Domain Model Description

665 Information Exchange Domain Model documentation acts as a guide to aid in the future  
666 implementation or modification of the GIEPD.

667

#### 668 **[Definition] Domain model**

669 A graphical representation of the consensus of a group of business subject-  
670 matter experts as to the structure and content of an exchange document.

671

#### 672 **[Definition] Domain model element**

673 Any symbol in a domain model used to describe a part of the structure of an  
674 exchange document. Elements include symbols used to describe individual data

675 elements, entire named data structures (or collections of data elements), and  
676 relationships between data elements or structures.

677 Domain modeling is an effort to establish agreement among project stakeholders regarding the  
678 content and structure of information in the document. It occurs after the business requirements of  
679 the exchange have been modeled, and before the exchange document structure is represented in  
680 XML Schema.

681 The output of domain modeling is a domain model. However, the creation of the actual model  
682 artifact is only part of the objective. Equally important is that business stakeholders—those who  
683 understand the necessary structure of the exchange document—reach consensus on what that  
684 structure shall be. Only when this consensus has been reached is the domain modeling step  
685 concluded.

686

687 A workgroup **MUST** create a domain model as part of the GIEPD document package. The  
688 domain model **MUST** adhere to the other methodology rules established in this section.

689

690 The domain model **MUST** be drafted before XML Schemas are created. The domain model **MAY**  
691 be edited as new requirements are discovered. Every data structure and data element in an  
692 exchange document's XML Schema(s) **MUST** be associated explicitly with at least one element in  
693 the domain model.

694

695 The domain model **MUST** be created in interactive modeling sessions with business subject-  
696 matter experts and is a representation of the consensus of the workgroup. Depending on the  
697 exchange under review, this could include a review of existing models, documents, vocabulary, a  
698 discussion of work flow processes, or other substantive discussions. The domain model **MUST** at  
699 all times reflect the consensus of the business subject-matter experts as to the structure and  
700 content of the exchange document.

701

702 Workgroups should recognize that the GJXDM already effectively models a wide range of justice  
703 domain concepts. Therefore, if GJXDM concepts fit the domain being modeled, the exchange  
704 document domain model **SHOULD** incorporate GJXDM concepts. However, care should be  
705 taken that the GJXDM data model does not take precedence over the functional business  
706 requirements.

707

708 • **Domain Model of business data and relationships.**

709

710 **Concept maps** - are simple graphical models of relationships between concepts. A  
711 concept map diagram contains two basic elements: blocks, circles, or ovals representing  
712 concepts (with the concept named by text inside the symbol), and arrows connecting  
713 concept symbols to indicate relationships (relationships are named by text on or near the  
714 arrow.)

715 In modeling exchange documents, the blocks, circles, or ovals represent data structures,  
716 and the arrows represent relationships between data structures. Arrows can be used to  
717 reflect inheritance (type-of or "is-a") relationships as well as containment, composition,  
718 and aggregation relationships. Cardinality of relationships such as one-to-one (1:1) and  
719 one-to-many (1:M) can also be included on the arrows.

720 In the justice community, concept map diagrams have been used to represent GJXDM  
721 concepts and their relationships in developer training workshops and several exchange  
722 document development projects.

723 The concept map or high level "*business* component model" **MAY** be used to facilitate the  
724 dialog between business and technical staff to determine the elements and relationships

725 of a data exchange. It is used to assist in the determination of the elements in the  
726 GJXDM dictionary that are to be used and provides technical staff building the schema  
727 with basic requirements that need to be reflected in the schema. It is a very good starting  
728 point easily understood by subject matter experts. A GIEPD MAY include this high-level  
729 model. Models SHOULD use diagramming techniques that are readily converted to UML  
730 diagramming notation. The concept map is a good starting point for developing the UML  
731 or UML equivalent detailed graphical representation of the domain model.

732 The main advantage of concept map diagrams is that the notation is very simple, with  
733 only two kinds of symbols (concepts and relationships.) The main disadvantage of  
734 concept map diagrams is that, in their simplicity, they lack some of the descriptive power  
735 of more formal notations like UML class diagrams. For instance, concept maps are  
736 unable to depict individual data elements within data structures (data elements must be  
737 handled as concepts in their own right.) For large and complex exchange documents,  
738 diagrams drawn with concept maps can become unwieldy. In addition, there is no  
739 normative, formal standard for representing concepts like inheritance and cardinality that  
740 are important to object-oriented domain modeling and that can be reflected directly in  
741 GJXDM-conformant XML Schemas. Finally, concept map diagrams are generally not  
742 interchangeable between modeling tools (there is no equivalent interchange standard for  
743 concept maps to correspond to XMI for UML diagrams.) A guide for formulating concept  
744 map diagrams can be found in the Non-Normative References [**CONCEPT Maps**].

745

746 **Class Diagram** - The Unified Modeling Language (UML) is a formal industry-standard  
747 specification for modeling notations. It is managed by the Object Management Group  
748 (OMG), an open industry standards body. It is a broad modeling specification, covering  
749 several types of diagrams and other notations. This specification refers only to UML  
750 Static Structure diagrams, also called Class diagrams.

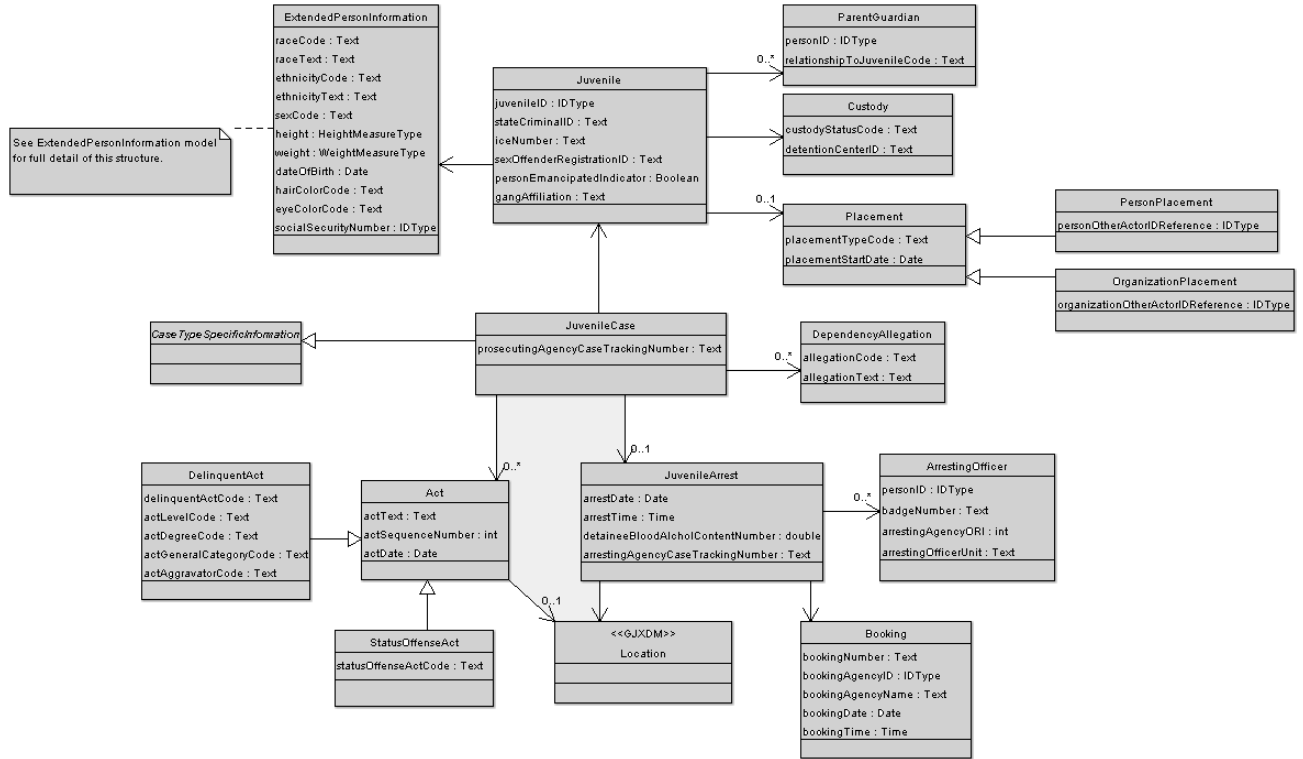
751 Class diagrams are similar to concept maps, but differ in that class diagrams offer more  
752 formal notation for representing concepts and relationships between them. Class  
753 diagrams describe the types of elements in a data exchange and the various kinds of  
754 static relationships that exist among them. Class diagrams also show the attributes of  
755 elements and the constraints that apply to the way elements are connected. Class  
756 diagrams provide a normative, formal standard for depicting individual data elements  
757 within class structures as well as representing concepts like inheritance, cardinality and  
758 constraints. Class diagrams provide for interchangeability between modeling tools. Class  
759 diagram structure and detail facilitate automation of XML schema generation.

760

761 For complete descriptions of UML class diagrams, see [**UML**] in the Normative  
762 References section.

763

764 A GIEPD MUST include a detailed graphical representation of the domain model of the  
765 business data and relationships. Following is an [Example] *Juvenile Case Filing* domain  
766 model graphical representation from the OASIS LegalXML Electronic Court Filing 3.0  
767 [**LegalXML ECF**] specification:



768

769 UML Class Diagrams are STRONGLY RECOMMENDED as a consistent method for graphically  
 770 depicting the domain model. They MAY be a generated artifact created from some other  
 771 representational source. A UML Class Diagram MAY NOT include non-standard UML symbols  
 772 and notations. Additionally, the mapping of elements to the GJXDM dictionary MAY NOT rely on  
 773 any Domain Model concept or element that cannot be represented in standard UML notation.

774 UML Class Diagrams MUST include an XML Metadata Interchange (XMI) formatted file as well as  
 775 an image file in a common open standard format such as jpeg, gif, png or svg. These files MUST  
 776 contain only pure UML.

777 Alternate representations MAY be used to show the business data and relationships, provided the  
 778 domain model is also supplied in one of the above approved formats (optional Concept Map and  
 779 UML Class Diagram or equivalent). Proprietary concepts MAY be included in this artifact as long  
 780 as the actual mapping results are not dependent on the concepts. The mapping MUST be  
 781 producible from the above approved formats alone.

782 If the domain model uses a notation that is different from UML, the domain diagram MUST be  
 783 semantically equivalent to a UML domain model.

784

- 785 • **Source documents or requirements specifications.**

786

787 A GIEPD MAY include source documents or supplemental requirement specifications  
 788 that help describe the data exchange. These materials may be actual paper documents,  
 789 or some electronic equivalent. As GIEP exchanges are not limited to existing paper  
 790 exchanges, there may not be existing documentation. If used, this documentation  
 791 SHOULD be included in the GIEPD to help ensure adequate coverage of existing  
 792 exchanges.

793

- 794 • **Spreadsheet of business data to GJXDM component mapping**

795

796 A GIEPD MUST contain a spreadsheet mapping business data requirements to GJXDM  
797 components. Note that a spreadsheet may be a generated artifact and not necessarily the  
798 instrument used to generate the requirements and their mappings to the GJXDM. The  
799 purpose of the spreadsheet is to make the business data requirements and mappings  
800 available in a universally readable format.

801

802 • **Subset Schema Generation Tool (SSGT) Want Lists**

803

804 A GIEPD MUST contain the appropriate SSGT Want List used to create its schemas.  
805 GJXDM schemas can be created by means other than the SSGT. However, in order to  
806 maintain

807

808 **3.2.1 Graphical Domain Model Documentation Rules**

809 **3.2.1.1 Graphical Modeling Notations**

810

811 The justice community has learned through experience that there is not a single domain modeling  
812 notation that satisfies the needs of all workgroups. The skills and experience of the facilitator and  
813 the availability of modeling tools are the most significant factors in determining which notation is  
814 appropriate for a workgroup.

815 At the same time, reusability of exchange document packages is improved if the range of  
816 modeling notations in use is kept to a minimum. The intent of this section is to balance these  
817 factors.

818 The modeling notations identified in this section have been chosen because they satisfy the  
819 following key criteria that are important to artifact reusability:

820 They are supported by tools supplied by multiple vendors, some of which are freely  
821 available and/or ubiquitous

822 They are familiar to a wide range of facilitator/analysts, and are well-described in publicly  
823 available documentation and training materials

824 They are accessible to the range of stakeholders involved in a typical exchange  
825 document development project, including technologists and business subject-matter  
826 experts (it is recognized that each notation may require a minimal amount of explanation  
827 and coaching to be accessible to newcomers)

828

829 **3.2.1.2 Graphical Domain Model Artifacts**

830

831 A concept map diagram MAY be utilized to initiate the data exchange dialog between  
832 workgroup members

833

834 A UML static structure (class) diagram OR alternate diagram that provides functionality  
835 equivalent to a UML diagram MUST be used

836

837 Graphical domain model artifacts, such as UML diagrams or informal diagrams, MUST be  
838 presented in either JPEG or PNG format.

839

840 UML models MUST be presented in XML Metadata Interchange (XMI) format, version 1.2  
841 or higher.

842

843 Domain model diagrams MAY BE presented in proprietary formats (i.e., formats that  
844 require processing by commercial tools), as long as they are also presented in  
845 accordance with the other rules in this section.

846

847 A domain model SHOULD be a UML static structure (class) diagram.

848

### 849 **3.2.2 Textual Domain Model**

850 A textual model is used to describe business data elements and their GJXDM mapping used in  
851 the information exchange. The textual model notation involves listing elements in the exchange  
852 document in a spreadsheet format. (This specification is neutral on the spreadsheet used, though  
853 it is expected that in many cases Microsoft Excel will be used due to its ubiquity.) Data elements  
854 are grouped into subject areas (or data structures). A definition is provided for each data  
855 element. Common business names for elements are listed with business oriented definitions that  
856 are sufficiently detailed to map to the GJXDM Dictionary. As mapping to the GJXDM Dictionary  
857 occurs, the GJXDM name, definition and GJXDM path are captured on the spreadsheet.

858

859 The business terms and definitions are used to determine if there is a GJXDM Dictionary entry.  
860 Terms in the dictionary must exactly match the business definitions for elements. If they do not  
861 match (close does not count), then an extension to the GJXDM must be defined.

862

863 The model MUST be in a tabular or spreadsheet format (that is, a format with cells, rows, and  
864 columns)

865

866 The mapping of the Textual Domain Model to GJXDM SHOULD BE recorded in a Microsoft Excel  
867 Spreadsheet.

868

869 The mapping MAY BE recorded within the a UML domain model , as long as the GJXDM  
870 mapping information can be presented along with the domain model structure in XML format.

871

872 The model MUST include grouping of data elements into data structures and the first column of  
873 the tabular or spreadsheet format MUST indicate the data structure / group and the second  
874 column MUST indicate the data element.

875

876 The column immediately to the right of the data element column MUST indicate the business  
877 definition of the data element.

878

879 The sort order for the mandatory and optional fields included on the tabular or spreadsheet format  
880 can use any approach that can be easily understood.

881

882 A key advantage of a textual model is that it is a simple notation, involving only textual elements.  
883 The chief disadvantage of a textual model is that it does not offer graphical elements for  
884 representing important concepts, like relationships between data structures or sharing of common  
885 data structures from multiple places. Optionally describing these concepts in a "notes" section  
886 MAY be done as a supplement to information contained on the Domain Model.

887

888 The information contained in the Textual Domain Model together with Graphical Domain Model  
889 detail provides the GIEPD requirements needed to build XML schema.

890

### 891 3.2.3 Textual Domain Model Documentation Rules

892 The following Textual Domain Model mapping table describes the name, definition, notes, and  
893 required or optional status of each item contained in the mapping documentation table.

894

[MAP1]	All Domain Model and GJXDM Mapping Documentation MUST include the required content identified in the GJXDM Domain Model Mapping table. The table MAY contain the specified optional content.
--------	--

895

896

<b>GJXDM TEXTUAL DOMAIN MODEL MAPPING STANDARD</b>			
Name	Definition	Notes	Required or Optional
Cardinality	Defines the minimum and maximum allowed occurrences for the element; left blank for class and relationship entries.	Examples of UML syntax follow below: 0..0 (element must not be used) 0..1 (optional element may only be used once) 0..n (optional element may be used up to n number of times) 0..* (optional element that may be used an unlimited number of times) 1..1 (required element that must be used once and only once) 1..n (required element that must be used at least once and may be used up to n number of times) 1..* (required element that must be used at least once may be used an unlimited number of times).	Required
Extension Inheritance	Documents the base type and type extensions of the GJXDM element(s) that were used to build the local extension, if any. .	This field is left blank for GJXDM content.	Required
Functional Element	The definition of the functional		Required



**GJXDM TEXTUAL DOMAIN MODEL MAPPING STANDARD**

Name	Definition	Notes	Required or Optional
Description	(business) use and meaning of the data element.		
Functional Element Name	The functional, or business, element name.	Use the name of the related class (as defined by an association, aggregation, or composition relationship) if this element describes a domain model relationship	Required
GJXDM Path	Describes the full GJXDM schema path for the class, property, or relationship.	Namespace prefixes must be used to identify extensions.  The path statement may include required attributes.  For classes, this is just the type name. For the other content, this is the full path statement.	Required
Class Name	The name of the class or category that contains the element.	Depending on the modeling approach, this could be the UML class as defined in the domain model, or a more general category, such as "arrest information".	Required
Code Set Source	The source, if any, of the codes assigned to this element.		Optional
Data Type	Defines the kind of information stored in the data element and describes how the field is formatted.		Optional
Examples	A sample value that illustrates how the element would be populated (e.g. "123" or "Accounts Payable".)		Optional
Exchange Notes	Contains any additional information about exchange-specific mandatory and optional business rules and describes any recent changes made to the element.		Optional
Extension	Indicates that the element requires an extension to the GJXDM.	This field is left blank for GJXDM content	Optional

<b>GJXDM TEXTUAL DOMAIN MODEL MAPPING STANDARD</b>			
Name	Definition	Notes	Required or Optional
Field Length	The maximum field length, if available.		Optional
GJXDM Element Description	The complete GJXDM element description as defined in the GJXDM schemas.	Providing the GJXDM description along with the functional element description allows positive verification of the accuracy of the GJXDM mapping.	Required
GJXDM Element Name	The name of the equivalent GJXDM-conformant element.		Required
GJXDM Mapping Notes	Provides additional information about mapping decisions and explanations about which objects and/or elements were assigned to local extension.	May be used to document required attributes.	Optional
ID	A unique, non-changing number assigned to each row in the spreadsheet when it is added	This information would be useful if the sample instance document includes pointers to the domain model content.	Optional
Relationship	Used to indicate whether or not the element represents a relationship, so that a decision about whether to represent them through inclusion or references can be made later.		Optional
Version	Documents the version of the specification in which this element was incorporated or updated.		Optional

897

898 Sample GIEPD spreadsheets can be found in Appendix D – Sample GIEPD(s)

899

### 900 **3.3 GIEPD Schemas**

901 XML Schemas are the mechanism used to define GIEPs for the exchange(s) defined by the  
 902 GIEPD. There are four different Schemas involved in a GIEPD. A supplemental discussion of the  
 903 role of each type of schema can be found under **[GJXDM IEPD]** in the Non-Normative  
 904 References section.

905

### 906 **3.3.1 Document Schema**

907 The GIEPD MUST include one-and-only-one document schema.

908

909 The document schema defines the root element of a GIEP and specifies the schema(s) to be  
910 used to validate the structure and contents of an XML Instance Document.

911

### 912 **3.3.2 Subset Schema**

913 The GIEPD MUST include one-and-only-one subset schema.

914 The Subset Schema removes unused GJXDM elements and type definitions. The Subset  
915 Schema narrows the focus on what information is actually being exchanged while also reducing  
916 the processing load required by validation. The Subset Schema is imported into the exchange via  
917 the Document Schema. Any instance document that validates against the Subset Schema MUST  
918 also validate against the GJXDM model as a whole.

919

### 920 **3.3.3 Constraint Schema**

921 If a constraint schema was developed or used, it MUST be included in the GIEPD.

922

923 The Constraint Schema embeds local constraints into GJXDM definitions. Constraint Schemas  
924 provide a variety of functionality. They provide a recommended place for enforcing cardinality  
925 rules. They also provide a means to enforce business rules that cannot be represented within the  
926 limitations of a Subset Schema.

927 When used solely to enforce cardinality, Constraint Schemas take the place of the Subset  
928 Schema for validation purposes. Enforcing cardinality is the more common use of a Constraint  
929 Schema.

