May 20, 2008

NIEM USER GUIDE

VOLUME 1

URI: http://reference.niem.gov/niem/guidance/user-guide/vol1/

NATIONAL INFORMATION EXCHANGE MODEL | NIEM.gov

U.S. Department of Justice
Office of Justice Programs
810 Seventh Street, NW
Washington, DC 20531

The Honorable Michael B. Mukasey
Attorney General

The Honorable Mark R. Filip
Deputy Attorney General

The Honorable Kevin O'Connor
Associate Attorney General

The Honorable Jeffrey L. Sedgwick
Acting Assistant Attorney General

The Honorable Domingo S. Herraiz
Director, Bureau of Justice Assistance

Office of Justice Programs
World Wide Web Home Page
www.ojp.usdoj.gov

Bureau of Justice Assistance
World Wide Web Home Page
www.ojp.usdoj.gov/BJA

For grant and funding information, contact
U.S. Department of Justice, Office of Justice Programs
Funding Opportunities
http://www.ojp.usdoj.gov/funding

BJA Bureau of Justice Assistance

This project was supported by Grant No. 2007-RG-CX-K021 awarded by the Bureau of Justice Assistance. The Bureau of Justice Assistance is a component of the Office of Justice Programs, which also includes the Bureau of Justice Statistics, the National Institute of Justice, the Office of Juvenile Justice and Delinquency Prevention, and the Office for Victims of Crime. Points of view or opinions in this document are those of the author and do not represent the official position or policies of the United States Department of Justice.

Table of Contents

List of Figures

List of Tables

1 Acknowledgements

The Bureau of Justice Assistance (BJA) and the NIEM Project Management Office (PMO) would like to thank the many individuals who contributed to the development of this document.

This *User Guide* would not have been possible without the tremendous amount of work and assistance of all the authors, editors, and volunteer reviewers. BJA and the NIEM PMO are especially grateful to members of the following committees and organizations for their collaborative efforts:

- Georgia Tech Research Institute (GTRI)
- Global Justice Information Sharing Initiative (Global), XML Structure Task Force
- IJIS Institute XML Advisory Committee
- National Center for State Courts (NCSC)
- NIEM PMO Committee Members and Volunteers
- SEARCH, The National Consortium for Justice Information and Statistics
- U.S. Department of Homeland Security (DHS)

NIEM User Guide Development Team

Mr. Philip Ardire
Tetrus Consulting

Ms. Carrie Boyle
Bearingpoint

Ms. Andrea Cafarelle
Tetrus Consulting

Mr. Tom Carlson
Tom Carlson Consulting

Mr. Scott Chontow
Bearingpoint

Mr. Donald Gabbin
IJIS Institute

Mr. Ashwini Jarral
IJIS Institute

Mr. Chandra Jonelagadda
Tetrus Consulting

Mr. Vivek Misra
URL Integration

Ms. Lisa Neal
IJIS Institute

Mr. Prem Neelakanta
Analyst International

Mr. Vipul Patel
IJIS Institute

Ms. Catherine Plummer
Information Sharing LLC

Mr. Sharad Rao
Tetrus Consulting

Mr. Dave Usery
URL Integration

2 Introduction

The National Information Exchange Model (NIEM) is a partnership of the U.S. Department of Justice (DOJ) and the U.S. Department of Homeland Security (DHS). It is designed to develop, disseminate, and support enterprise-wide information sharing standards and processes, providing a framework for communities of interest throughout the nation to collaborate and share critical information effectively. NIEM enables information sharing across all levels of government, including Federal, state, local, and Tribal governments, and is supportive of both day-to-day operations and real-time emergency situations.[1]

The *NIEM User Guide Volume I* provides detailed guidance about how to develop information exchanges utilizing this model. It provides a detailed description of the rationale for the creation of NIEM, an architectural overview, and technical concepts derived from NIEM Program Management Organization (PMO) documentation. This volume takes the reader further into a methodology for defining the business requirements of the information exchange, as well as creating an Information Exchange Package Documentation (IEPD) that fully specifies the exchange in conformance with NIEM guidelines. Also included is information about tools to assist development, resources for education and peer assistance, emerging technologies and how they relate to NIEM, and the national partners that bring it all together.

The primary audience for this document is engineers and developers who intend to use the NIEM standard to support interagency information sharing. The reader is expected to have an understanding of the concepts of object oriented design, UML, and XML technologies.