930 When used to enforce business rules that go beyond the capacity of a Subset Schema, then  
931 Instance Documents are validated against both the Subset Schema (or data model as a whole)  
932 and the Constraint Schema in parallel. The Constraint Schema temporarily “stands-in” for the  
933 Subset Schema. Validation against such a Constraint Schema will only validate the additional  
934 business rules, not GJXDM conformance itself. For an Instance Document to be valid, it MUST  
935 validate against both the Subset Schema (or data model as a whole) and the Constraint Schema.

936

937 In addition, alternate constraint methods MAY be used, where the schema itself is more general  
938 and value constraints are applied using different XML methodologies (such as Schematron).

939

940 No new elements are permitted to be defined within the GJXDM constraint schema.

941

### 942 **3.3.4 Extension Schema**

943 If an extension schema(s) are developed, they MUST be included in the GIEPD.

944

945 The Extension Schema defines local extensions. While large, the data model can not include  
946 every element and type that could possibly be used in the context of an exchange of justice  
947 information. Some types of information are specific to particular jurisdictions or areas. It would be  
948 inappropriate and impractical for these local types of information to be contained in a national-  
949 scope data model.

950 The Extension Schema provides a means for this type of local information to be included in  
951 exchanges within the local area. Extension Schemas are imported into the exchange via the  
952 Document Schema.

953

## 954 **3.4 GIEPD Instance Documents**

955 [Sample instances] should include samples of both simple and complex information exchanges.  
956 Realistic data should be used (although data should be “sanitized” to omit actual identifying  
957 information that would violate privacy).

958 Instance documents are XML documents containing actual, although usually fictional, data. They  
959 are the payload, or “Package,” in an information exchange. They often show the selected  
960 elements of a Subset Schema more clearly than the schema itself, especially to a non-technical  
961 audience.

962 Instance Documents MAY show the minimal number of elements required to validate against the  
963 collected Schemas. Instance Documents MAY also show elements that are optional in the  
964 collected Schemas. It is RECOMMENDED that a GIEPD contain both minimal Instance  
965 Documents and more fully populated examples.

966 Instance Documents MUST validate though the schemas that are part of the Information  
967 Exchange Package Documentation.

968

### 969 **3.4.1 Sample Source Documents & XML Instance Documents**

970 Sample real world documents are a valuable tool to help others understand the actual data  
971 exchanged. Much of the technical material present in a GIEPD is difficult to read and understand.  
972 Sample source documents can clarify the scope and details of a GIEPD.

973 A GIEPD MAY contain one or more real world business documents representing component(s) of  
974 the GIEPD

975

#### 976 **3.4.1.1 Sample Instance Documents**

977 A GIEPD MUST contain at least one sample XML instance document.

978

979 [Sample instances] SHOULD include samples of both simple and complex information  
980 exchanges. Realistic data should be used (although data should be “sanitized” to omit actual  
981 identifying information that would violate privacy).

982 Instance documents are XML documents containing actual, although usually fictional, data. They  
983 are the payload, or “Package,” in an information exchange. They often show the selected  
984 elements of a Subset Schema more clearly than the schema itself, especially to a non-technical  
985 audience.

986 Instance Documents MAY show the minimal number of elements required to validate against the  
987 collected Schemas. Instance Documents MAY also show elements that are optional in the  
988 collected Schemas. It is RECOMMENDED that a GIEPD contain both minimal Instance  
989 Documents and more fully populated examples.

990 Instance Documents MUST validate though the schemas that are part of the Information  
991 Exchange Package Documentation.

992

#### 993 **3.4.1.2 Sample Style Sheets and Styled Documents**

994 A GIEPD MAY contain sample style sheets and resulting styled documents.

995 Even an XML instance document can be confusing for a layperson to read and understand. If the  
996 GIEPD was based on existing paper documents, it may also be confusing as to whether the  
997 resulting GIEPD meets the same requirements as the original paper documents.

998 Sample style sheets and the resulting styled documents can clarify the connections between the  
999 XML Instance Documents and existing exchanges. If the GIEPD is based on existing exchanges,  
1000 then sample style sheets and the resulting styled documents SHOULD be included.

1001 Additionally, style sheets may be used to transform XML instance documents into other variations  
1002 of XML or into other data formats entirely.

1003 Where included, style sheets SHOULD use existing styling technologies and produce freely  
1004 consumable output formats, such as HTML and PDF.

1005

### 1006 **3.5 Other Supporting Documentation**

1007 A wide and growing variety of tools is available for creating GIEPDs. Any supporting  
1008 documentation that would be useful to others for further use and refinement of the GIEPD  
1009 SHOULD be included in the GIEPD.

1010

### 1011 **3.6 Packaging of GIEPD Artifacts (i.e., zip) and Naming rules**

1012 This MNDR specifies File and Folder Naming rules for producing a GIEPD as follows:

1013

[NMS16]	<p>GIEPD artifacts MUST be combined into a ZIP file.</p> <p>The ZIP file MUST have the filename: &lt;IEP name&gt;-&lt;IEP version&gt;.zip</p> <p>Where &lt;IEP name&gt; &amp; &lt;IEP version&gt; = &lt;IEP name&gt; &amp; &lt;IEP version&gt; in the IEP document namespace.</p> <p>For example, if IEP GJXDM xmlns = <a href="http://www.myDomainName.com/Citation/1.0/document">http://www.myDomainName.com/Citation/1.0/document</a> , THEN the GIEPD ZIP filename would be:</p> <p>Citation-1.0.zip</p> <p>represents an GIEPD ZIP file for version 1.0 of a Citation IEP where &lt;IEP name&gt; = "Citation" and &lt;IEP version&gt; = "1.0"</p>
---------	--

1014

[NMS17]	<p>The GIEPD ZIP file MUST have the following structure and contents.</p> <p>The root directory in the ZIP archive must contain the following files and directories:</p>
---------	--

- The GIEPD Overview document file, in a suitable file format, and named “GIEPD Overview” with a file extension corresponding to the format. The “GIEPD Overview” MUST include the filename of the GIEPD ZIP file.
- A directory named “domain model artifacts”
- A directory named “mapping artifacts”
- A directory named “schemas”
- A directory named “sample instances”

The directory named “domain model artifacts” MUST contain all artifacts related to the domain model of the IEP, as discussed in section 3 of this specification.

The directory named “mapping artifacts” MUST contain all artifacts related to the mapping of the domain model to GJXDM, as discussed in section 3 of this specification.

The directory named “sample instances” MUST contain one or more sample XML instances that are valid against the document, extension, constraint, and subset schemas in the IEP.

Each sample XML instance:

- 1) MUST associate referenced IEP namespaces by using the `xsi:schemaLocation` attribute on the XML Instance root element;
- 2) MUST use the `xsi:schemaLocation` attribute
- 3) MUST use a relative URL, valid within the IEP structure documented here, to locate the schema for each namespace.

The directory named “schemas” MUST contain the following:

- A document schema file (see rule NMS12 and NMS13 for filename rules)
- Extension schema files(s) (see rule NMS10 and NMS11 for filename rules)
- A constraint schema file, if the IEP uses a constraint schema (see rule NMS14 and NMS15 for filename rule)
- A directory named “subset” that contains the subset schema set; the sub-directory structure underneath the “subset” directory must match the directory structure of the GJXDM distribution version being used. *Note that the subset schema is not a single schema; rather, it is a directory structure that contains many related schemas.*

	To the extent that unzipped schemas in the IEP import each other, the schemaLocation attribute for each schema's xsd:import element(s) MUST use a <u>relative</u> URL to locate the imported schema. The relative URL MUST be valid within the structure of the ZIP file specified above.
--	---

1015

1016 A sample GIEPD ZIP file can be found in Appendix D – Sample GIEPD(s)

1017

---

1018 **4 Schema Set and Instance Document Naming**  
1019 **and Design Rules**

1020 This section provides detailed Naming & Design rules for developing a GJXDM conformant  
1021 GIEPD. Included in this section are Naming & Design standards for GIEPD XML schemas,  
1022 including XML types, elements, attributes, permitted and non-permitted XSD methods and  
1023 documentation standards. This section also describes the rules for constructing XML Instance  
1024 documents, including requirements for root elements and validation methods.

1025

1026 **4.1 General Schema Set Naming and Design Rules**

1027 Schema language provides many redundant features that allow a developer to represent a logical  
1028 data model many different ways. Heterogeneous data models can become an interoperability  
1029 problem in the absence of a comprehensive set of naming, definition, and declaration design  
1030 rules.

1031 This subsection establishes rules for XML schema elements, attributes, and type creation.  
1032 Because the W3C XML specifications are flexible, comprehensive rules are needed to achieve a  
1033 balance between establishing uniform schema design while still providing developers flexibility  
1034 across the Justice and Public Safety domain.

1035 Adherence to these rules will ensure that semantics are unambiguous, enabling the practitioner  
1036 teams to conduct straightforward comparisons and make recommendations with respect to  
1037 enterprise reusability across their respective organizations. GJXDM information exchange  
1038 schema(s) and XML Instances rules are modeled after the same naming and design conventions  
1039 used to develop the Global Justice XML Data Model (GJXDM).

1040

1041 **4.1.1 General Naming Rules**

1042 The W3C XML Schema Definition Language has become the generally accepted schema  
1043 language that is experiencing the most widespread adoption. Although other schema  
1044 languages exist that offer their own advantages and disadvantages, DOJ-Global has determined  
1045 that the best approach for developing a national XML exchange standard is to base its work on  
1046 W3C XSD.

1047

[STA1]	All GJXDM information exchange schema design rules MUST be based on the W3C XML Schema 1.0 Recommendations: XML Schema Part 1: Structures and XML Schema Part 2: Datatypes.
--------	---

1048

1049 A W3C technical specification holding recommended status represents consensus within  
1050 the W3C and has the W3C Director's stamp of approval. Recommendations are  
1051 appropriate for widespread deployment and promote W3C's mission. Before the Director  
1052 approves a recommendation, it must show an alignment with the W3C architecture. By  
1053 aligning with W3C specifications holding recommended status, DOJ-Global can ensure that its  
1054 products and deliverables are well suited for use by the widest possible audience with the best  
1055 availability of common support tools.



1056

[STA2]	All GJXDM information exchange schema and payloads MUST be based on the W3C suite of technical specifications holding recommended or higher status.
--------	---

1057

1058 The English language has many spelling variations for the same word. For example, American  
 1059 English “program” has a corresponding British spelling “programme.” This variation has the  
 1060 potential to cause interoperability problems when exchanging XML components because of the  
 1061 different names used by the same elements. Providing a dictionary standard for spelling will  
 1062 mitigate this potential interoperability issue.

1063

[GNR1]	User-defined information exchange XML elements, attributes and type names MUST be composed in the English language, using the primary English spellings provided in the Webster’s English Dictionary.
--------	---

1064 The ebXML Core Component Technical Specifications (CCTS) provides a rule set for precisely  
 1065 defining the semantics of a data element in terms of a tripartite naming convention specified by  
 1066 ISO 11179 Part 5 (object class, [qualifiers], property term, and representation term). The three  
 1067 parts are combined to form the names of GJXDM information exchange XML elements, complex  
 1068 types, and attributes.

1069

[GNR2]	User-defined information exchange XML element, attribute and type names MUST be ebXML CCTS ISO 11179 compliant
[GNR3]	User-defined information exchange XML element, attribute and type names MUST NOT include spaces, other separators, or characters not allowed by W3C XML 1.0 for XML names.

1070

1071 Element names and the derivative names MUST be consistent with the GJXDM, even when this  
 1072 would result in conflicts with GNR1 or GNR2.

1073

[GNR9]	GJXDM element names, attributes and type names MUST not be modified, even when GJXDM names conflict with rules GNR1 – GNR3. For example, the use of periods in GJXDM conflicts with GNR3.
--------	---

1074

1075 Acronyms and abbreviations impact semantic interoperability and are to be  
 1076 avoided to the maximum extent practicable. Since some abbreviations will inevitably be  
 1077 necessary, GJXDM maintains a normative list of authorized acronyms and abbreviations.

1078

1079 Appendix B provides the current list of permissible acronyms, abbreviations and word  
 1080 truncations. The intent of this restriction is to facilitate the use of common semantics and to foster  
 1081 greater understanding. Appendix B is a living document and will be updated by the Global XSTF  
 1082 task force to reflect growing requirements.

1083

[GNR4]	GJXDM information exchange XML element, attribute, and simple and complex type names MUST NOT use acronyms, abbreviations, or other word truncations, except those in the list of exceptions published in Appendix B.
[GNR5]	The acronyms and abbreviations listed in Appendix B MUST always be used.

1084

1085 Generally speaking, the names for GJXDM information exchange XML constructs must always be  
1086 singular. The only exception permissible is where the concept itself is pluralized.

1087

[GNR6]	GJXDM information exchange XML element, attribute and type names MUST be in singular form unless the concept itself is plural.
--------	--

1088

1089 Example:

1090

```
1091 PersonPhysicalFeature, PhysicalFeatureType  
1092 PersonPhysicalDetails, PersonPhysicalDetailsType  
1093 personNameInitialIndicator
```

1094

1095 XML is case sensitive. Consistency in the use of case for a specific XML component (element,  
1096 attribute, type) is essential to ensure every occurrence of a component is treated  
1097 the same. This is especially true in a business-based data-centric environment such as  
1098 that addressed by GJXDM. Additionally, the use of visualization mechanisms such  
1099 as capitalization techniques assist in ease of readability and ensure consistency in  
1100 application and semantic clarity. The ebXML architecture document specifies a standard  
1101 use of upper and lower case for expressing XML elements and attributes  
1102 respectively. GJXDM adheres to the ebXML standard. Specifically, GJXDM element and  
1103 type names will be in UpperCamelCase (UCC).

1104

[GNR7]	The UpperCamelCase (UCC) convention MUST be used for naming elements and types.
--------	---

1105

1106 UpperCamelCase Example:

1107

{Capitalizes the first letter "P" and each sub-word "N"}

1109

```
1110 PersonName  
1111 JewelryStone
```

1112

1113 GJXDM information exchange attribute names will be in lowerCamelCase (LCC).

1114

[GNR8]	The lowerCamelCase (LCC) convention MUST be used for naming attributes.
--------	---

1115

1116 LowerCamelCase Example:

1117

```
1118 {lowercase used for the first letter "a" and upper-case for each sub-  
1119 word "C", "C", "L" ...}
```

1120

```
1121 amountCurrencyCodeListVersionID  
1122 characterSetCode
```

1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131

GJXDM instance documents are designed to effect data and document electronic exchanges. Including mixed content in exchange documents is undesirable because exchange transactions are based on exchange of discrete pieces of data that must be clearly unambiguous. The white space aspects of mixed content make processing unnecessarily difficult and add a layer of complexity not desirable in information exchanges.

[MDC1]	Mixed content MUST NOT be used except where contained in an xsd:documentation element.
--------	--

1132  
1133  
1134  
1135  
1136

**[Definition] Mixed Content**

An XML <element> that contains both “string data” and other <element> data is defined as having mixed content.

1137  
1138

[Example of Mixed Content]

1139  
1140  
1141

The XML element <book> has mixed content because it contains string data “The Three Musketeers” and <element> data which is <author>

1142  
1143  
1144  
1145

```
<book>
  The Three Musketeers
  <author> Alexander Dumas </author>
</book>
```

1146  
1147  
1148

[Example without Mixed Content]

1149  
1150  
1151

The XML element <book> has only <element> content namely <title> and <author> and therefore is NOT mixed content.

1152  
1153  
1154  
1155  
1156

```
<book>
  <title>The Three Musketeers</title>
  <author> Alexander Dumas </author>
</book>
```

1157  
1158  
1159  
1160  
1161

The features of W3C XML Schema allow for flexibility of use for many different and varied types of implementation. The GJXDM information exchange MNDR uses the following rules to allow for a more consistent use of these features:

[GXS4]	The root element in all GJXDM information exchange Schema modules MUST contain the following namespace declaration:  “xmlns:xsd=http://www.w3.org/2001/XMLSchema.”
--------	--

1162

1163 To avoid overloading implementation systems with unnecessary documentation, developers have  
1164 an option to create a schema without the documentation. This schema is for run-time and must  
1165 be a functional equivalent of the documented version.  
1166

[GXS5]	GJXDM information exchange schema developers MAY provide a run-time schema devoid of documentation in addition to the fully annotated version.
--------	--

1167

#### 1168 **4.1.1.1 Schema Built-in Simple Types**

1169 There are 44 simple types built into XML Schema. They are specified in Part 2 of the XML  
1170 Schema Recommendation. These built-in types were used in the designing of GJXDM schema.  
1171 Simple types are the concrete representations of the datatypes defined by ebXML Core  
1172 Components and specialized dataTypes defined by GJXDM referred to as proxied GJXDM simple  
1173 types j-xsd:, rather than using the xsd:schema simple types directly. Extensions to simple types  
1174 must use as their base the set of provided simple types defined in GJXDM.

1175

1176 The GJXDM Schema module incorporates XML Schema built-in types and fundamental CCTS  
1177 types. The GJXDM Schema module declares the built-in types to be used.

1178

[GXS6]	Any user defined types with simple content MUST be derived via extension or restriction on the proxied GJXDM simpleTypes defined in <b><a href="http://www.it.ojp.gov/jxdm/3.0/proxy/xsd/1.0 xsd.xsd">http://www.it.ojp.gov/jxdm/3.0/proxy/xsd/1.0 xsd.xsd</a></b>
--------	--

1179

#### 1180 **4.1.1.2 XSD:appinfo**

1181 The `xsd:appinfo` feature is used by schema to convey processing instructions to a  
1182 processing application, Stylesheet, or other tool. Some users have determined  
1183 that this technique poses a security risk and have employed techniques for stripping  
1184 `xsd:appinfo` from schemas. However, as GJXDM MNDR is committed to ensuring the  
1185 widest possible target audience, this feature may be used to convey non-normative information.  
1186 Non-normative information means non-standard.

1187

1188

[GXS14]	GJXDM designed schema MAY use <code>xsd:appinfo</code> . If used, <code>xsd:appinfo</code> MUST only be used to convey non-normative information.  Note: <code>appinfo</code> is a recent addition to GJXDM in Version 3.0.2
---------	--

1189

#### 1190 **4.1.2 Namespaces and Schema Locations**

1191

[NMS1]	Every GJXDM information exchange schema MUST have a namespace declared using the <code>xsd:targetNamespace</code> attribute.
--------	--

1192

1193

[NMS2]	Every GJXDM information exchange schema version MUST have its own unique namespace.
--------	---

1194

[NMS4]	GJXDM namespaces MUST only contain GJXDM conformant schema modules.
--------	---

1195

[NMS5]	<p>GJXDM published namespaces MUST never be changed. The namespace names for GJXDM reference schema releases are of the form:</p> <p><code>http://www.it.ojp.gov/jxdm/{major version . minor release . revision}</code></p> <p>For example the following namespace <code>http://.../jxdm/3.1.1</code> would be major-release 3 and minor-release 1.1 of the GJXDM schema.</p>
--------	---

1196

1197 **4.1.3 External Code List Rules**

[CDL1]	All Global JXDM Code Lists MUST be part of a Global JXDM or externally maintained Code List; they MUST NOT be included in a document or extension schema..
[CDL2]	The Global JXDM SHOULD identify and use external standardized code lists whenever practical rather than develop its own Global JXDM-native code lists.
[CDL3]	The Global JXDM information exchange developer, through extension/restriction, MAY design and use a “contextually” defined code list where an existing GJXDM code list needs to be extended, or where no suitable external code list exists.
[CDL4]	All GJXDM maintained or information exchange developer Code Lists MUST be enumerated using the GJXDM Code List Schema Module. (See <b>Global JXDM standards for NonStandardCodeType properties.</b> ) <b>TRUE???? Need to clarify the intent of NonStandardCodeType with the XSTF. (Tom Carlson will investigate).</b>
[CDL5]	The name of each GJXDM information exchange Code List Schema MUST be of the form: {Owning Organization}_{Code List Name}_{version number}.xsd ;
[CDL6]	An <code>xsd:import</code> element MUST be declared for every code list required in a GJXDM information exchange schema. Each codelist MUST be in its own namespace; the namespace identifier MUST be consistent with the same rules as extension schemas.
[CDL7]	When creating a local code list, an information exchange developer MUST follow the UBL code list schema and annotation rules. (need to coordinate this rule with CDL4. Does CDL4 mean GJXDM has a standard for user-defined codelists???) <a href="http://docs.oasis-open.org/ubl/cd-UBL-1.0/doc/cl/wd-ublclsc-codelist-">http://docs.oasis-open.org/ubl/cd-UBL-1.0/doc/cl/wd-ublclsc-codelist-</a>

	<a href="#">20040420.pdf</a>
[CDL8]	Users of the GJXDM MAY identify any subset they wish from an identified code list for their own trading community conformance requirements.
[CDL9]	The <code>xsd:schemaLocation</code> MUST include the complete URI used to identify the relevant code list schema.