This ***NIEM User Guide*** consists of the following sections:

Section		Description
1	Acknowledgements	This section lists the individuals and agencies that were involved in the creation of the *NIEM User Guide*.
2	Introduction	This section provides an overview of the document and describes the notations used throughout.
3	The Need for Information Sharing	This section provides an overview of the need for information sharing, some key concepts required for information sharing and an overview of NIEM.
4	NIEM Overview	This section provides an overview of the NIEM data model.
5	NIEM Data Model Concepts NIEM Data Model Concepts	This section discusses data model concepts of the NIEM model. It details types, properties, namespaces, and other concepts in NIEM.
6	NIEM Data Model Content	This section describes the content of the NIEM data model. It identifies the current domains that the NIEM data model covers.
7	Building NIEM-Conformant Data Exchanges	This section discusses the suggested methodology for building NIEM-conformant data exchanges.
8	IEPD Artifacts	This section identifies the artifacts that may be produced as a result of an IEPD development.
9	IEPD Metadata	This section defines the metadata that must be created to enable this IEPD to be discovered by other individuals when they search the IEPD repository.

[1] http://www.niem.gov/whatIsNiem.php.

Section	Description
Appendix A: Data Model Conformance Guidelines	This appendix discusses the conformance guidelines for NIEM.
Appendix B: NIEM Tools	This appendix discusses the tools available to the reader to develop NIEM-conformant IEPDs.
Appendix C: NIEM Resources	This appendix discusses the resources that are available to the reader for obtaining additional information about NIEM.
Appendix D: NIEM Constructs vs. GJXDM Constructus	This appendix briefly demonstrates some differences between the NIEM and GJXDM constructs.
Appendix E: Glossary of Terms and Acronyms	This appendix provides definitions for terms and acronyms that appear in bold throughout this document.
Appendix F: NIEM 2.0 Reference Schemas	This appendix presents the code lists and external schemas that are utilized in NIEM.

Table 1: About This Document.

2.1 Typographical Conventions Used in This Document

Throughout this document, the following typographical conventions provide you with clues as to the significance or context of the material being discussed.

This is an alert. When you see information presented in this manner, pay special attention—information presented in this manner is critical to your understanding of the concept being discussed.

This is a note. Information presented in this manner is important but not critical to your understanding of the concept being discussed.

```
Example code appears in this typeface.
```

3 The Need for Information Sharing

Information sharing involves the business processes, policies, procedures, architecture, and governance that support effective decision-making and mission-focused actions by providing timely, accurate, and relevant information to the appropriate individuals across all levels of government. Essentially, it is this need that makes the business case for the creation and use of a standard such as NIEM.

A variety of emergency situations in recent years have demonstrated the potentially tragic consequences that can result from the inability of jurisdictions and agencies to effectively share information. Terrorist attacks, natural disasters, and large-scale organized criminal incidents serve as case studies that reveal weaknesses in our nation's information sharing capabilities. Moreover, enterprise-wide information sharing is also required to support the critical day-to-day operations of federal, state, local, and tribal officials.

Current information collection and dissemination practices have not been planned as part of a unified national strategy but, rather, have evolved incrementally over time to meet specific one-off challenges as they have surfaced. Agencies are often unable to effectively share information in a timely, secure manner, and there can be fundamental differences in the nature and understanding of information that can be shared between agencies. While sharing does occur today, it often occurs to a limited degree, or within stovepipe information systems. A tremendous quantity of information that should be shared is still not effectively done, nor is this information utilized effectively among relevant communities of interest (COIs).

3.1 Challenges to Information Sharing

Previous efforts to improve this situation have been beset by a multitude of challenges. These challenges include:

- **Stovepipe information systems leading to inability to connect the dots.** Independent agencies have separate data systems, funding streams, and chains of command. This separation of data and ownership can obscure relationships and inhibit the ability of law enforcement, justice and public safety, and homeland security officials to have the right information at the right time to assist in proper decision making. By providing these leaders with the technology framework to share information, the nation's capacity to combat crime and terrorism, as well as improve the administration of justice and homeland security, can be greatly improved.
- **Large number of organizations at the Federal, State, Local and Tribal levels including the private sector.** There are a large number of jurisdictions at the public level as well the private level with disparate information systems, governance and activities that need to share information. The sheer number of organizations and their autonomous nature engender inconsistent policies, practices, and systems, thereby making coordination more difficult.
- **Lack of consistent policies and practices.** Information sharing practices and policies often vary from agency to agency with respect to such issues as privacy protection, security, data quality control, and access. These inconsistent approaches combined with lack of advised memoranda of understanding

(MOUs) in place make it difficult—and sometimes illegal—to share information with other agencies.