1198

#### 1199 **4.1.4 General Type Definitions**

1200 Since GJXDM document and extension schema elements and types are intended to be reusable,  
 1201 all types must be named. This permits other types to establish elements that reference these  
 1202 types, and also supports the use of extensions for the purposes of versioning and customization.

1203

[GTD1]	All types MUST be named.
--------	--------------------------

1204

1205 Example:

```
1206 <xsd:complexType name="PersonType">
1207 ...
1208 </xsd:complexType>
```

1209

1210

#### 1211 **4.1.5 Complex Type Definitions**

1212 Since even simple datatypes are modeled as property sets in most cases, the XML expression of  
 1213 these models primarily employs `xsd:complexType`. To facilitate reuse,  
 1214 versioning, and customization, all complex types are named. In the GJXDM information exchange  
 1215 model, `xsd:complexType(s)` with complex content are considered classes(objects) .

1216

[CTD1]	For every class identified in GJXDM extension and document schema, a named <code>xsd:complexType</code> MUST be defined.
--------	--

1217

1218

1219 Example:

```
1220 <xsd:complexType name="BuildingType">
1221 ...
1222 </xsd:complexType>
```

1223

1224

1225

1226 GJXDM classes(objects) are defined in schema as named complexTypes. The sequence of  
 1227 elements contained within the complex type represent the properties of the class(object). These  
 1228 property elements are defined as global elements where each global element references a  
 1229 corresponding complex or simple type. GJXDM named complexTypes may be customized by  
 1230 creating a user-defined complexType. The user-defined complexType would utilize the

1231 xsd:extension base=GJXDMtype and add additional xsd:sequence  
1232 elements to extend the GJXDM class(object).  
1233

[CTD2]	Every GJXDM user-defined xsd:complexType definition content model MUST use the xsd:sequence element with appropriate global element references to reflect each property of its class. This does not preclude the use of xsd:choice (see <i>rule name GXS13</i> )
--------	--

1234  
1235  
1236 GJXDM Example:

```
1237 <xsd:complexType name="BailType">  
1238 <xsd:complexContent>  
1239 <xsd:extension base="ActivityType">  
1240 <xsd:sequence>  
1241 <xsd:element ref="BailSetAmountText" minOccurs="0" maxOccurs="unbounded"  
1242 />  
1243 <xsd:element ref="BailSetCourt" minOccurs="0" maxOccurs="unbounded" />  
1244 <xsd:element ref="BailSetCourtReference" minOccurs="0" maxOccurs =  
1245 "unbounded" />  
1246 ...  
1247 </xsd:sequence>  
1248 </xsd:extension>  
1249 </xsd:complexContent>
```

1250  
1251  
1252

There is a direct one-to-one relationship between ebXML CoreComponentTypes and GJXDM PrimaryRepresentationTerms. Additionally, there are several GJXDM SecondaryRepresentationTerms that are subsets of their parent GJXDM PrimaryRepresentationTerm. The total set of ISO 11179 Representation Terms by their nature represent GJXDM Datatypes. Specifically, for each GJXDM PrimaryRepresentationTerm or GJXDM SecondaryRepresentationTerm, an ebXML UnspecializedDatatype exists. In the GJXDM , these ebXML UnspecializedDatatypes are expressed as complex or simple types that correspond to an ebXML CoreComponentType.

1260  
1261

The set of valid GJXDM datatypes are based on ebXML Core Component Technical Specification v1.9 and include the following:

1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273

- 1) Amount
- 2) BinaryObject (secondary: Graphic,Picture,Sound,Video)
- 3) Code
- 4) \*DateTime (secondary: Date, Time)
- 5) Identifier (authorized abbreviation: ID)
- 6) Indicator
- 7) Measure
- 8) Numeric (secondary: Value, Rate, Percent)
- 9) Quantity

1274 10) Text (secondary: Name)  
1275  
1276 Reference:GTRI May 2004 Developer Workshop  
1277  
1278 \*note: DateTime element not in GJXDM but secondary Date and Time elements are included  
1279  
1280

[CTD3]	For every user-defined datatype based on the valid set of GJXDM datatypes, a named xsd:complexType or xsd:simpleType MUST be defined.
--------	---

1281

#### 1282 4.1.6 Complex Type Naming Rules

1283 GJXDM identifies naming rules for types, namely for complex types  
1284 based on Primary Representation Terms, Secondary Representation Terms and the ebXML Core  
1285 Component Types. Each of these complex and simple types are a fully  
1286 qualified type name based on ISO 11179. As such, these names convey explicit semantic  
1287 clarity with respect to the data being described. Accordingly, these naming standards  
1288 ensure that GJXDM xsd:complexType names are semantically unambiguous, and that there are  
1289 no duplications of GJXDM type names for different xsd:type constructs.

1290

1291 GJXDM xsd:complexType names follow general naming rules, and append the suffix  
1292 "Type" to denote a Type Name versus an Element Name.

1293

1294

[CTN1]	A user-defined xsd:complexType <b>name</b> MUST name the object suffixed by the word "Type". For example <xsd:complexType name="PersonType"> is correct And <xsd:complexType name="Person"> would be incorrect.
--------	---

1295

1296

#### 1297 4.1.7 Attribute Declarations

1298 Attributes are W3C Schema constructs associated with elements that provide further  
1299 information regarding elements. While elements can be thought of as containing data,  
1300 attributes can be thought of as containing metadata. Unlike elements, attributes cannot be nested  
1301 within each other—there are no "subattributes." Therefore, attributes cannot be extended as  
1302 elements can. Attribute order is not enforced by XML processors—that is, if the attribute order in  
1303 an XML instance document is different than the order in which the attributes are declared in the  
1304 schema to which the XML instance document conforms, no error will result. These limitations  
1305 dictate that GJXDM MNDR restrict the use of attributes to XSD built-in attributes, or to the  
1306 GJXDM SuperTypeMetadataAttributeGroup defined by GJXDM.



1307

1308 For a more complete discussion of the application of attributes within GJXDM, the reader should  
1309 reference the **[GJXDM USER]** link provided in the Non-Normative References Section.  
1310 Specifically, Part 3 “Metadata in the Global Justice XML Data Model” and Part 5 “XML Schema  
1311 Elements Versus Attributes” of the GJXDM Users Manual should be reviewed.

1312

[ATD1]	User defined attributes SHOULD NOT be used.
--------	---

1313

#### 1314 **4.1.7.1 Global Attributes**

1315 The current GJXDM has attributes that are common to all GJXDM and proxy Code List  
1316 elements. These common attributes have been declared using the  
1317 `xsd:globalattributegroup` element and utilizes the following rule. These rules are  
1318 included to ensure interoperability.  
1319

[ATD2]	If a Schema Expression contains one or more common attributes that apply to all elements contained or included or imported therein, the common attributes SHOULD be declared as part of a global attribute group. ( For an example about how to do this, see the Global JXDM global attribute group named “ <b>SuperTypeMetadata</b> ”)
--------	---

1320

[ATD3]	<p>For each Global JXDM user-defined element of <code>simpleType</code> and a <code>xsd:restriction</code> element;</p> <p>an <code>xsd:base</code> attribute MUST be declared and set to the appropriate GJXDM datatype.</p> <p>Note: the set of valid GJXDM datatypes are based on ebXML Core Component Technical Specification v1.9 and include the following 10 <code>simpleTypes</code>:</p> <ul style="list-style-type: none"><li>• Amount</li><li>• BinaryObject (secondary: Graphic, Picture,Sound,Video)</li><li>• Code</li><li>• DateTime (secondary: Date, Time)</li><li>• Identifier (authorized abbreviation: ID)</li><li>• Indicator</li><li>• Measure</li><li>• Numeric (secondary: Value, Rate, Percent)</li><li>• Quantity</li><li>• Text (secondary: Name)</li></ul>
--------	--

1321

1322

#### 4.1.7.2 XSD:nil

1323 You can indicate in a schema that an element may be nil in the instance document. Empty  
1324 content vs nil:

- 1325 • Empty: an element with an empty content is constrained to have no content.
- 1326 • nil: an instance document element may indicate no value is available by setting an  
1327 attribute - xsi:nil - equal to 'true'

1328

1329 Example:

1330

```

1331 XML Schema:
1332 <xsd:complexType name="PersonNameType">
1333   <xsd:complexContent>
1334     <xsd:extension base="j:SuperType">
1335       <xsd:sequence>
1336         ...
1337         <xsd:element ref="j:PersonGivenName" minOccurs="0"
1338 maxOccurs="unbounded" />
1339         ...
1340         <xsd:element ref="j:PersonMiddleName" minOccurs="0"
1341 maxOccurs="unbounded" />
1342         ...
1343         <xsd:element ref="j:PersonSurName" minOccurs="0"
1344 maxOccurs="unbounded" />
1345         ...
1346       </xsd:sequence>
1347     </xsd:extension>
1348   </xsd:complexContent>
1349 </xsd:complexType>
1350
1351 <xsd:element name="PersonName" type="j:PersonNameType" nillable="true" />
1352
1353 <xsd:element name="PersonGivenName" type="j:PersonNameTextType"
1354 nillable="true" />
1355
1356 <xsd:element name="PersonMiddleName" type="j:PersonNameTextType"
1357 nillable="true" />
1358
1359 <xsd:element name="PersonSurName" type="j:PersonNameTextType" nillable="true" />
1360
1361 XML instance document:
1362
1363 <PersonName>
1364   <PersonGivenName>Hannibal</PersonGivenName>
1365   <PersonMiddleName xsi:nil="true" />
1366   <PersonSurName>Lecter</PersonSurName>
1367 </PersonName>
1368
1369
1370

```

1371

[ATD5]	For local extensions based on GJXDM version 3.0.2 and above, the xsd built-in nillable attribute MUST be explicitly defined for any Global JXDM user-defined element which has simpleContent . The user-defined element must set nillable to <b>"true"</b> or <b>"false"</b> ;
--------	--

Note: An example from GJXDM v3.0.2

<xsd:element name="WitnessLocationDescriptionText" type="j:TextType" nillable="true">
---

1372 **4.1.7.3**

1373 **4.1.7.4 Empty Elements**

1374 Empty elements may cause application or XML processing difficulty and must be avoided

1375

[ELD5]	Empty elements MUST NOT be declared.
--------	--------------------------------------

1376

1377 **4.1.7.5 XSD:Any Element**

1378 GJXDM MNDR disallows the use of `xsd:any`, because this feature permits the introduction of  
1379 potentially unknown elements into an XML instance. GJXDM MNDR intends that all constructs  
1380 within the instance be described by the schemas describing that instance - `xsd:any` is seen as  
1381 working counter to the requirements of interoperability. In consequence, particular attention is  
1382 given to the need to enable meaningful validation of the document instances.

1383

1384 Were it not for this, `xsd:any` might have been allowed.

1385

[ELD7]	The <code>xsd:any</code> element MUST NOT be used.
--------	--

1386

1387 GJXDM information exchange schema disallows the use of `xsd:anyType`, because this  
1388 feature permits the introduction of potentially unknown types into an XML instance. The `<any>`  
1389 element enables the instance document author to extend his/her document with elements not  
1390 specified by the schema. GJXDM intends that all constructs within the instance be described by  
1391 the schemas describing that instance. Per UBL-NDR version 1.0.1, "`xsd:anyType` is seen  
1392 as working counter to the requirements of interoperability. In addition, use of `xsd:anyType` has  
1393 been identified as a significant security risk. In consequence, particular attention is given to the  
1394 need to enable meaningful validation of the document instances. Were it not for this,  
1395 `xsd:anyType` might have been allowed. "

1396

[GTD2]	The <code>xsd:anyType</code> MUST NOT be used.
--------	--

1397 **4.1.7.6**

1398 **4.1.7.7 XSD:anyAttribute**

1399 GJXDM information exchange schema disallows the use of `xsd:anyAttribute`, because this  
1400 feature permits the introduction of potentially unknown attributes into an XML instance. GJXDM

1401 information exchange packages intend that all constructs within the instance be described by the  
1402 schemas describing that instance - `xsd:anyAttribute` is seen as working counter to the  
1403 requirements of interoperability. In consequence, particular attention is given to the need to  
1404 enable meaningful validation of the GJXDM conformant document instances. (Also see rule  
1405 GTD2 on the `xsd:anyType` and ELD7 `xsd:any` element )

1406

[ATD6]	The <code>xsd:any</code> attribute MUST NOT be used.
--------	--

1407

## 1408 4.1.8 Element Declarations and Naming Rules

1409

### 1410 4.1.8.1 Elements Bound to Complex Types

1411 W3C XSD allows for any globally declared element to be the document root element. To keep  
1412 consistency in the instance documents and to adhere to the underlying process model that  
1413 supports each GJXDM information exchange Schema, it is desirable to have one and only one  
1414 element function as the root element. Since GJXDM follows a global element declaration scheme  
1415 (See Rule ELD2), each GJXDM Schema will identify one element declaration in each schema as  
1416 the document root element. This will be accomplished through an `xsd:annotation` child  
1417 element for that element in accordance with the following rule:

1418

[ELD1]	Each GJXDM <code>DocumentSchema</code> MUST identify one and only one global element declaration that defines the exchange document being conveyed in the Schema expression. That global element MUST include an <code>xsd:annotation</code> child element which MUST further contain an <code>xsd:documentation</code> child element that declares " <i>This element MUST be conveyed as the root element in any instance document based on this DocumentSchema.</i> "
--------	---

1419

1420

1421 [Definition] Document schema:

1422 The overarching schema within a specific namespace that conveys the business document  
1423 functionality of that namespace. The document schema declares a target namespace and is likely  
1424 to pull in additional schema by importing external schema modules. Each namespace will have  
1425 one, and only one, document schema.

1426 Example:

```
1427 <xsd:element name="Rapsheet" type="RapsheetType">  
1428 <xsd:annotation>  
1429 <xsd:documentation>This element MUST be conveyed as the root element in  
1430 any instance  
1431 document based on this Schema expression</xsd:documentation>  
1432 </xsd:annotation>  
1433 </xsd:element>  
1434
```

1435

1436 Global elements are declared as direct children of a root schema element. Global element  
1437 names, because they are globally reusable, express universally unambiguous semantics in their  
1438 names. By their nature, global elements can be reused consistently across the Justice & Public

1439 Safety enterprise XML domain, wherein every occurrence will have exactly the same meaning  
1440 and map to exactly the same authoritative source data. Therefore, the GJXDM information  
1441 exchange schema approach uses global elements in conjunction with global complex types.

1442

1443 Eliminating potential barriers to Justice XML system interoperability is a key driver for choosing  
1444 the global element approach. The advantages of this approach become more evident when  
1445 considering inter-agency XML interoperability and data exchange. Because the recommended  
1446 approach relies on global elements to carry unique semantics, there cannot be a duplicate  
1447 occurrence of elements with different characteristics in the GJXDM enterprise namespace. This  
1448 reduces the level of effort to analyze, map, and transform elements that are unique to a particular  
1449 functional area.

1450

[ELD2]	All element declarations MUST be global.
--------	--

1451

#### 1452 **4.1.8.2 Element Names for complexType(s)**

1453 For each class (complexType) defined in an extension, subset or document schema, a global  
1454 element name MUST be declared with the same name as the name of the corresponding  
1455 `xsd:complexType` to which it is bound, with the word "Type" removed.

1456

1457 Example:

1458 The `xsd:complexType` named `TelephoneNumberType` would require  
1459 a corresponding global element be declared. The the name of  
1460 this corresponding global element must be `TelephoneNumber`.

1461

1462

```
1463 <xsd:element name="TelephoneNumber" type="j:TelephoneNumberType"  
1464 nillable="true">  
1465  
1466 <xsd:complexType name="TelephoneNumberType">  
1467 ...  
1468 </xsd:complexType>
```

1469

1470

[ELD3]	For every class defined as a GJXDM user-defined types, a global element bound to the corresponding <code>xsd:complexType</code> MUST be declared.
--------	---

For example, a schema defining a `complexType` named `my:FavoritePersonType` would need to declare a global element named "FavoritePerson" of `objectType = my:FavoritePersonType` to bind a global element name to the `complexType`.

1471

1472

#### 1473 **4.1.8.3 ComplexTypes with SimpleContent**

1474 Elements bound to ebXML core-component types and GJXDM j-xsd: proxy built-in `xsd`  
1475 `dataTypes`.

1476

[ELD4]	For every user-defined simpleType, an xsd:restriction element MUST be declared
--------	--

1477

1478

1479

#### 4.1.8.4 Global Elements with simpleContent

1480 The GJXDM Global elements with simpleContent are reused in multiple  
1481 contexts. Their reuse in a specific context is typically identified in part through the use of  
1482 qualifiers. However, these qualifiers do not change the nature of the underlying concept  
1483 of the GJXDM core component that the element is derived from. As such, qualified  
1484 global elements with simpleContent are always bound to the same type as that of their  
1485 unqualified corresponding xsd: built-in datatype, j-xsd or GJXDM (eXML core component)  
1486 datatype.

1487

1488 Example:

1489

```
<xsd:element name="StreetNumberText" type="j:TextType"/>
```

1490

[ELD6]	Global simpleType elements declared with Qualified Properties must be of the same type as their corresponding Unqualified Property.
--------	---

1491

#### 4.1.9 Schema Documentation and Annotations

1492

1493

1494

##### 4.1.9.1 Embedded documentation

1495 The information about each GJXDM user-defined "Type" or "Element" must be documented in  
1496 schema. Fully annotated Schemas are valuable tools to implementers to assist in understanding  
1497 the nuances of the information contained therein. GJXDM information exchange schema  
1498 annotations will consist of a set of metadata elements.

1499

1500 The absence of an optional annotation inside the structured set of annotations in the  
1501 documentation element implies the use of the default value. For example, there are  
1502 several annotations relating to context such as `BusinessContext` or  
1503 `IndustryContext` whose absence implies that their value is "all contexts".

1504

1505 The following general documentation rule describes the documentation requirements for GJXDM  
1506 user-defined "Types" or "Elements".

1507

1508

[DOC1]	The xsd:documentation element for every GJXDM user-defined Element MUST contain a structured set of annotations in the following sequence and pattern: <ul style="list-style-type: none"><li>• Version (optional): An indication of the evolution over time of the</li></ul>
--------	--

	<p>Datatype.</p> <ul style="list-style-type: none"> <li>• Definition(mandatory): The semantic meaning of an Element</li> <li>• Cardinality(mandatory): Indication whether the complexType Element (Property) represents a not-applicable, optional, mandatory and/or repetitive characteristic of the parent complexType</li> <li>• AssociatedObjectClassQualifier (optional): Associated Object Class Qualifiers describe the 'context' of the relationship with another complexType object. That is, it is the role the contained Element plays within its association with the containing complexType object.</li> <li>• AssociatedObjectClass (mandatory); Associated Object Class is the Object Class at the other end of this association. It represents the Aggregate Business Information Entity contained by the Association Business Information Entity. (For example: the element PersonName within the complexType PersonType has Name as the AssociatedObjectClass contained in the Aggregate Business Information Entity called PersonType.)</li> <li>• AlternativeBusinessTerms (optional): Any synonym terms under which the Element is commonly known and used in the business.</li> <li>• Examples (optional): Examples of possible values for the Element</li> </ul> <p>Context Element should go here as well if not defined in a later NDR, so this is a placeholder for now. The list of Context Drivers is in the UBL ccts: core-component parameters schema. The current values for Context drivers are:</p> <ol style="list-style-type: none"> <li>1 Business Process</li> <li>2 Product Classification</li> <li>3 Geopolitical Region</li> <li>4 Official Constraint</li> <li>5 Business Process Role</li> <li>6 Supporting Role</li> <li>7 System Capabilities</li> </ol>
--	--

1509

1510 Example Extension Element Documentation:

1511

1512

1513

```

<xsd:annotation>
  <xsd:documentation>
    <Component>
      <ElementName>ChargeArrestReference</ElementName>
    <Version>1.0</Version>
    <Definition>A reference to the Arrest which resulted in the
    filing of a charge..</Definition>
    <Cardinality>0..1</Cardinality>
    <ObjectClass>Charge</ObjectClass>
    <PropertyTerm>Arrest</PropertyTerm>
    <RepresentationTerm>Reference</RepresentationTerm>
    <AssociatedObjectClass>Reference</AssociatedObjectClass>
  </Component>

```

1514

1515

1516

1517

1518

1519

1520

1521

1522

1523

1524

1525

1526  
1527  
1528  
1529  
1530

```
</xsd:documentation>  
</xsd:annotation
```

[NMS18]

Also, if adopted, the UBL core-components parameters schema (place where schema annotation/documentation elements are defined including context drivers) could be added back into this mndr. <http://docs.oasis-open.org/ubl/cd-UBL-1.0/xsd/common/UBL-CoreComponentParameters-1.0.xsd> Namespace could be referenced directly from extension schema.

1531

#### 1532 4.1.10 Schema Version Numbering Rules

1533 GJXDM user-defined namespaces are suffixed with a IEP name and IEP version. The  
1534 GJXDM MNDR has decided to include versioning information to immediately follow the IEP  
1535 name component of the namespace. The version information is divided into major and minor  
1536 fields. The minor field has an optional revision extension. For example, the namespace URI  
1537 for an IEP Charging Document schema has this form:

1538

```
1539 http://<IEP owner domain name>/<IEP name>/<IEP version>/<IEP schema  
1540 type[-suffix]>  
1541  
1542 note: where <schema Type> denotes document or extension
```

1543

1544 The major-version field is "1" for the first release of a namespace. Subsequent major  
1545 releases increment the value by 1.

1546

1547 For example, the first namespace URI for the first major release of the Charging document has  
1548 the form:

1549

```
1550 http://myNS.com/Complaint/1.0/document
```

1551

1552 The second major release will have a URI of the form:

1553

```
1554 http://myNS.com/Complaint/2.0/document
```

1555

1556 The distinguished value "0" (zero) is used in the minor-version position when defining a  
1557 new major version. In general, the namespace URI for every major release of the Complaint  
1558 domain has the form:

1559

1560 Example:

```
1561 http://myNS.com/<IEP name>/<major-number>.0[.<revision>]/<schema type>
```

1562



[VER1]	Every GJXDM information exchange schema and schema module <u>major version</u> <b>draft</b> MUST have a version number of the form:  <major>.0[.<revision>]
--------	---

1563

1564 When a major version reaches Standard status the [.<revision>] must not be present.

1565

1566

[VER2]	Every GJXDM Information exchange Schema and schema module <u>major version</u> <b>Standard</b> MUST have a version number of the form:  <major>.0
--------	---

1567

1568 For each document produced by the TC, the TC will determine the value of the <IEP name>  
1569 variable. In GJXDM MNDR, the major-version field of a namespace URI must be changed in a  
1570 release that breaks compatibility with the previous release of that namespace. If a change does  
1571 not break compatibility then only the minor version need change. Subsequent minor releases  
1572 begin with minor-version 1.

1573

1574 Example

1575

1576 The namespace URI for the first minor release of the Complaint document has this form:

1577

1578 `http://myNS.com/Complaint/1.1/document`

1579

[VER3]	Every <u>minor version</u> release of a GJXDM Information exchange schema or schema module <b>draft</b> MUST have a version number of the form:  <major >.<non-zero>[.<revision>]
--------	---

1580

1581 When a minor version reaches Standard status the [.<revision>] must not be present.

1582

[VER4]	Every <u>minor version</u> release of a GJXDM information exchange schema or schema module <b>Standard</b> MUST have a version number of the form:  <major >.<non-zero>
--------	---

1583

1584 Once a schema version is assigned a namespace, that schema version and that namespace  
1585 will be associated in perpetuity. Any change to any schema module mandates association  
1586 with a new namespace.

1587

[VER5]	For GJXDM information exchange schema <u>minor version</u> changes, the <IEP name> MUST NOT change.
--------	---

1588

1589

1590

1591 If a GJXDM schema namespace URI changes then any schema that imports the new version of  
1592 the namespace must also change (to update the namespace declaration). And since the  
1593 importing schema changes, its namespace URI in turn must change. The outcome is twofold:

1594

1595 • **There should never be ambiguity at the point of reference in a namespace**

1596 declaration or version identification. A dependent schema imports precisely

1597 the version of the namespace that is needed. The dependent schema never

1598 needs to account for the possibility that the imported namespace can change.

1599

1600 • **When a dependent schema is upgraded to import a new version of a schema,**

1601 the dependent schema's version (in its namespace URI) must change.

1602

1603 Version numbers are based on a logical progression. All major and minor version

1604 numbers will be based on positive integers. Version numbers always increment positively

1605 by one.

1606

[VER6]	For every GJXDM information exchange schema and schema module, the <u>major version</u> number MUST be a sequentially assigned, incremental number greater than zero.
--------	---

1607

1608

1609

[VER7]	For every GJXDM information exchange schema and schema module, the <u>minor version</u> number MUST be a sequentially assigned, incremental non-negative integer.
--------	---

1610

1611

1612 In keeping with rules NMS1 and NMS2, each schema minor version will be assigned a

1613 separate namespace. A minor revision (of a namespace) imports the schema module for the  
1614 previous version.

1615

1616 For instance, the document schema defining:

1617

1618 `http://myNS.com/Complaint/1.2/document`

1619

1620 will import the namespace:

1621

1622 `http://myNS.com/Complaint/1.1/document`

1623

1624 The `version 1.2` revision may define new complex types by extending or restricting  
1625 `version 1.1` types. It may define brand new complex types and elements. It must not use  
1626 the XSD `redefine` element to change the definition of a type or element in the `1.1` version.

1627

1628 The opportunity exists in the `version 1.2` revision to rename derived types. For  
1629 instance if `version 1.1` defines `Address` and `version 1.2` specializes `Address` it  
1630 would be possible to give the derived `Address` a new name, e.g. `NewAddress`. This is not  
1631 required since namespace qualification suffices to distinguish the two distinct types.

1632

1633 The minor revision may give a derived type a new name only if the semantics of the two  
1634 types are distinct.

1635

1636 For a particular namespace, the minor versions of a major version form a linearly-linked  
1637 family. The first minor version imports its parent major version. Each successive minor  
1638 version imports the schema module of the preceding minor version.

1639

1640 Example

1641

```
1642 http://myNS.com/Complaint/1.2/document
```

1643

1644 imports

1645

```
1646 http://myNS.com/Complaint/1.1/document
```

1647

1648 which imports

1649

```
1650 http://myNS.com/Complaint/1.0/document
```

1651

1652

1653 Ensuring semantic compatibility across minor versions is essential. Semantic compatibility in this  
1654 sense pertains to preserving the business function.

1655

[VER8]	GJXDM information exchange schema and schema module <u>minor version</u> changes MUST not break semantic compatibility with prior versions; nor may they break existing document instances that are based on any earlier minor version of the last major version. For example, an instance document build on a 1.1 minor version must be able to be processed by any later minor release, for example, a 1.9 version. Minor versions maintain forward compatibility.
--------	--

1656

1657 Major versions of schema do NOT necessarily have to maintain forward compatibility with the  
1658 previous major version.  
1659

[VER9]	GJXDM information exchange schema and schema module <u>major version</u> changes MAY break semantic and/or structural compatibility with prior versions. No backward compatibility is guaranteed.
--------	---

1660

## 1661 **4.1.11 Import versus Include**

1662

### 1663 **4.1.11.1 Schema Modularity**

1664 GJXDM MNDR supports modularity in schema design. The full GJXDM schema may be  
1665 modularized by creating multiple subset schemas from GJXDM.

1666

[SSM1]	GJXDM Schema MAY be split into a smaller subset schema, but only one GJXDM subset can be created for a given document schema, because the GJXDM schema must reside in one and only one namespace.
--------	---

1667

1668

1669 GJXDM based document schemas will be developed over time, each of which expresses a  
1670 separate business function for transaction data or a business document. The GJXDM MNDR  
1671 schema modularity approach is structured so that users can reuse individual document schemas  
1672 as is or with modification for local usage.

1673

1674 Additionally, a document schema can import individual schema modules without having to import  
1675 the entire GJXDM reference schema module. Each document schema will define its own  
1676 dependencies. The GJXDM MNDR schema modularity model ensures that logical associations  
1677 exist between document schema and a set of applicable imported schemas (GJXDM subset,  
1678 extension schema, codelists etc.) . The imported schemas could be reused in defining additional  
1679 document schemas supporting reuse to the maximum extent possible.

1680

1681 This is accomplished through the use of document schema and external schema modules.

1682 There are two types of schema in the GJXDM MNDR – document schema and schema modules  
1683 (subset, extension, CodeList(s)...) Document schemas are always in their own namespace.  
1684 External Schema modules are in separate namespace(s) such as the GJXDM proxy CodeList(s),  
1685 GJXDM subset(s) and any GJXDM extension schema(s). External schema modules are  
1686 conformant with W3C XSD.

1687

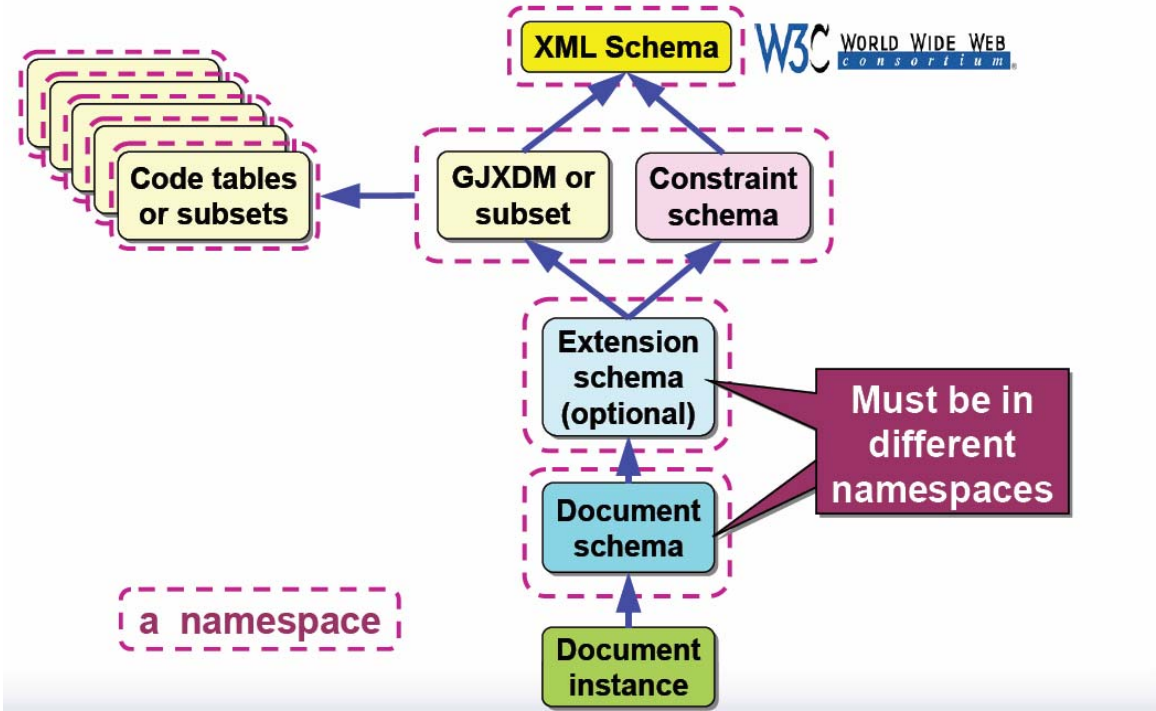
1688 A namespace is an indivisible grouping of types. A “piece” of a namespace can never be  
1689 used without all its pieces. GJXDM document schemas may have zero or more external schema  
1690 modules that they import. The document schema for a namespace then imports those  
1691 external modules.

1692

1693

1694 **[Definition] External schema module –**

1695 A schema module that is imported by another schema to expand the scope of schema applicable  
 1696 in validating a document instance.  
 1697  
 1698



1699  
 1700

1701 Figure 1.1 shows how the GJXDM MNDR namespace standard for segregating GJXDM schema  
 1702 from locally developed schema.

1703  
 1704 Any schema module may import other schemas from other namespaces.  
 1705

1706 **4.1.11.2 Module Conformance**

1707 GJXDM has defined a set of naming and design rules to ensure maximum interoperability and  
 1708 standardization.

[SSM2]	Imported schema modules MUST be fully conformant with GJXDM information exchange schema naming and design rules.
--------	--

1709

1710 **4.1.11.3 External Schema Modules**

1711 Developers will create schema modules which, as illustrated in Figure 1.1 be located in a  
 1712 separate namespace from the corresponding document schema.

1713

[SSM3]	GJXDM schema modules MUST be treated as external schema modules of the document schema.
--------	---

1714

1715  
1716  
1717  
1718  
1719  
1720

#### 4.1.11.4 Internal Schema Modules & xsd:include Statement

Internal schema modules do not declare a target namespace, but instead reside in the namespace of their parent schema. All internal schema modules are accessed using xsd:include. The MNDR does not support the use of Internal Schema Modules and therefore IEP's must not use the xsd:Include statements in schema.

[SSM4]	xsd:include MUST NOT be used in development of IEP's because this MNDR does not support use of Internal Schema modules.
--------	---

1721

1722

#### 4.1.11.5 External Schema Modules

1723 GJXDM is dedicated to maximizing reuse. As the complex types and global element  
1724 declarations will be reused in multiple schemas, a logical modularity approach is to  
1725 create GJXDM schema modules based on collections of reusable types and elements.  
1726

[SSM5]	GJXDM schema module(s) MAY be created for reusable components.
--------	--

1727

1728

1729 Developers will create external schema modules. These external schema modules will be based  
1730 on logical groupings of contents. The set of possible schema modules includes:

1731

1732

- GJXDM Reference Schema

1733

- GJXDM Subset Schema(s)

1734

- GJXDM Constraint Schema(s)

1735

- GJXDM Extension Schema(s)

1736

- GJXDM proxy Code List(s)

1737

- Non-GJXDM External Schema(s) and Code List(s)

1738

- If adopted, CCTS Core Component Parameters (for documentation/annotation of new schema elements)

1739

1740

1741

#### 4.1.12 Character encoding

1742

[IND3]	In conformance with ISO/IETF/ITU/UNCEFACT Memorandum of Understanding Management Group (MOUMG) Resolution 01/08 (MOU/MG01n83), all GJXDM XML SHOULD be expressed using UTF-8.
--------	---

1743

1744 **4.1.13 XSD:notation**

1745 XSD:notation is used to declare the format of non-XML data. A notation in XML is just like the  
1746 notation declarations in DTDs. The main difference is that W3C XML Schema notations are  
1747 namespace-aware and can be imported between schemas. When these declarations are used,  
1748 the notations are used in xsd:enumeration facets to create simple types.

1749 The notation datatype is used to declare links to external non-XML content (for example, image  
1750 data) and then associate that content with an external application that handles it.

1751 Notation is a built-in legacy simple type and are very seldom used in production applications, and  
1752 not optimal for the Justice and Public Safety community.

1753

[GXS10]	xsd:notations MUST NOT be used.
---------	---------------------------------

1754

1755 **4.1.14 XSD:all**

1756 Used within a group, xsd:all has the same meaning as when it is used directly under  
1757 xsd:complexType, except that there are no minOccurs and maxOccurs attributes and it cannot be  
1758 marked as optional. The xsd:all compositor requires occurrence indicators of minOccurs=0 and  
1759 maxOccurs=1. The xsd:all compositor allows for elements to occur in any order. The result is that  
1760 in an instance document, elements can occur in any order, are always optional, and never occur  
1761 more than once. Such restrictions are inconsistent with data-centric scenarios such as most of  
1762 the work in the Justice & Public Safety community.

1763 Another disadvantage of xsd:all is that it cannot be repeated any further. This limits the use of  
1764 xsd:all to the first occurrence of its set of elements. If a content model requires an element that  
1765 occurs more than once, then xsd:all cannot be used.

1766

[GXS11]	The xsd:all element MUST NOT be used.
---------	---------------------------------------

1767

1768 **4.1.15 XSD:choice**

1769 The xsd:choice compositor allows for any element declared inside it to occur in the  
1770 instance document, but only one. While xsd:choice is a very useful construct in situations  
1771 where customization and extensibility are not a concern, GJXDM MNDR recommends against  
1772 using xsd:choice because it cannot be extended. If extension is not a concern, then  
1773 xsd:choice may be used.

1774

[GXS13]	The xsd:choice element SHOULD NOT be used where customization and extensibility are a concern.
---------	--

1775 **4.2 Subset Schema naming and Design Rules**

1776 GJXDM subset schema(s) provide for limiting the full GJXDM set of object classes and set of  
1777 element(s) within each class down to a subset which is relevant for a specific Document Schema  
1778 or specific domain within a local jurisdiction.

1779

1780 **4.2.1 Rules for Conformant Subset Schemas**

1781 Any instance which validates against a schema subset must be able to validate against the full  
1782 GJXDM reference schema.

1783

1784 Conformant Subset Schemas MUST NOT:

1785

- 1786 • Add local components
- 1787 • Flatten type structures
- 1788 • Modify namespaces
- 1789 • Change object types
- 1790 • Change element or type names
- 1791 • Change type inheritance
- 1792 • Make the subset inconsistent with the full reference schema

1793

1794 **4.2.2 Subset Namespace and Filename Rules**

1795

[NMS3]	A GJXDM schema subset and constraint schema MUST declare the same xsd:targetNamespace as the GJXDM baseline schema.  For example: “http://www.it.ojp.gov/jxdm/3.0.1” is the required targetNamespace for GJXDM version 3.0.1
--------	---

1796

[NMS7]	Each schema, for a GIEP, MUST be maintained in a separate namespace but will share a common group path.  That GIEP group path MUST be of the form:  “http:// <IEP owner domain name>/<IEP name>/<IEP version>”.  For example, xmlns = “http://www.myDomainName.com/Citation/1.0/” represents a local namespace copy of a GIEP group of schemas that support the exchange or representation of a Citation reference document. The version of 1.0 is assigned to the GIEP namespace by the provider of the reference document.  Note: All associated subset, document, constraint and extension schemas must be placed within this path. If any of the associated schemas, within the GIEP group change, the GIEP version MUST change.
--------	---

1797



[NMS8]	<p>Document and Extension schema's xsd:import element(s) MUST use a <u>relative URL</u> to locate the imported GJXDM Subset schema set.</p> <p>The GJXDM Subset path MUST be of the form:</p> <p>“http:// &lt;IEP owner domain name&gt;/&lt;IEP name&gt;/&lt;IEP version&gt;/subset/jxdm/&lt;GJXDM version&gt;/”.</p> <p>For example:</p> <p>xmlns=“http://www.myDomainName.com/Citation/1.0/subset/jxdm/3.0/“</p> <p>represents the location of a conformant subset of the full GJXDM v3.0 schema for a version 1.0 “Citation” IEP</p> <p>The relative URL MUST be valid within the structure of the GIEPD ZIP file specified in NMS17</p>
--------	---

1798

1799

1800 **4.2.3 Subset Schema File Layout**

1801

[GXS1]	<p>GJXDM subset schema or constraint schema MUST conform to the following physical layout as applicable:</p> <ul style="list-style-type: none"> <li>• XML Declaration</li> <li>• &lt;!-- ===== Required Documentation Comments Block ===== --&gt;</li> <li>• &lt;!-- ===== Name (common): ..... ===== --&gt;</li> <li>• &lt;!-- ===== Authoring agency/jurisdiction/generation date: ===== --&gt;</li> <li>• &lt;!-- ===== Description of business usage: ..... ===== --&gt;</li> <li>• &lt;!-- ===== xsd:schema Element With Namespaces Declarations===== --&gt; <ul style="list-style-type: none"> <li>• xsd:schema element to include Attribute definitions attributeFormDefault="unqualified" elementFormDefault="qualified"</li> </ul> <p>followed by Namespace Declarations in this order:</p> <ul style="list-style-type: none"> <li>• Target namespace</li> <li>• Default namespace</li> </ul> </li> <li>• &lt;!-- ===== Imports ===== --&gt;</li> </ul>
--------	--

	<ul style="list-style-type: none"> <li>• External Codelist Namespaces <ul style="list-style-type: none"> <li>• xmlns:xsd</li> <li>• External Codelist import schemaLocations and namespaces</li> </ul> </li> <li>• &lt;!-- ===== Global Attributes ===== --&gt; <ul style="list-style-type: none"> <li>• Global Attributes and Attribute Groups</li> </ul> </li> <li>• &lt;!-- ===== Complex Types and Simple Types ===== --&gt;</li> <li>• &lt;!-- ===== in alphabetized order xsd:TypeDefinitions ===== --&gt; <ul style="list-style-type: none"> <li>• Complex and Simple Types</li> </ul> </li> <li>• &lt;!-- ===== Attribute Declarations SHOULD BE in alphabetized order = --&gt;</li> <li>• &lt;!-- ===== Element Declarations SHOULD BE in alphabetized order == --&gt;</li> </ul>
--	--

1802

1803 **4.2.4 Subset Schema Generation Tool (SSGT)**

1804 A “Freely Available” software tool for generating conformant GJXDM subset schema(s) may be  
1805 found at:

1806

1807 <http://gjxdmtools.gtri.gatech.edu/ssgt/subset>

1808

1809 Other GJXDM schema generation tools may be used or developed by other parties, both open  
1810 source and proprietary.

1811

[NMS9]	<p>All IEP’s utilizing a GJXDM subset schema MUST produce the same files and filenames as the GJXDM Subset Schema Generator Tool (SSGT)</p> <p>For example:</p> <p>The SSGT generates “jxdm.xsd” as the standard filename for any GJXDM subset and produces a set of want-lists with a specified standard file structure. These SSGT artifacts must be produced for the IEP even if another software tool is used to produce them.</p>
--------	--

1812

1813 **4.3 Constraint Schema Naming and Design Rules**

1814

1815 **4.3.1 Rules for Conformant Constraint Schemas**

1816 GJXDM Constraint schemas embed localized constraints into GJXDM definitions.

1817

1818 The GJXDM namespace remains the same – just change the schema location attribute(s).

1819

1820 YOU CAN

1821 1. Change object types

1822 2. Change/drop type inheritance

1823 3. Create differently constrained types based on a single GJXDM type

1824 4. Make local component definitions

1825 5. Add localized constraints

1826 6. Force elements to appear or not appear

1827

1828 YOU CANNOT

1829 1. Change element names

1830 2. Change tag order or hierarchy

1831 3. Define new components to be referenced outside of this schema

1832 4. Leave out components required by instances

1833

1834 For example, a component is required if it:

1835

1836

1837 •May appear in a valid instance.

1838 •May be referred to by a schema outside GJXDM.

1839 •Is required by another required component.

1840

1841 You may omit entities which are not required, including:

1842

1843 •xmlns - namespace prefix declarations

1844 •xsd:import

1845 •xsd:complexType and xsd:simpleType elements that are "top-level"

1846 (direct children of the xsd:schema element)

1847 •xsd:element elements that are children of the xsd:schema element

1848 •xsd:attribute elements that are children of the xsd:schema element

1849 •xsd:element elements that are contained in a type definition

1850 •xsd:attribute elements that are contained in a type definition

1851 •xsd:attribute elements that appear in SuperType

1852 MetadataAttributeGroup.

1853 •xsd:enumeration elements for enumerations that are not relevant to the applications or instances

1854 •xsd:annotation or xsd:documentation elements

1855

1856

1857 Also see Rule GXS1 for Constraint schema file layout.

1858

### 1859 4.3.2 Constraint Namespace and Filename Rules

1860

[NMS14]	The Document and Extension schema's xsd:import element(s) MUST use a <u>relative URL</u> to locate the imported GJXDM Constraint schema.  The relative URL MUST be valid within the structure of the GIEPD ZIP file specified in NMS17
---------	--

1861

[NMS15]	<p>An IEP MAY have zero or one Constraint schema defined. The GJXDM Constraint schema module MUST have a schema filename of the following form:</p> <p style="margin-left: 20px;"><code>&lt;IEP name&gt;-&lt;IEP schema type&gt;.xsd</code></p> <p>where  <code>&lt;IEP name&gt;</code> = <code>&lt;IEP name&gt;</code> in the Document or Extension schema namespace  <code>&lt;IEP schema type&gt;</code> = constraint</p> <p>For example, if the IEP document schema namespace is <code>xmlns = http://www.myDomainName.com/Citation/1.0/document</code>, THEN the constraint schema filename would be:</p> <p style="margin-left: 20px;">Citation-constraint.xsd</p> <p>represents a constraint schema for version 1.0 of a Citation IEP where <code>&lt;IEP name&gt;</code> = “Citation” and <code>&lt;IEP schema type&gt;</code> = “constraint”</p>
---------	---

1862

1863 See RULE GX1 for Constraint Schema File Layout.

1864

1865 **4.4 Extension Schema Naming and Design Rules**

1866 Extension schemas define common local extensions

1867

[NMS10]	<p>A GJXDM IEP Extension schema targetNamespace MUST be of the form:</p> <p style="margin-left: 20px;">“<code>http://&lt;IEP owner domain name&gt;/&lt;IEP name&gt;/&lt;IEP version&gt;/&lt;IEP schema type[-suffix]&gt;</code>”.</p> <p>The “IEP name” and “IEP version” components of this identifier are chosen by the IEP workgroup (or chartering governance body). The IEP version component must follow the namespace versioning rules stated elsewhere in this specification. The “IEP name” component must be declared by the workgroup in the IEP overview document.</p> <p>The “IEP schema type” component of this identifier MUST be “extension” for the IEP’s extension namespace. If the IEP has multiple extension namespaces, the IEP workgroup MUST choose appropriate “suffix” namespace name values and document them in the IEP overview document.</p> <p>For example,  <code>xmlns:ext = “http://www.myDomainName.com/Citation/1.0/extension”</code></p>
---------	---

	represents the namespace for a version 1.0 Citation extension schema containing user-defined Types derived from extending or restricting GJXDM schema complexTypes and simpleTypes.
--	---

1868

[NMS11]	<p>The GJXDM Extension schema module MUST have a schema filename of the following form:</p> <p>&lt;IEP name&gt;-&lt;IEP schema type&gt; [&lt;-suffix&gt;].xsd</p> <p>where</p> <p>&lt;IEP name&gt; = &lt;IEP name&gt; defined in the Extension schema namespace</p> <p>&lt;IEP schema type&gt; = extension</p> <p>[suffix] = the [-suffix] optional name or number in the Extension schema namespace to support multiple extension schemas per IEP.</p> <p>For example, the IEP GJXDM extension schema filename for xmlns = "http://www.myDomainName.com/Citation/1.0/extension-joe" would be:</p> <p>Citation-extension-joe.xsd</p> <p>represents an extension schema for version 1.0 of a Citation IEP where &lt;IEP name&gt; = "Citation", &lt;IEP schema type&gt; = "extension" and &lt;suffix&gt; = "joe".</p>
---------	---

1869

[GTD3]	Extension schemas MUST NOT declare elements of type SuperType.
--------	--

1870

[GTD4]	Extension schemas MUST NOT declare complex types that extend SuperType without adding additional elements.
--------	--

1871

1872 Also see Rule GXS2 for extension schema layout.

1873

#### 1874 **4.4.1 Extension Schema File Layout**

[GXS2]	GJXDM extension schema MUST conform to the following physical layout as applicable:
--------	---

	<ul style="list-style-type: none"> <li>• XML Declaration</li> <li>• &lt;!-- ===== Required Documentation Comments Block ===== --&gt;</li> <li>• &lt;!-- ===== Name (common): ..... ===== --&gt;</li> <li>• &lt;!-- ===== Authoring agency/jurisdiction/generation date: ===== --&gt;</li> <li>• &lt;!-- ===== Description of business usage: ..... ===== --&gt;</li> <li>• &lt;!-- ===== xsd:schema Element With Namespaces Declarations===== --&gt;</li> <li>• xsd:schema element to include Attribute definitions attributeFormDefault="unqualified" elementFormDefault="qualified"</li> <li>followed by Namespace Declarations in this order:</li> <li>• Target namespace for Extension schema (http://{my namespace}/../extension</li> <li>• No Default namespace, a token such as ext: should be used for the Extension schema targetNamespace (eg. xmlns:ext="http://{my namespace}/../extension")</li> <li>• Declare the GJXDM schema, subset schema or constraint schema namespace (eg. xmlns:j="http://www.it.ojp.gov/jxdm/{jxdm version}")</li> <li>• xmlns:xsd</li> <li>• &lt;!-- ===== Imports ===== --&gt;</li> <li>• External GJXDM reference , subset schema or constraint schema import namespace</li> <li>• &lt;!-- == Extended/Restricted GJXDM Complex Types and Simple Types ==--&gt;</li> <li>• &lt;!-- ===== in alphabetized order xsd:TypeDefinitions ===== --&gt;</li> <li>• Complex and Simple Types</li> <li>• &lt;!-- ===== Element Declarations in alphabetized order ===== --&gt;</li> </ul>
--	---

1875

1876 **4.4.2 Extension Patterns**

1877

1878 **4.4.2.1 Scenarios for mapping business data/documents to GJXDM**

1879 Scenario #1

1880 Every property and relationship I need is in GJXDM

1881

1882 Scenario #2

1883 I need additional “property” elements within a currently defined GJXDM Type

1884

1885 Scenario #3

1886 I need to create new relationships between GJXDM types or elements by:

1887

1888 Inclusion (defining a “relational” Element to extend GJXDM type)

1889 Reference (use GJXDM ReferenceType Elements or add ReferenceType Elements)

1890 GJXDM Relationship Element (explicit named referencing )

1891

1892 Scenario #4

1893 I need a new type that doesn’t inherit or extend any elements of GJXDM

1894

1895 Scenarios #2 - #3 represent methods for extending/restricting the GJXDM components elements  
1896 and relationships to define the content of a specific document or data exchange set of elements.

1897

#### 1898 **4.4.2.2 Restriction of GJXDM components**

1899 An existing GJXDM type can be modified to fit the requirements of customization through XSD  
1900 derivation. Restriction of ComplexTypes should NOT be done using the xsd: base=restriction  
1901 construct. The recommended approach is to utilize the GJXDM subset schema generator to  
1902 define a restricted set of elements for an existing GJXDM ComplexType.

1903

1904 As noted in Rule [ELD4], xsd: base=restriction should only be used for SimpleType components  
1905 of GJXDM to add constraints and/or other facets to restrict the permissible range of values for  
1906 the SimpleType.

1907

1908

[GXS15]	Complex Type extension and Simple Type restriction MAY be used where appropriate.
---------	---

1909

[GTD5]	Extended types MUST be derived from SuperType
--------	---

1910

1911

#### 1912 **4.4.2.3 Extensions to GJXDM components**

1913 The following methods may be applied for extending GJXDM components in creating GJXDM  
1914 compliant Reference Documents and Data exchanges:

1915

1916 Method 1 : Extension using “Type Substitution”

1917 Method 2: Extension using a “Concrete Typing” construction to “Avoid Type Substitution”

1918 Method 3: Extension of relationships using ReferenceType elements

1919 Method 4: Extensions of relationships using GJXDM Relationship Element

1920

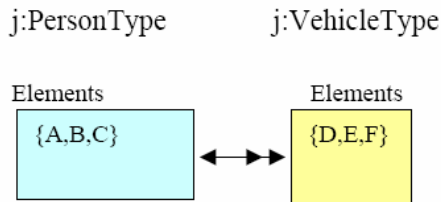
1921  
1922

Method 1 : Extension using "Type Substitution"

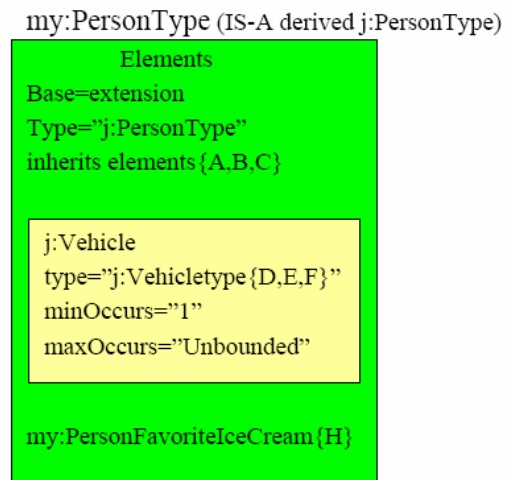
This method and the remaining methods will all assume we are extending PersonType(component) {A,B,C} with a 1:M or M:M "relationship element" to the VehicleType (component). Depending on the method we will use j:Vehicle {D,E,F}, j:VehicleReference {F}, my:PersonVehicle{G} and Relationship Qname="xxxx" {R1,R2} to illustrate the extension method. I will also add a "property element" as "my:PersonFavoriteIceCream" {H}

**Type Substitution**

*Before Extension*



*After Extension(new 1:M relationship)*



This **Method-1** construction requires a type substitution declaration in the XML instance to dynamically invoke my:PersonType to replace j:PersonType at runtime. Following is an XML snippet of what is required:

```

<my:SampleDocument>
  <j:Person xsi:type="my:PersonType" > ***type substitution***
    <j:PersonElements {A,B,C}>
    <j:Vehicle>
      <j:Vehicle Elements {D,E,F}>
    </j:Vehicle>
    <my:PersonFavoriteIceCream/ {H}>
  </j:Person>
</my:SampleDocument>

```

1923 •

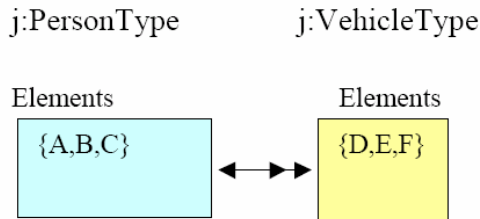


Method 2 : Extension using “Concrete Typing” to “Avoid Type Substitution”

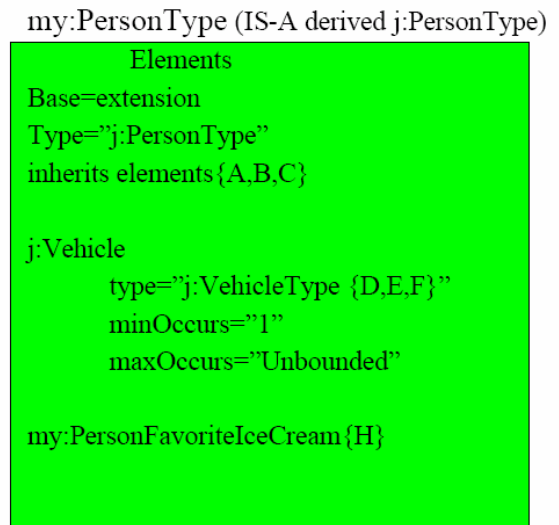
This method extends PersonType(component) {A,B,C} with a 1:M “relationship element” to the VehicleType (component) {D,E,F} using my:PersonVehicle{G}. I will also add a “property element” as “my:PersonFavoriteIceCream” {H}

### Extension Using “Concrete Typing”

*Before Extension*



*After Extension(new 1:M relationship)*



This **Method-2** construction eliminates the “type substitution” declaration in the method-1 XML instance and avoids requirement to dynamically invoke my:PersonType. Following is an XML snippet of what it looks like:

```
<my:SampleDocument>
  <my:Person > *** No type substitution required ***
    <j:PersonElements {A,B,C}>
    <j:Vehicle>
      <j:Vehicle Elements {D,E,F}>
    </j:Vehicle>
    <my:PersonFavoriteIceCream/ {H}>
  </my:Person>
</my:SampleDocument>
```

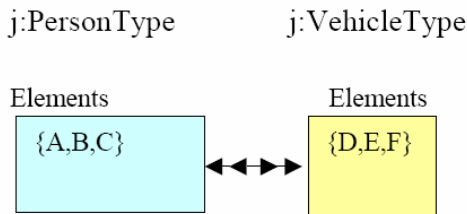
note: Cascade of **my:Person** under **my:SampleDocument** instead of dynamically substituting j:Person with my:PersonType as done in Method-1.

Method 3 : Extension using ReferenceType elements

This method extends PersonType(component) {A,B,C} with a M:M “relationship element” to the VehicleType (component) {D,E,F} using j:VehicleReference {F}. I will also add a “property element” as “my:PersonFavoriteIceCream” {H}. Both elements will be added using a **Method-2** approach.

**Extension Using “Concrete Typing” and Reference Elements**

*Before Extension*



*After Extension(new M:M relationship)*

my:PersonType (IS-A derived j:PersonType)

```
Elements
Base="extension"
Type="j:PersonType"
inherits elements {A,B,C}

j: VehicleReference
  type="j:ReferenceType"
  minOccurs="1"
  maxOccurs="Unbounded"

my:PersonFavoriteIceCream {H}
```

my:VehicleType (IS-A derived j:VehicleType)

```
Elements
Base="extension"
Type="j:VehicleType"
inherits elements {D,E,F}

j: PersonReference
  type="j:ReferenceType"
  minOccurs="1"
  maxOccurs="Unbounded"
```

This **Method-3** construction includes the `j:VehicleReference` element in `my:PersonType` and `j:PersonReference` element in `my:VehicleType` for relating one or more persons to one or more vehicles by reference-id's instead of the "Inclusion" method for adding "*relationship elements*" to a component. In the following XML example, this avoids duplicating `Vehicle#1` elements for `Person_A` and `Person_B`. Instead, `Person_A` has a `VehicleReference j:ref="xx"` and `Person_B` has a `VehicleReference j:ref="xx"` and `Vehicle#1` is only defined once in the XML instance. This would not be the case if we instead used "Inclusion" with the element `j:Vehicle` as shown in Method-2 above. Following is an XML snippet of what it looks like:

```

<my:SampleDocument>
  <my:Person j:id="yy" > (Person_A)
    <j:PersonElements {A,B,C}>
      <j:VehicleReference j:ref="xx"/ > (pointer to Vehicle#1)
    <my:PersonFavoriteIceCream/ {H}>
  </my:Person>
  <my:Person j:id="zz" > (Person_B)
    <j:PersonElements {A,B,C}>
      <j:VehicleReference j:ref="xx"/ > (pointer to Vehicle#1)
    <my:PersonFavoriteIceCream/ {H}>
  </my:Person>
  <my:Vehicle j:id="xx" > (Vehicle#1)
    <j:Vehicle elements {D,E,F} >
  </my:Vehicle >
  <my:Vehicle j:id="tt" > (Vehicle#2)
    <j:Vehicle elements {D,E,F} >
      <j:PersonReference j:ref="zz"> (pointer to Person B)
    </my:Vehicle >
</my:SampleDocument>

```

In this example Person A and Person B has-a relationship to Vehicle#1 and Vehicle#2 has-a relationship to Person B

Note: The (VehicleReference) relationship of Person A and Person B to Vehicle#1 may not be the same as the (PersonReference) relationship of Vehicle#2 to Person B.

For example Person A and Person B may have the relationship of "photographed" Vehicle#1 but the Vehicle#2 to Person B relationship may be Vehicle#2 was "VandalizedBy" Person B.

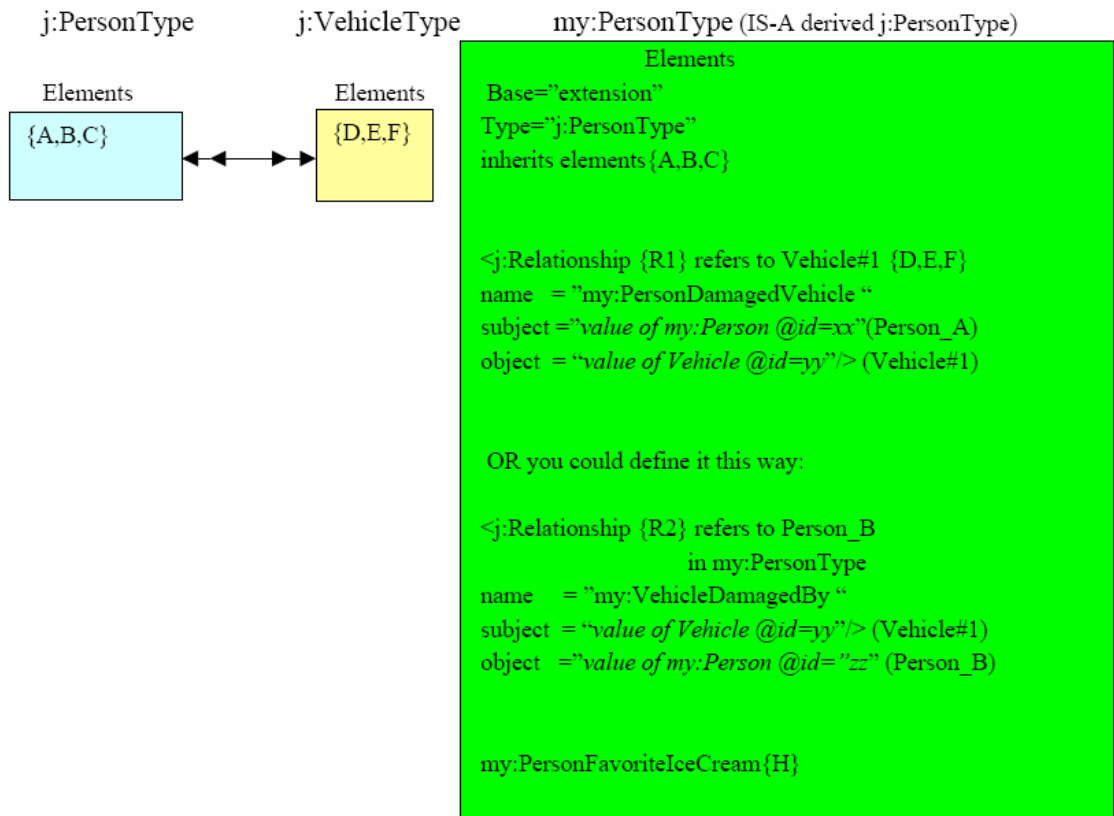
**Method4 : Extension using GJXDM Relationship element for Many-to-Many relationships**

This method extends PersonType(component) {A,B,C} with a M:M “relationship element” to the VehicleType (component) {D,E,F} using j:Relationship Elements {R1,R2}. I will also add a “property element” as “my:PersonFavoriteIceCream” {H}. All elements will be added using a **Method-2** approach.

**Extension Using “Concrete Typing” and Relationship Elements**

*Before Extension*

*After Extension(new M:M relationship)*



This **Method-4** construction includes two j:Relationship elements in my:PersonType for relating one or more persons to one or more vehicles by reference instead of the “Inclusion” method for adding “relationship elements” to a component. This avoids duplicating Vehicle#1 elements for Person\_A and Person\_B. Instead, Person\_A has a Relationship Qname=“my:PersonDamaged” with subject=“xx” and Person\_B with subject=“zz” and Vehicle#1. Vehicle#1 is defined once in the

XML instance. This would not be the case if we instead used “Inclusion” with the element j:Vehicle as shown in Method-2 above. Following is an XML snippet of what it looks like:

```

<my:SampleDocument>
  <my:Person j:id="xx" >
    <j:PersonElements {A,B,C}>
      <j:Relationship name=" my:PersonDamagedVehicle “ {R1}
        subject ="xx"(Person_A)
        object = "yy"/> (Vehicle#1)
      <my:PersonFavoriteIceCream/ {H}>
    </my:Person>
  <my:Person j:id="zz" >
    <j:PersonElements {A,B,C}>
      <j:Relationship name=" my:VehicleDamagedBy “ {R2}
        subject ="yy"( Vehicle#1)
        object = "zz"/> (Person_B)
      <my:PersonFavoriteIceCream/ {H}>
    </my:Person>
  <j:Vehicle j:id="yy" > (Vehicle#1)
    <j:Vehicle elements {D,E,F} >
  </j:Vehicle >
</my:SampleDocument>

```

There are many included “*relationship elements*” already defined in GJXDM which should be used before defining a new relation between objects. For example, j:PropertyDisposition, j:CaseWitness, j:CaseCharge etc. all are describing “*relationship elements*” or non-hierarchical relationships between two GJXDM Components (Classes). The ReferenceType and RelationshipType elements need only be used when many-to-many relationships in your data model need to be resolved.

1928

1929

#### 1930 4.4.3 Controlling Type extension/restriction (final)

1931 XSD:final attribute is utilized to stop further restriction or extension on simple or  
1932 complexTypes

1933

1934 Attribute values for preventing Type extension/restriction:

1935

1936 Final="#all" no extension or restriction of Type allowed

1937 Final="restriction" no restriction of Type allowed

1938 Final="extension" no extension of Type allowed

1939

1940 <xsd:complexType name="Publication" final="#all" ...> Publication cannot be extended nor  
1941 restricted

1942

1943 <xsd:complexType name="Publication" final="restriction" ...> Publication cannot be restricted  
1944

1945 <xsd:complexType name="Publication" final="extension" ...> Publication cannot be  
1946 extended

1947

[GXS8]	The xsd:final attribute MUST be used when schema developers want to prevent restriction or extension of a user-defined simpleTypes or complexType extensions
--------	--

1948

#### 1949 4.4.4 Controlling Type and Element Substitution (block)

1950 Schema provides the xsd:attribute named "block" which can be used to block  
1951 extensions/restrictions and/or substitutions to Components (ComplexTypes & SimpleTypes) and  
1952 Elements. Following is the block attribute syntax which would be used in Schema at the element  
1953 or component level as deemed appropriate:

1954

1955 block = "#all" No extension, restriction or substitution allowed

1956 block = "restriction" No restriction allowed

1957 block = "extension" No extension allowed

1958 block = "substitution" No substitution allowed

1959 block = "restriction,extension" No restriction or extension allowed (but substitution okay)

1960 block = "restriction,substitution" No restriction or substitution (but extension okay)

1961 block = "extension,substitution" No extension or substitution (but restriction okay)

1962

1963 For example, the following two snippets of schema will prevent element substitution and type  
1964 substitution :

1965

1966 <xsd:element name="..." type="..." **block="substitution"/>**>

1967

1968 <xsd:complexType name="PublicationType" block="substitution">

1969

1970

[GXS9]	The xsd:block attribute MUST be used when schema developers want to prevent use of "type substitution" and "element substitution" in XML instance documents.
--------	--

1971

1972

1973

#### 4.4.4.1 XML Instance Type Substitution

1974 Using method 2 “concrete typing” to include new elements into GJXDM components for  
1975 extensions versus “type substitution” provides the schema developer with greater control over  
1976 XML Instances and a greater assurance that XML Instance authors are not inadvertently  
1977 substituting new types which the developer never intended. Also, “type substitution” at run-time  
1978 may provide implementation errors for vendor product(s) due to the “dynamic” substitution of  
1979 validating schema within an Instance XML document.

1980

1981

#### 4.4.4.2 Substitution groups, redefines

1982 XSD:substitution is “a feature of W3C XML Schema, allowing you to define groups of elements  
1983 that may be used interchangeably in instance documents. They are not declared as element  
1984 groups, but through the substitutionGroup attribute of xsd:element global definitions.”

1985 The GJXDM information exchange MNDR has made the decision to not allow the use of  
1986 substitution groups due to the following issues:

1987 ♦ SubstitutionGroup “may work with some schema processors but relies on a liberal  
1988 interpretation of the Recommendation, which may lead to interoperability issues.”

1989 ♦ SubstitutionGroup “introduces multiple names for the same GJXDM element  
1990 leading to confusion”

1991

[GXS7]	The xsd:SubstitutionGroups feature MUST NOT be used.
--------	--

1992

1993 <Redefine> enables you to create a local schema which includes all of the components and  
1994 elements of the full GJXDM and then <redefine> selected GJXDM component(s) or element(s)  
1995 within your local schema. The redesigned (redefined) GJXDM component(s) and element(s)  
1996 OVERRIDE the definitions/structures defined in the Official GJXDM dictionary. All your XML  
1997 instance tags from the redefines will appear to be original GJXDM elements instead of  
1998 extensions|restrictions to GJXDM elements.

1999

2000 There is concern that the XML instance, when using <redefine>, will be indistinguishable from a  
2001 schema comprised of GJXDM elements only because the local extension|restriction elements will  
2002 all have the GJXDM namespace. For that reason, the GJXDM MNDR has formulated the  
2003 following rule:

2004

2005

[GXS12]	The xsd:redefine element MUST NOT be used.
---------	--

2006

## 4.5 Document Schema Naming and Design Rules

2007

2008

[NMS12]	A GJXDM IEP Document schema targetNamespace MUST be of the form: “http:// <IEP owner domain name>/<IEP name>/<IEP version>/<IEP schema type[-
---------	--

	<p>suffix]&gt;”.</p> <p>The “IEP name” and “IEP version” components of this identifier are chosen by the IEP workgroup (or chartering governance body). The IEP version component must follow the namespace versioning rules stated elsewhere in this specification. The “IEP name” component must be declared by the workgroup in the IEP overview document.</p> <p>The “IEP schema type” component of this identifier MUST be “document” for the IEP’s document namespace. If the IEP has multiple document namespaces, the IEP workgroup MUST choose appropriate “suffix” namespace name values and document them in the IEP overview document.</p> <p>For example,  xmlns = “http://www.myDomainName.com/Citation/1.0/document”  represents the namespace for a version 1.0 Citation IEP document schema</p>
--	--

2009

[NMS13]	<p>The GJXDM Document schema module MUST have a schema filename of the following form:</p> <p>&lt;IEP name&gt;-&lt;IEP schema type&gt; [&lt;-suffix&gt;].xsd</p> <p>where</p> <p>&lt;IEP name&gt; = &lt;IEP name&gt; in the Document schema namespace  &lt;IEP schema type&gt; = document  [suffix] = the [-suffix] optional name or number in the Document schema namespace to support multiple document schemas per IEP.</p> <p>For example, the IEP GJXDM document schema filename for xmlns = “http://www.myDomainName.com/Citation/1.0/document” would be:</p> <p>Citation-document.xsd</p> <p>represents a document schema for version 1.0 of a Citation IEP where &lt;IEP name&gt; = “Citation”, &lt;IEP schema type&gt; = “document” and &lt;suffix&gt; = option not used.</p>
---------	--

2010

2011 **4.5.1 Document Schema File Layout**

2012 The basic GJXDM MNDR document schema template can be summarized in the following steps:

2013



- 2014 1) Define the target namespace of the document
- 2015
- 2016 • Reference/import the GJXDM
- 2017 • Define the root element and type for the document
- 2018 • Extend the root type from GJXDM DocumentType (or a derivative)
- 2019 • Add document content
- 2020
- 2021
- 2022 2) Create a local element and reuse a GJXDM type
- 2023
- 2024 • use actual GJXDM type
- 2025 • extend GJXDM type
- 2026

[GXS16]	All GJXDM document schema MUST NOT declare new complexTypes or simpleTypes within the document schema. Any complexTypes and simpleTypes needed by the document schema must be imported from external schema.
---------	--

- 2027
- 2028

[GXS3]	<p>GJXDM Document schema MUST conform to the following physical layout as applicable:</p> <ul style="list-style-type: none"> <li>• XML Declaration</li> <li>• &lt;!-- ===== Required Documentation Comments Block ===== --&gt;</li> <li>• &lt;!-- ===== Name (common): ..... ===== --&gt;</li> <li>• &lt;!-- ===== Authoring agency/jurisdiction/generation date: ===== --&gt;</li> <li>• &lt;!-- ===== Description of business usage: ..... ===== --&gt;</li> <li>• &lt;!-- ===== xsd:schema Element With Namespaces Declarations===== --&gt;</li> <li>• xsd:schema element followed by Namespace Declarations in this order: <ul style="list-style-type: none"> <li>• Target namespace for Document schema</li> <li>• No Default namespace, a token such as doc: or rap: (for a rapsheet) should be used for the Document schema targetNamespace</li> <li>• Declare the “optional” GJXDM extension schema namespace (eg. xmlns:ext=”http://{my namespace}.../extension”)</li> <li>• Declare the GJXDM schema, subset schema or schema constraint namespace (eg. xmlns:j=”http://www.it.ojp.gov/jxdm/{jxdm version}”)</li> </ul> </li> </ul>
--------	---

	<ul style="list-style-type: none"> <li>• xmlns:xsd</li> <li>• &lt;!-- ===== Imports ===== --&gt;</li> <li>• External “optional” GJXDM extension schema import namespace</li> <li>• External GJXDM schema, subset schema or constraint schema import namespace</li> <li>• &lt;!-- ===== Root Element ===== --&gt;</li> <li>• Root Element Declaration</li> <li>• Root Element Type Definition</li> <li>• &lt;!-- ===== Type Definition ===== --&gt;</li> <li>• Define Root Type; extend from j:DocumentType; where complexType name=“{Root Element Name}Type” (eg. for Root Element Name “CitationDocument” type=“doc:CitationDocumentType” then the complexType name = “CitationDocumentType” which is the Root Element Name suffixed with the word “Type”).</li> </ul>
--	---

2029

2030

2031 **4.6 Instance Naming and Design Rules**

2032 This subsection describes the rules for constructing instance documents, including requirements  
2033 for root elements and validation methods.

2034

2035 **4.6.1 Root Element**

2036 In XSD, every global element is eligible to act as a root element in an instance document. Rule  
2037 ELD1 requires the identification of a single global element in each GJXDM MNDR schema to be  
2038 carried as the root element in the instance document. GJXDM MNDR exchange documents (XML  
2039 instances) must have a single root element as defined in the corresponding GJXDM MNDR  
2040 document schema.

2041

<b>[RED1]</b>	Every GJXDM instance document must have as its root element the single global element defined in its IEPD document schema.
---------------	--

2042

2043

2044 **4.6.2 XML Instance validation**

2045 Business information exchanges require a high degree of precision to ensure  
2046 that application processing and corresponding business cycle actions are reflective of the

2047 purpose, intent, and information content agreed to by both trading partners. Schemas  
2048 provide the necessary mechanism for ensuring that instance documents do in fact support  
2049 these requirements.  
2050

[IND1]	All GJXDM instance documents MUST validate to a corresponding Document schema.
--------	--

2051  
2052

[IND4]	All GJXDM instance documents MUST contain the following namespace declaration in the root element: xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
--------	--

2053

2054 It is Recommended that Validating parsers SHOULD be able to override the schemaLocation  
2055 attribute in processing XML Instance Documents. (note: Relying on XML Instance  
2056 schemaLocation values without verifying before processing is a Security Risk)

2057

#### 2058 **4.6.2.1 Schema Location**

2059 GJXDM is rapidly becoming a national and potentially international standard that will be used in  
2060 perpetuity by justice agencies around the globe. It is important that these users have unfettered  
2061 access to all GJXDM conformant schema.

2062

[ATD4]	Each xsd:schemaLocation attribute declaration MUST contain a system-resolvable URI, referencing the location of the schema or schema module.
--------	--

2063  
2064

#### 2065 **4.6.3 Character encoding**

[IND2]	All GJXDM instance documents MUST always identify their character encoding with the XML declaration.
--------	--

2066  
2067

[IND3]	In conformance with ISO/IETF/ITU/UNCEFACT Memorandum of Understanding Management Group (MOUMG) Resolution 01/08 (MOU/MG01n83), all GJXDM XML SHOULD be expressed using UTF-8.
--------	---

2068

#### 2069 **4.6.4 Empty content**

2070 Usage of empty elements within XML instance documents are a source of controversy  
2071 for a variety of reasons. An empty element does not simply represent data that is missing.  
2072 It may express data that is not applicable for some reason, trigger the expression of an  
2073 attribute, denote all possible values instead of just one, mark the end of a series of data, or  
2074 appear as a result of an error in XML file generation. Conversely, missing data elements  
2075 can also have meaning - data not provided by a trading partner. In information exchange

2076 environments, different trading partners may allow, require or ban empty elements. GJXDM  
2077 MNRD has determined that empty elements do not provide the level of assurance necessary for  
2078 business information exchanges and as such will not be used.

2079

[IND5]	GJXDM conformant instance documents MUST NOT contain an element devoid of content unless explicitly indicated by the xsi:nil="true" attribute
--------	---

2080

2081 Absence of data should only be represented by using the “nillable” attribute as defined in  
2082 Rule ATD5.

2083

[IND6]	The absence of a construct or data in a GJXDM instance document MUST NOT carry meaning.
--------	---

2084

2085 Example:

2086

2087

Valid:

2088

```
<PersonName>  
  <PersonGivenName>John</PersonGivenName>  
  <PersonMiddleName xsi:nil="true"/>  
  <PersonSurName>Doe</PersonSurName>  
</PersonName>
```

2089

2090

2091

2092

2093

2094

2095

Invalid:

2096

2097

```
<PersonName>  
  <PersonGivenName>John</PersonGivenName>  
  <PersonMiddleName/> ***EMPTY Content***  
  <PersonSurName>Doe</PersonSurName>  
</PersonName>
```

2098

2099

2100

2101

2102

2103

## Appendix A. GJXDM MNDR Checklist

2104 The following checklist incorporates relevant UBL XML naming and design rules as defined in  
2105 UBL Naming and Design Rules version 1.0.1, 15 November 2004. UBL rules modified for GJXDM  
2106 information exchange schema(s) are highlighted in GREEN. Additional rules are included to  
2107 address GJXDM information exchange specific Naming & Design rules drawn from GTRI GJXDM  
2108 Training Workshop materials and related documents from IJIS and OASIS LegalXML Integrated  
2109 Justice Technical Committee documents. The checklist is in alphabetical sequence as follows:  
2110

Attribute Declaration Rules	(ATD)
Code List Rules	(CDL)
ComplexType Definition Rules	(CTD)
ComplexType Naming Rules	(CTN)
Documentation Rules	(DOC)
Element Declaration Rules	(ELD)
General Naming Rules	(GNR)
General Type Definition Rules	(GTD)
General XML Schema Rules	(GXS)
Instance Document Rules	(IND)
Mapping Documentation Rules	(MAP)
Modeling Constraints Rules	(MDC)
GIEP Mapping Rules	(MAP)
Namespace Rules	(NMS)
Root Element Declaration Rules	(RED)
Schema Structure Modularity Rules	(SSM)
Standards Adherence Rules	(STA)
Versioning Rules	(VER)

2111

2112

<h2>A.1 Attribute Declaration Rules</h2>	
[ATD1]	User defined attributes SHOULD NOT be used.
[ATD2]	If a Schema Expression contains one or more common attributes that apply to all elements contained or included or imported therein, the common attributes MUST be declared as part of a global attribute group. ( For example: see the Global JXDM global attribute group named <b>”SuperTypeMetadata”</b> )
[ATD3]	<p>For each Global JXDM user-defined element of simpleType and a xsd:restriction element;</p> <p>an xsd:base attribute MUST be declared and set to the appropriate GJXDM datatype.</p> <p>Note: the set of valid GJXDM datatypes are based on ebXML Core Component Technical Specification v1.9 and include the following 10 simpleTypes:</p> <ul style="list-style-type: none"> <li>• Amount</li> <li>• BinaryObject (secondary: Graphic, Picture,Sound,Video)</li> <li>• Code</li> <li>• DateTime (secondary: Date, Time)</li> <li>• Identifier (authorized abbreviation: ID)</li> <li>• Indicator</li> <li>• Measure</li> <li>• Numeric (secondary: Value, Rate, Percent)</li> <li>• Quantity</li> <li>• Text (secondary: Name)</li> </ul> <p>Reference:GTRI May 2004 Developer Workshop</p>
[ATD4]	Each xsd:schemaLocation attribute declaration MUST contain a system-resolvable URI, referencing the location of the schema or schema module.

[ATD5]	<p>The xsd built in nillable attribute MUST be used and set nillable="true" for any Global JXDM user-defined element which has simpleContent.</p> <p>Note: An example from GJXDM v3.0.2 :</p> <pre>&lt;xsd:element name="WitnessLocationDescriptionText" type="j:TextType" nillable="true"&gt;</pre>
[ATD6]	<p>The xsd:any attribute MUST NOT be used.</p>

2115  
2116

<h2 style="color: blue;">A.2 Code List Rules</h2>	
[CDL1]	<p>All Global JXDM Code Lists MUST be part of a Global JXDM or externally maintained Code List; they MUST NOT be included in a document or extension schema..</p>
[CDL2]	<p>The Global JXDM SHOULD identify and use external standardized code lists whenever practical rather than develop its own Global JXDM-native code lists.</p>
[CDL3]	<p>The Global JXDM information exchange developer, through extension/restriction, MAY design and use a “contextually” defined code list where an existing GJXDM code list needs to be extended, or where no suitable external code list exists.</p>
[CDL4]	<p>All GJXDM maintained or information exchange developer Code Lists MUST be enumerated using the GJXDM Code List Schema Module. <b>(See Global JXDM standards for NonStandardCodeType properties. ) TRUE???? Need to clarify the intent of NonStandardCodeType with the XSTF. (Tom Carlson will investigate).</b></p>
[CDL5]	<p>The name of each GJXDM information exchange Code List Schema MUST be of the form: {Owning Organization}_{Code List Name}_[version number].xsd ;</p>
[CDL6]	<p>An xsd:import element MUST be declared for every code list required in a GJXDM information exchange schema. Each codelist MUST be in its own namespace; the namespace identifier MUST be consistent with the same rules as extension schemas.</p>
[CDL7]	<p>When creating a local code list, exchange developer MUST follow the UBL code list schema and annotation rules. <b>(need to coordinate this rule with CDL4. Does CDL4 mean GJXDM has a standard for user-defined</b></p>

codelists???)  
<http://docs.oasis-open.org/ubl/cd-UBL-1.0/doc/cl/wd-ublcisc-codelist-20040420.pdf>

2117  
2118

## A.2 Code List Rules

[CDL8]	Users of the GJXDM MAY identify any subset they wish from an identified code list for their own trading community conformance requirements.
[CDL9]	The <code>xsd:schemaLocation</code> MUST include the complete URI used to identify the relevant code list schema.

2119

## A.3 ComplexType Definition Rules

[CTD1]	For every class identified in GJXDM extension and document schema, a named <code>xsd:complexType</code> MUST be defined.
[CTD2]	Every GJXDM user-defined <code>xsd:complexType</code> definition content model MUST use the <code>xsd:sequence</code> element with appropriate global element references to reflect each property of its class.
[CTD3]	For every user-defined datatype based on the valid set of GJXDM datatypes, a named <code>xsd:complexType</code> or <code>xsd:simpleType</code> MUST be defined.

2120  
2121  
2122  
2123

## A.4 ComplexType Naming Rules

[CTN1]	<p>A user-defined <code>xsd:complexType</code> <i>name</i> MUST name the object suffixed by the word "Type".</p> <p>For example <code>&lt;xsd:complexType name="PersonType"&gt;</code> is correct</p> <p>And <code>&lt;xsd:complexType name="Person"&gt;</code> would be incorrect.</p>
--------	---



## A.5 Documentation Rules

<p>[DOC1]</p>	<p>The <code>xsd:documentation</code> element for every GJXDM user-defined Element MUST contain a structured set of annotations in the following sequence and pattern:</p> <ul style="list-style-type: none"> <li>• Version (optional): An indication of the evolution over time of the Datatype.</li> <li>• Definition(mandatory): The semantic meaning of an Element</li> <li>• Cardinality(mandatory): Indication whether the complexType Element (Property) represents a not-applicable, optional, mandatory and/or repetitive characteristic of the parent complexType</li> <li>• AssociatedObjectClassQualifier (optional): Associated Object Class Qualifiers describe the 'context' of the relationship with another complexType object. That is, it is the role the contained Element plays within its association with the containing complexType object.</li> <li>• AssociatedObjectClass (mandatory); Associated Object Class is the Object Class at the other end of this association. It represents the Aggregate Business Information Entity contained by the Association Business Information Entity. (For example: the element <code>PersonName</code> within the complexType <code>PersonType</code> has Name as the AssociatedObjectClass contained in the Aggregate Business Information Entity called <code>PersonType</code>.)</li> <li>• AlternativeBusinessTerms (optional): Any synonym terms under which the Element is commonly known and used in the business.</li> <li>• Examples (optional): Examples of possible values for the Element</li> </ul> <p>Context Element should go here as well if not defined in a later NDR, so this is a placeholder for now. The list of Context Drivers is in the UBL ccts: core-component parameters schema. The current values for Context drivers are:</p> <ul style="list-style-type: none"> <li>8 Business Process</li> <li>9 Product Classification</li> <li>10 Geopolitical Region</li> <li>11 Official Constraint</li> <li>12 Business Process Role</li> <li>13 Supporting Role</li> <li>14 System Capabilities</li> </ul>
---------------	---

2127  
2128

A.6 Element Declaration Rules	
[ELD1]	Each GJXDM information exchange document schema MUST identify one and only one global element declaration that defines the exchange document being conveyed in the Schema expression. That global element MUST include an <code>xsd:annotation</code> child element which MUST further contain an <code>xsd:documentation</code> child element that declares " <i>This element MUST be conveyed as the root element in any instance document based on this Schema expression.</i> "
[ELD2]	All element declarations MUST be global
[ELD3]	For every class defined as a GJXDM user-defined types, a global element bound to the corresponding <code>xsd:complexType</code> MUST be declared.  For e.g. a schema defining a <code>complexType</code> named <code>my:FavoritePersonType</code> would need to declare a global element named "FavoritePerson" of <code>objectType = my:FavoritePersonType</code> to bind a global element name to the <code>complexType</code> .

2129

A.6 Element Declaration Rules	
[ELD4]	For every user-defined <code>simpleType</code> , an <code>xsd:restriction</code> element MUST be declared
[ELD5]	Empty elements MUST NOT be declared.
[ELD6]	Global <code>simpleType</code> elements declared with Qualified Properties must be of the same type as their corresponding Unqualified Property.
[ELD7]	The <code>xsd:any</code> element MUST NOT be used.

2130  
2131  
2132

## A.7 General Naming Rules

[GNR1]	User-defined information exchange XML elements, attributes and type names MUST be composed in the English language, using the primary English spellings provided in the Webster's English Dictionary.
[GNR2]	GJXDM information exchange XML element, attribute and type names MUST be ebXML CCTS ISO 11179 compliant
[GNR3]	GJXDM information exchange XML element, attribute and type names MUST NOT include spaces, other separators, or characters not allowed by W3C XML 1.0 for XML names. The only <b>exception</b> to this rule is the use of periods in GJXDM element names.
[GNR4]	GJXDM information exchange XML element, attribute, and simple and complex type names MUST NOT use acronyms, abbreviations, or other word truncations, except those in the list of exceptions published in Appendix B.
[GNR5]	The acronyms and abbreviations listed in Appendix B MUST always be used.

2133

[GNR6]	GJXDM information exchange XML element, attribute and type names MUST be in singular form unless the concept itself is plural.
[GNR7]	The UpperCamelCase (UCC) convention MUST be used for naming elements and types.
[GNR8]	The lowerCamelCase (LCC) convention MUST be used for naming attributes.
[GNR9]	GJXDM element names, attributes and type names MUST not be modified, even when GJXDM names conflict with rules GNR1 – GNR3. For example, the use of periods in GJXDM conflicts with GNR3.

2134

<b>A.8 General Type Definition Rules</b>	
[GTD1]	All types MUST be named.
[GTD2]	The <code>xsd:anyType</code> MUST NOT be used.
[GTD3]	Extension schemas MUST NOT declare elements of type SuperType.
[GTD4]	Extension schemas MUST NOT declare complex types that extend SuperType without adding additional elements.
[GTD5]	Extended types MUST be derived from SuperType

## A.9 General XML Schema Rules

[GXS1]	<p>GJXDM subset schema or constraint schema MUST conform to the following physical layout as applicable:</p> <ul style="list-style-type: none"> <li>• XML Declaration</li> <li>• &lt;!-- ===== Required Documentation Comments Block ===== --&gt;</li> <li>• &lt;!-- ===== Name (common): ..... ===== --&gt;</li> <li>• &lt;!-- ===== Authoring agency/jurisdiction/generation date: ..... ===== --&gt;</li> <li>• &lt;!-- ===== Description of business usage: ..... ===== --&gt;</li> <li>• &lt;!-- ===== xsd:schema Element With Namespaces Declarations ===== --&gt; <ul style="list-style-type: none"> <li>• xsd:schema element to include Attribute definitions attributeFormDefault="unqualified" elementFormDefault="qualified" followed by Namespace Declarations in this order: <ul style="list-style-type: none"> <li>• Target namespace</li> <li>• Default namespace</li> </ul> </li> </ul> </li> <li>• &lt;!-- ===== Imports ===== --&gt; <ul style="list-style-type: none"> <li>• External Codelist Namespaces</li> <li>• xmlns:xsd</li> <li>• External Codelist import schemaLocations and namespaces</li> </ul> </li> <li>• &lt;!-- ===== Global Attributes ===== --&gt; <ul style="list-style-type: none"> <li>• Global Attributes and Attribute Groups</li> </ul> </li> <li>• &lt;!-- ===== Complex Types and Simple Types ===== --&gt;</li> <li>• &lt;!-- ===== in alphabetized order xsd:TypeDefinitions ===== --&gt; <ul style="list-style-type: none"> <li>• Complex and Simple Types</li> </ul> </li> <li>• &lt;!-- ===== Attribute Declarations SHOULD BE in alphabetized order ===== --&gt;</li> <li>• &lt;!-- ===== Element Declarations SHOULD BE in alphabetized order ===== --&gt;</li> </ul>
[GXS2]	<p>GJXDM extension schema MUST conform to the following physical layout as applicable:</p> <ul style="list-style-type: none"> <li>• XML Declaration</li> </ul>

	<ul style="list-style-type: none"> <li>• &lt;!-- ===== Required Documentation Comments Block ===== --&gt;</li> <li>• &lt;!-- ===== Name (common): ..... ===== --&gt;</li> <li>• &lt;!-- ===== Authoring agency/jurisdiction/generation date: ===== --&gt;</li> <li>• &lt;!-- ===== Description of business usage: ..... ===== --&gt;</li>   <li>• &lt;!-- ===== xsd:schema Element With Namespaces Declarations ===== --&gt; <ul style="list-style-type: none"> <li>• xsd:schema element to include Attribute definitions attributeFormDefault="unqualified" elementFormDefault="qualified" followed by Namespace Declarations in this order:</li> <li>• Target namespace for Extension schema (http://{my namespace}/.../extension[/name]/version</li> <li>• No Default namespace, a token such as ext: should be used for the Extension schema targetNamespace (eg. xmlns:ext="http://{my namespace}/.../extension/...")</li> <li>• Declare the GJXDM schema, subset schema or constraint schema namespace (eg. xmlns:j="http://www.it.ojp.gov/jxdm/{jxdm version}")</li> <li>• xmlns:xsd</li> </ul> </li>   <li>• &lt;!-- ===== Imports ===== --&gt; <ul style="list-style-type: none"> <li>• External GJXDM reference , subset schema or constraint schema import namespace and relevant schemaLocation</li> </ul> </li>   <li>• &lt;!-- ===== Extended/Restricted GJXDM Complex Types and Simple Types = --&gt;</li> <li>• &lt;!-- ===== in alphabetized order xsd:TypeDefinitions ===== --&gt; <ul style="list-style-type: none"> <li>• Complex and Simple Types</li> </ul> </li>   <li>• &lt;!-- ===== Element Declarations in alphabetized order ===== --&gt;</li> </ul>
[GXS3]	<p>GJXDM Document schema MUST conform to the following physical layout as applicable:</p> <ul style="list-style-type: none"> <li>• XML Declaration</li>   <li>• &lt;!-- ===== Required Documentation Comments Block ===== --&gt;</li> <li>• &lt;!-- ===== Name (common): ..... ===== --&gt;</li> <li>• &lt;!-- ===== Authoring agency/jurisdiction/generation date: ..... ===== --&gt;</li>   <li>• &lt;!-- ===== Description of business usage: ..... ===== --&gt;</li>   <li>• &lt;!-- ===== xsd:schema Element With Namespaces Declarations ===== --&gt;</li> </ul>

	<ul style="list-style-type: none"> <li>• xsd:schema element followed by Namespace Declarations in this order: <ul style="list-style-type: none"> <li>• Target namespace for Document schema</li> <li>• No Default namespace, a token such as doc: or rap: (for a rapsheet) should be used for the Document schema targetNamespace</li> <li>• Declare the “optional” GJXDM extension schema namespace (eg. xmlns:ext=”http://{my namespace}.../extension/...”)</li> <li>• Declare the GJXDM schema, subset schema or schema constraint namespace (eg. xmlns:j=”http://www.it.ojp.gov/jxdm/{jxdm version}”)</li> <li>• xmlns:xsd</li> </ul> </li> <li>• &lt;!-- ===== Imports ===== --&gt; <ul style="list-style-type: none"> <li>• External “optional” GJXDM extension schema import namespace and schemaLocation</li> <li>• External GJXDM schema, subset schema or constraint schema import namespace and schemaLocation</li> </ul> </li> <li>• &lt;!-- ===== Root Element ===== --&gt; <ul style="list-style-type: none"> <li>• Root Element Declaration</li> <li>• Root Element Type Definition</li> </ul> </li> <li>• &lt;!-- ===== Type Definition ===== --&gt; <ul style="list-style-type: none"> <li>• Define Root Type; extend from j:DocumentType; where complexType name=”{Root Element Name} Type” (eg. for Root Element Name “CitationDocument” type=”doc:CitationDocumentType” then the complexType name = “CitationDocumentType” which is the Root Element Name suffixed with the word “Type”).</li> </ul> </li> </ul>
[GXS4]	<p>The root element in all GJXDM information exchange Schema modules MUST contain the following declaration:  “xmlns:xsd=http://www.w3.org/2001/XMLSchema.”</p>
[GXS5]	<p>GJXDM information exchange schema developers MAY provide a run-time schema devoid of documentation in addition to the fully annotated version.</p>
[GXS6]	<p>GJXDM defined xsd:simpleTypes SHOULD be used as the base for any user-defined simpleTypes via extension or restriction to the GJXDM simpleType.</p>

[GXS7]	The <code>xsd:SubstitutionGroups</code> feature <b>MUST NOT</b> be used.
[GXS8]	The <code>xsd:final</code> attribute <b>MUST</b> be used to control extensions.
[GXS9]	The <code>xsd:block</code> attribute <b>SHOULD</b> be used to restrict use of “type substitution” and “element substitution” in XML instance documents.
[GXS10]	<code>xsd:notations</code> <b>MUST NOT</b> be used.
[GXS11]	The <code>xsd:all</code> element <b>MUST NOT</b> be used.
[GXS12]	The <code>xsd:redefine</code> element <b>MUST NOT</b> be used.
[GXS13]	The <code>xsd:choice</code> element <b>SHOULD NOT</b> be used where customization and extensibility are a concern.

2137

<b>A.9 General XML Schema Rules</b>	
[GXS14]	GJXDM designed schema <b>MAY</b> use <code>xsd:appinfo</code> . If used, <code>xsd:appinfo</code> <b>MUST</b> only be used to convey non-normative information.  Note: <code>appinfo</code> is a recent addition to GJXDM in Version 3.0.2
[GXS15]	Complex Type extension and Simple Type restriction <b>MAY</b> be used where appropriate.
[GXS16]	All GJXDM document schema <b>MUST NOT</b> declare new ComplexTypes or SimpleTypes within the document schema. Any ComplexTypes and SimpleTypes needed by the document schema must be imported from external schema.

2138

<b>A.10 Instance Document Rules</b>	
[IND1]	All GJXDM instance documents <b>MUST</b> validate to a corresponding Document schema.
[IND2]	All GJXDM instance documents <b>MUST</b> always identify their character encoding with the XML declaration.

[IND3]	In conformance with ISO/IETF/ITU/UNCEFACT Memorandum of Understanding Management Group (MOUMG) Resolution 01/08 (MOU/MG01n83), all GJXDM XML SHOULD be expressed using UTF-8.
[IND4]	All GJXDM instance documents MUST contain the following namespace declaration in the root element: <code>xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"</code>
[IND5]	GJXDM conformant instance documents MUST NOT contain an element devoid of content unless explicitly indicated by the <code>xsi:nil="true"</code> attribute
[IND6]	The absence of a construct or data in a GJXDM instance document MUST NOT carry meaning.

2139  
2140

## A.11 Mapping Documentation Rules

[MAP1]	All Domain Model and GJXDM Mapping Documentation MUST include the required content identified in the GJXDM Domain Model Mapping table. The table MAY contain the specified optional content.
--------	--

2141  
2142

## A.12 Modeling Constraints Rules

[MDC1]	Mixed content MUST NOT be used except where contained in an <code>xsd:documentation</code> element.
--------	---

2143  
2144

## A.13 Namespace and Schema Filename Rules

[NMS1]	Every GJXDM information exchange schema MUST have a namespace declared using the <code>xsd:targetNamespace</code> attribute.
[NMS2]	Every GJXDM information exchange schema version MUST have its own unique namespace.



[NMS3]	<p>A GJXDM schema subset and constraint schema MUST declare the same <code>xsd:targetNamespace</code> as the GJXDM baseline schema.</p> <p>For example:  “<code>http://www.it.ojp.gov/jxdm/3.0.1</code>” is the required <code>targetNamespace</code> for GJXDM version 3.0.1</p>
[NMS4]	GJXDM namespaces MUST only contain GJXDM conformant schema modules.

2145

<h2 style="color: blue;">A.13 Namespace and Schema Filename Rules</h2>	
[NMS5]	<p>GJXDM published namespaces MUST never be changed. The namespace names for GJXDM reference schema releases are of the form:</p> <p><code>http://www.it.ojp.gov/jxdm/{major version . minor release . revision}</code></p> <p>For example the following namespace <code>http://.../jxdm/3.1.1</code> would be major-release 3 and minor-release 1.1 of the GJXDM schema.</p>
[NMS6]	<p>Each GJXDM imported <code>CodeList</code> schema MUST be maintained in a separate namespace. The proxy Codelist URL MUST be of the form:</p> <p>“<code>http://www.it.ojp.gov/jxdm/{GJXDM version}/proxy/{external original codelist name}/{original codelist version}</code>”.</p> <p>For example,  <code>xmlns:j-usps = “http://.../jxdm/3.0.2/proxy/usps_states/1.0”</code> represents the United States Postal Services States Code table version 1.0 which is imported into GJXDM schema version 3.0.2 . The proxy namespace “j-usps” is a concatenation of the Justice namespace ‘j’ and the original source code-list namespace “usps” forming the general rule for Proxy Namespace as “j-xsd”; where <code>xsd</code> is the same as the “original” codelist namespace.</p> <p>Proxy schemas provide an intermediary between external namespaces and the GJXDM, and use of Proxies guarantee SuperType metadata on elements based on entities from external namespaces.</p>
[NMS7]	<p>Each schema, for a GIEP, MUST be maintained in a separate namespace but will share a common group path.</p> <p>That GIEP group path MUST be of the form:</p> <p>“<code>http:// &lt;IEP owner domain name&gt;/&lt;IEP name&gt;/&lt;IEP version&gt;/</code>”.</p>

	<p>For example,  xmlns = “http://www.myDomainName.com/Citation/1.0/” represents a local namespace copy of a GIEP group of schemas that support the exchange or representation of a Citation reference document. The version of 1.0 is assigned to the GIEP namespace by the provider of the reference document.</p> <p>Note: All associated subset, document, constraint and extension schemas must be placed within this path. If any of the associated schemas, within the GIEP group change, the GIEP version MUST change.</p>
[NMS8]	<p>Document and Extension schema’s xsd:import element(s) MUST use a <u>relative URL</u> to locate the imported GJXDM Subset schema set.</p> <p>The GJXDM Subset path MUST be of the form:</p> <p>“http:// &lt;IEP owner domain name&gt;/&lt;IEP name&gt;/&lt;IEP version&gt;/subset/jxdm/&lt;GJXDM version&gt;/”.</p> <p>For example:</p> <p>xmlns=“http://www.myDomainName.com/Citation/1.0/subset/jxdm/3.0/“</p> <p>represents the location of a conformant subset of the full GJXDM v3.0 schema for a version 1.0 “Citation” IEP</p> <p>The relative URL MUST be valid within the structure of the GIEPD ZIP file specified in NMS17</p>
[NMS9]	<p>All IEP’s utilizing a GJXDM subset schema MUST produce the same files and filenames as the GJXDM Subset Schema Generator Tool (SSGT)</p> <p>For example:</p> <p>The SSGT generates “jxdm.xsd” as the standard filename for any GJXDM subset and produces a set of want-lists with a specified standard file structure. These SSGT artifacts must be produced for the IEP even if another software tool is used to produce them.</p>
[NMS10]	<p>A GJXDM IEP Extension schema targetNamespace MUST be of the form:</p> <p>“http:// &lt;IEP owner domain name&gt;/&lt;IEP name&gt;/&lt;IEP version&gt;/&lt;IEP schema type[-suffix]&gt;/”.</p> <p>The “IEP name” and “IEP version” components of this identifier are chosen by the IEP workgroup (or chartering governance body). The IEP version component must follow the namespace versioning rules stated elsewhere in this specification. The “IEP name” component must be declared by the workgroup in the IEP overview document.</p>

	<p>The “IEP schema type” component of this identifier MUST be “extension” for the IEP’s extension namespace. If the IEP has multiple extension namespaces, the IEP workgroup MUST choose appropriate “suffix” namespace name values and document them in the IEP overview document.</p> <p>For example,  xmlns:ext = “http://www.myDomainName.com/Citation/1.0/extension” represents the namespace for a version 1.0 Citation extension schema containing user-defined Types derived from extending or restricting GJXDM schema complexTypes and simpleTypes.</p>
[NMS11]	<p>The GJXDM Extension schema module MUST have a schema filename of the following form:</p> <p>&lt;IEP name&gt;-&lt;IEP schema type&gt; [&lt;-suffix&gt;].xsd</p> <p>where  &lt;IEP name&gt; = &lt;IEP name&gt; defined in the Extension schema namespace  &lt;IEP schema type&gt; = extension  [suffix] = the [-suffix] optional name or number in the Extension schema namespace to support multiple extension schemas per IEP.</p> <p>For example, the IEP GJXDM extension schema filename for xmlns = “http://www.myDomainName.com/Citation/1.0/extension-joe” would be:</p> <p>Citation-extension-joe.xsd</p> <p>represents an extension schema for version 1.0 of a Citation IEP where &lt;IEP name&gt; = “Citation”, &lt;IEP schema type&gt; = “extension” and &lt;suffix&gt; = “joe”.</p>
[NMS12]	<p>A GJXDM IEP Document schema targetNamespace MUST be of the form:</p> <p>“http://&lt;IEP owner domain name&gt;/&lt;IEP name&gt;/&lt;IEP version&gt;/&lt;IEP schema type[-suffix]&gt;”.</p> <p>The “IEP name” and “IEP version” components of this identifier are chosen by the IEP workgroup (or chartering governance body). The IEP version component must follow the namespace versioning rules stated elsewhere in this specification. The “IEP name” component must be declared by the workgroup in the IEP overview document.</p> <p>The “IEP schema type” component of this identifier MUST be “document” for the</p>

	<p>IEP’s document namespace. If the IEP has multiple document namespaces, the IEP workgroup MUST choose appropriate “suffix” namespace name values and document them in the IEP overview document.</p> <p>For example,  xmlns = “http://www.myDomainName.com/Citation/1.0/document” represents the namespace for a version 1.0 Citation IEP document schema</p>
[NMS13]	<p>The GJXDM Document schema module MUST have a schema filename of the following form:</p> <p>&lt;IEP name&gt;-&lt;IEP schema type&gt; [&lt;-suffix&gt;].xsd</p> <p>where  &lt;IEP name&gt; = &lt;IEP name&gt; in the Document schema namespace  &lt;IEP schema type&gt; = document  [suffix] = the [-suffix] optional name or number in the Document schema namespace to support multiple document schemas per IEP.</p> <p>For example, the IEP GJXDM document schema filename for xmlns = “http://www.myDomainName.com/Citation/1.0/document” would be:</p> <p>Citation-document.xsd</p> <p>represents a document schema for version 1.0 of a Citation IEP where &lt;IEP name&gt; = “Citation” , &lt;IEP schema type&gt; = “document” and &lt;suffix&gt; = option not used.</p>
[NMS14]	<p>The Document and Extension schema’s xsd:import element(s) MUST use a <u>relative</u> URL to locate the imported GJXDM Constraint schema.</p> <p>The relative URL MUST be valid within the structure of the GIEPD ZIP file specified in NMS17</p>
[NMS15]	<p>An IEP MAY have zero or one Constraint schema defined. The GJXDM Constraint schema module MUST have a schema filename of the following form:</p> <p>&lt;IEP name&gt;-&lt;IEP schema type&gt; .xsd</p> <p>where  &lt;IEP name&gt; = &lt;IEP name&gt; in the Document or Extension schema namespace  &lt;IEP schema type&gt; = constraint</p> <p>For example, if the IEP document schema namespace is xmlns =</p>

	<p>http://www.myDomainName.com/Citation/1.0/document, THEN the constraint schema filename would be:</p> <p>Citation-constraint.xsd</p> <p>represents a constraint schema for version 1.0 of a Citation IEP where &lt;IEP name&gt; = "Citation" and &lt;IEP schema type&gt; = "constraint"</p>
<p>[NMS16]</p>	<p>GIEPD artifacts MUST be combined into a ZIP file.</p> <p>The ZIP file MUST have the filename: &lt;IEP name&gt;-&lt;IEP version&gt;.zip</p> <p>Where &lt;IEP name&gt; &amp; &lt;IEP version&gt; = &lt;IEP name&gt; &amp; &lt;IEP version&gt; in the IEP document namespace.</p> <p>For example, if IEP GJXDM xmlns = http://www.myDomainName.com/Citation/1.0/document , THEN the GIEPD ZIP filename would be:</p> <p>Citation-1.0.zip</p> <p>represents an GIEPD ZIP file for version 1.0 of a Citation IEP where &lt;IEP name&gt; = "Citation" and &lt;IEP version&gt; = "1.0"</p>
<p>[NMS17]</p>	<p>The GIEPD ZIP file MUST have the following structure and contents.</p> <p>The root directory in the ZIP archive must contain the following files and directories:</p> <ul style="list-style-type: none"> <li>• The GIEPD Overview document file, in a suitable file format, and named "GIEPD Overview" with a file extension corresponding to the format. The "GIEPD Overview" MUST include the filename of the GIEPD ZIP file.</li> <li>• A directory named "domain model artifacts"</li> <li>• A directory named "mapping artifacts"</li> <li>• A directory named "schemas"</li> <li>• A directory named "sample instances"</li> </ul> <p>The directory named "domain model artifacts" MUST contain all artifacts related to the domain model of the IEP, as discussed in section 3 of this specification.</p>

	<p>The directory named “mapping artifacts” MUST contain all artifacts related to the mapping of the domain model to GJXDM, as discussed in section 3 of this specification.</p> <p>The directory named “sample instances” MUST contain one or more sample XML instances that are valid against the document, extension, constraint, and subset schemas in the IEP.</p> <p>Each sample XML instance:</p> <ol style="list-style-type: none"> <li>1) MUST associate referenced IEP namespaces by using the xsi:schemaLocation attribute on the XML Instance root element;</li> <li>2) MUST use the xsi:schemaLocation attribute</li> <li>3) MUST use a <u>relative</u> URL, valid within the IEP structure documented here, to locate the schema for each namespace.</li> </ol> <p>The directory named “schemas” MUST contain the following:</p> <ul style="list-style-type: none"> <li>• A document schema file (see rule NMS12 and NMS13 for filename rules)</li> <li>• Extension schema files(s) (see rule NMS10 and NMS11 for filename rules)</li> <li>• A constraint schema file, if the IEP uses a constraint schema (see rule NMS14 and NMS15 for filename rule)</li> <li>• A directory named “subset” that contains the <u>subset schema set</u>; the sub-directory structure underneath the “subset” directory must match the directory structure of the GJXDM distribution version being used. <i>Note that the subset schema is not a single schema; rather, it is a directory structure that contains many related schemas.</i></li> </ul> <p>To the extent that unzipped schemas in the IEP import each other, the schemaLocation attribute for each schema’s xsd:import element(s) MUST use a <u>relative</u> URL to locate the imported schema. The relative URL MUST be valid within the structure of the ZIP file specified above.</p>
[NMS18]	<p>Also, if adopted, the UBL core-components parameters schema (place where schema annotation/documentation elements are defined including context drivers) could be added back into this mndr. <a href="http://docs.oasis-open.org/ubl/cd-UBL-1.0/xsd/common/UBL-CoreComponentParameters-1.0.xsd">http://docs.oasis-open.org/ubl/cd-UBL-1.0/xsd/common/UBL-CoreComponentParameters-1.0.xsd</a> Namespace could be referenced directly from extension schema.</p>

2146  
2147  
2148

A.14	Root Element Declaration Rules
[RED1]	Every GJXDM instance document must use the global element defined as the root element in the schema as its root element.

2149

2150

2151

<h2 style="color: blue;">A.15 Schema Structure Modularity Rules</h2>	
[SSM1]	GJXDM Schema MAY be split into a smaller subset schema, but only one GJXDM subset can be created for a given document schema, because the GJXDM schema must reside in one and only one namespace.
[SSM2]	Imported schema modules MUST be fully conformant with GJXDM information exchange schema naming and design rules.
[SSM3]	GJXDM schema modules MUST be treated as external schema modules of the document schema.
[SSM4]	xsd:include MUST NOT be used in development of IEP's because this MNDR does not support use of Internal Schema modules.
[SSM5]	GJXDM schema module(s) MAY be created for reusable components.

2152

2153

<h2 style="color: blue;">A.16 Standards Adherence rules</h2>	
[STA1]	All Global JXDM information exchange schema design rules MUST be based on the W3C XML Schema Recommendations: XML Schema Part 1: Structures and XML Schema Part 2: Datatypes.
[STA2]	All Global JXDM information exchange schema and messages MUST be based on the W3C suite of technical specifications holding recommendation status.

2154

2155

2156

<h2 style="color: blue;">A.17 Versioning Rules</h2>
---

[VER1]	Every GJXDM information exchange schema and schema module <u>major version</u> committee <i>draft</i> MUST have a version number of the form:  <major>.0[.<revision>]
[VER2]	Every GJXDM Information exchange Schema and schema module <u>major version</u> OASIS <i>Standard</i> MUST have a version number of the form:  <major>.0
[VER3]	Every <u>minor version</u> release of a GJXDM Information exchange schema or schema module <i>draft</i> MUST have a version number of the form:  <major >.<non-zero>[.<revision>]

2157

## A.17 Versioning Rules Continued

[VER4]	Every <u>minor version</u> release of a GJXDM information exchange schema or schema module <i>Standard</i> MUST have a version number of the form:  <major >.<non-zero>
[VER5]	For GJXDM information exchange schema <u>minor version</u> changes, the <IEP name> MUST NOT change.
[VER6]	For every GJXDM information exchange schema and schema module, the <u>major version</u> number MUST be a sequentially assigned, incremental number greater than zero.
[VER7]	For every GJXDM information exchange schema and schema module, the <u>minor version</u> number MUST be a sequentially assigned, incremental non-negative integer.

2158

2159

[VER8]	GJXDM information exchange schema and schema module <u>minor version</u> changes MUST not break semantic compatibility with prior versions; nor may they break existing document instances that are based on any earlier minor version of the last major version. For example, an instance document build on a 1.1 minor version must be able to be processed by any later minor release, for example, a 1.9 version. Minor versions maintain forward compatibility.
[VER9]	GJXDM information exchange schema and schema module <u>major version</u> changes MAY break semantic and/or structural compatibility with prior versions. No backward compatibility is guaranteed.



[VER10??]	<p>(5/13/05: Need to include rule(s) explaining how to use revision; Sylvia Webb will provide additional clarification..</p> <p>NOTES:</p> <ul style="list-style-type: none"><li>- To support pre-release work (alpha suffix)</li><li>- Post release, useful when there have been no functional/behavior changes, but changes in comments, documentation, or syntax definitions (numeric suffix).</li></ul>
-----------	---

---

2160 **Appendix B. Approved Acronyms and**  
2161 **Abbreviations**

2162 The following Acronyms and Abbreviations have been approved by the GJXDM  
2163 MNDR Subcommittee for GJXDM use:

2164 "Automated Fingerprint Identification System" *must* appear as "AFIS".

2165 "American National Standards Institute" *must* appear as "ansi".

2166

2167 "Commercial Motor Vehicle" *must* appear as "CMV"

2168

2169 "Deoxyribonucleic Acid" *must* appear as "DNA".

2170

2171 "Federal Bureau of Investigation" *must* appear as "FBI".

2172 "Federal Information Processing Standards" *must* appear as "FIPS".

2173 "Foreign Government Issue" *must* appear as "FGI".

2174

2175 "Hazardous Materials" *must* appear as "HazMat".

2176

2177 "Intelligence Community" *must* appear as "IC"

2178 "International Standards Organization" *must* appear as "ISO"

2179 "Manufacturers Certificate of Origin" *must* appear as "MCO"

2180 "Manufacturers Suggested Retail Price" *must* appear as "MSRP"

2181

2182 "Military Grid Reference System" *must* appear as "MGRS"

2183

2184 "Monthly Arrest and Citation Register" *must* appear as "MACR".

2185

2186 "National Crime Information Center" *must* appear as "NCIC".

2187

2188 "National Incident Based Reporting System" *must* appear as "NIBRS".

2189

2190 "Originating Agency Indicator" *must* appear as "ORI".

2191

2192 "Police Official Standards and Training" *must* appear as "POST".

2193

2194 "Scars, Marks & Tatoos " *must* appear as "SMT".

2195

2196 "Social Security Number" *must* appear as "SSN".

2197

2198 "Uniform Crime Report" *must* appear as "UCR".

2199

2200 "Uniform Resource Identifier" *must* appear as "URI".

2201

2202 "United States" *must* appear as "US".

2203

2204 "Universal Transverse Mercator" *must* appear as "UTM".

2205

2206

2207 This list will henceforth be maintained by the GJXDM XSTF committee, and  
2208 additions included in current and future versions of the GJXDM standard will be  
2209 maintained and published separately.

## Appendix C. (Informative) Technical Terminology

Ad hoc schema processing	Doing partial schema processing, but not with official schema validator software; e.g., reading through schema to get the default values out of it.
Aggregate Business Information Entity (ABIE)	A collection of related pieces of business information that together convey a distinct business meaning in a specific Business Context. Expressed in modeling terms, it is the representation of an Object Class, in a specific Business Context.
Application-level validation	Adherence to business requirements, such as valid account numbers.
Assembly	Using parts of the library of reusable GJXDM components to create a new kind of business document type.
Business Context	Defines a context in which a business has chosen to employ an information entity. The formal description of a specific business circumstance as identified by the values of a set of <i>Context Categories</i> , allowing different business circumstances to be uniquely distinguished.
class	A description of a set of objects that share the same attributes, operations, methods, relationships, and semantics. A class may use a set of interfaces to specify collections of operations it provides to its environment. See interface.

class diagram	Shows static structure of concepts, types, and classes. Concepts show how users think about the world; types show interfaces of software components; classes show implementation of software components. (OMG Distilled) A diagram that shows a collection of declarative (static) model elements, such as classes, types, and their contents and relationships. (Rational Unified Process)
classification scheme	This is an officially supported scheme to describe a given <i>Context Category</i>
Common attribute	An attribute that has identical meaning on the multiple elements on which it appears. A common attribute might or might not correspond to an XSD global attribute.
component	One of the individual entities contributing to a whole.
Concept Map	Simple graphical models of relationships between concepts
context	Defines the circumstances in which a Business Process may be used. This is specified by a set of Context Categories known as Business Context. (See Business Context.)
context category	A group of one or more related values used to express a characteristic of a business circumstance.
Domain model	A graphical representation of the consensus of a group of business subject-matter experts as to the structure and content of an exchange document.
Domain Model Element	Any symbol in a domain model used to describe a part of the structure of an exchange document. Elements include symbols used to describe individual data elements, entire named data structures (or collections of data elements), and relationships between data elements or structures.

Document schema	A schema document corresponding to a single namespace, which is likely to pull in (by importing) schema modules.
instance	An individual entity satisfying the description of a class or type.
Leaf element	An element containing only character data (though it may also have attributes). Note that, because of the XSD mechanisms involved, a leaf element that has attributes must be declared as having a complex type, but a leaf element with no attributes may be declared with either a simple type or a complex type.
Lower-level element	An element that appears inside a business message. Lower-level elements consist of intermediate and leaf level.
Mixed Content	An XML <element> that contains both “string data” and other <element> data is defined as having mixed content.
Object Class	The logical data grouping (in a logical data model) to which a data element belongs (ISO11179).
(XML) Schema	An XML Schema consists of components such as type definitions and element declarations. These can be used to assess the validity of well-formed element and attribute information items (as defined in <a href="#">[XML-Infoset]</a> ), and furthermore may specify augmentations to those items and their descendants.
Schema module	A collection of XML constructs that together constitute an XSD conformant schema. Schema modules are intended to be used in combination with other XSD conformant schema.

Schema Processing	Schema validation checking plus provision of default values and provision of new info set properties.
Schema Validation	Adherence to an XSD schema.
Semantic	Relating to meaning in language; relating to the connotations of words.
Subset Schema Generation Tool (SSGT) Want List	<p>“The wantlist is the xml file that SSGT uses to record and maintain the state of the user's schema subset. When the user generates a schema subset package (and it is assumed that the user is finished using the SSGT), a wantlist is also generated and saved with the user's work. The wantlist is a specification for the user's schema subset and can be reloaded into SSGT to continue editing.”</p> <p>In short, the WantList is simply an XML-formatted list of elements and types requested for inclusion in a Subset Schema, plus any elements and types included in the Extension Schema.</p>
Top-level element	An element that encloses a whole business message. Note that GJXDM business messages might be carried by messaging transport protocols that themselves have a higher-level XML structure. Thus, a GJXDM top-level element is not necessarily the root element of the XML document that carries it.
type	<p>Description of a set of entities that share common characteristics, relations, attributes, and semantics.</p> <p>A stereotype of class that is used to specify an area of instances (objects) together with the operations applicable to the objects. A type may not contain any methods.</p>

---

2211 **Appendix D. (Informative) Example GIEPD(s)**

2212 TBD – Need to reference and/or insert portions of GIEPD(s) conformant to this MNDR  
2213 specification.

2214

2215 Use OASIS Court Filing IEPD's???



---

## 2216 **Appendix E. (Informative) Ongoing Work Items**

- 2217 Appendix D needs to be populated with references and samples of compliant GIEPD's to  
2218 complete this first version of the MNDR
- 2219
- 2220 MNDR needs vetting with the full IJTC and Organizations referenced in the Status Section on the  
2221 cover page of this specification draft.
- 2222
- 2223 In the June-August 2006 timeframe, the MNDR needs to be evaluated against the new NIEM and  
2224 "component based version of GJXDM" to determine the scope of changes to this MNDR that  
2225 would be required to comply with NIEM 0.x and GJXDM post version 3.0.3.
- 2226
- 2227 Committee needs to evaluate need for developing and hosting additional workshops for creating  
2228 new GIEPDs
- 2229
- 2230 Committee needs to evaluate whether it wants to commit to an effort to review existing GIEPD's  
2231 to identify the degree of common use of GJXDM components across GIEPD's and to publish  
2232 findings/recommendations.
- 2233

2234

---

## Appendix F. (Informative) Acknowledgments

2235

2236 The following individuals were members of the committee during the approval of this draft:

2237

**Participants:**

2238

2239

Jim Beard, Individual Member

2240

Donald Bergeron, Reed Elsevier

2241

James Cabral, MTG Management Consultants, LLC.

2242

Scott Came, Individual Member

2243

Tom Carlson, National Center for State Courts

2244

Rolly Chambers, American Bar Association

2245

James Bryce Clark, OASIS

2246

Thomas Clarke, National Center for State Courts

2247

Scott Edson, LA County Information Systems Advisory Body

2248

Paul Embley, Chairperson – Global XML Structure Task Force (XSTF)

2249

Robin Gibson, Missouri Office of State Courts Administrator

2250

David Goodwin, Maricopa County

2251

Aaron Gorrell, URL Integration

2252

John Greacen, Individual Member

2253

Jim Harris, National Center for State Courts

2254

Marcus Leon, LA County Information Systems Advisory Body

2255

Rex McElrath, Judicial Council of Georgia

2256

John Messing, American Bar Association

2257

Ellen Perry, MTG Management Consultants, LLC.

2258

Catherine Plummer, Search Group, Inc.

2259

John Ruegg, LA County Information Systems Advisory Body

2260

Nancy Rutter, Maricopa County

2261

Christopher Smith, California Administrative Office of the Courts

2262

Marguerite Soto, LA County Internal Services Department

2263

Eric Tingom, Individual Member

2264

Winfield Wagner, Crossflo Systems Inc.

2265

Sylvia Webb, Individual Member

2266

Roger Winters, Washington State Administrator for the Courts

2267

## Appendix G. (Informative) Revision History

2268

[This appendix is optional, but helpful. It should be removed for specifications that are at OASIS Standard level.]

2269

2270

Rev	Date	By Whom	What
Section 2 wd-01	2005-02-21	Tom Carlson	Initial version
Section 2 wd-02	2005-04-10	IJIS TC	Comments
Section 2 wd-03	2005-11-04	Tom Carlson	Incorporation of comments

2271

Rev	Date	By Whom	What
Section 6 wd-.01	2005-02-10	John Ruegg	Initial version
Section 6 wd-.02	2005-03-10	John Ruegg	Incorporates decision to omit UBL component terminology for this first version
Sections 1 thru 7 as separate documents	2005-11-22	John Ruegg Tom Carlson Scott Came Ellen Perry	Prior Version of the MNDR Sections can be found at <a href="http://www.oasis-open.org/apps/org/workgroup/legalxml-intj-exmndr/documents.php">http://www.oasis-open.org/apps/org/workgroup/legalxml-intj-exmndr/documents.php</a>
wd-ijtc-MNDR-1.0.0	2006-04-20	John Ruegg Marcus Leon Marguerite Soto	Edited and Combined single document version of MNDR draft.