- **Lack of common standards for the description and definition of data and information.** Without common standards, data is developed and used within information systems in a myriad of different ways, causing data duplication, increasing inaccuracies, and making information usage and alignment across jurisdictions very difficult. In addition, the consistent definition of the sensitivity level or classification of data is often lacking across potential partners, inhibiting confidence in the sharing of secure and protected information.
- **Interagency mistrust.** As a result of inconsistent policies and practices, those who do share sensitive information cannot always be sure how it will be used, whether it will be protected, how it will be disseminated to a third party, and who will ultimately have access to it.
- **Categorization of otherwise shareable information into non-shareable categories.** Another barrier to information sharing is created when information that should be categorized as shareable is categorized in a way that prevents it from being shared. This is primarily due to the lack of department-wide training and awareness strategy with regard to information handling.
- **Privacy with regard to information sharing.** Ethical and legal obligations compel every professional in the justice system to protect privacy interests when sharing justice information. Today, increased security needs not only dictate enhanced justice information sharing but also highlight the need to balance privacy protection and justice information access. The ease of digital access now makes analysis of privacy obligations a more complex process. Nonetheless, the underlying foundations for privacy policy exist in our current laws and customs. Constitutions, statutes, regulations, policies, procedures, and common law requirements still control justice entity collection and sharing of information. What is new is the need for justice practitioners to articulate the rules that control their information gathering and sharing activities in a manner that both supports information sharing and protects constitutional privacy rights.
- **Lack of coordination on information sharing efforts.** In many cases, regional information sharing initiatives have not been coordinated with one another or with their federal partners and vice versa. Since the terrorist attacks of September 11, 2001, the President and Congress have sought to address these challenges by mandating information sharing through various Executive Orders and by directing agencies to increase cooperation and sharing, especially as it relates to critical information that affects the security of the homeland.

3.2 Information Sharing Architectures

The Information sharing architectures that have been developed provide the framework for coordinating business processes, information exchanges, technology components, and performance metrics in relation to information sharing. These include the Federal Enterprise Architecture (FEA), the Justice Reference Architecture (JRA) developed by the Global

Infrastructure and Standards Working Group (GISWG), the ISE Enterprise Architecture Framework (EAF) developed by the Program Manager for the Information Sharing Environment (PM-ISE).

These architectures support the sharing of information. NIEM is not a competitor to those activities, but rather complements them as a method used to implement the data exchange layer within these architectures.

4 NIEM Overview

NIEM, as a platform for information sharing, is based on **eXtensible Markup Language** (XML). XML is a structured language for describing information being sent electronically by one entity to another. XML schema defines the rules and constraints for the characteristics of the data, such as structure, relationships, allowable values, and data types.

XML is:

- In-text format, readable by both machines and humans
- license-free
- platform-independent
- well-supported by industry

XML specifications[2] are guided by the W3C standards.

The NIEM data model is represented in XML but provides specialized XML tag names and other structure for data that is constrained to meet the specific information exchange requirements of the justice and homeland security domains. In other words, NIEM utilizes XML to provide a concise and defined vocabulary for sharing critical information throughout the nation. This is true regardless of whether the agency sharing the information is local, state, tribal, or federal and regardless of whether the information is exchanged horizontally or vertically within existing or emerging systems.

NIEM provides a common language with which federal, state, local, and tribal agencies can describe, structure, and share critical information in both emergency and routine situations. NIEM is designed to facilitate information exchange among different domains, such as justice, public safety, emergency and disaster management, intelligence, and homeland security. NIEM makes this possible by providing the data standards and exchange development methods for defining these cross-domain exchanges.

[2] http://www.w3.org/XML/.

4.1 Background

DOJ and DHS launched the NIEM program on February 28, 2005. Among other requirements, NIEM complies with *Homeland Security Presidential Directive-5* (HSPD-5),[3] which assigns the Secretary of Homeland Security the role of principal federal official for domestic incident management. The *Homeland Security Act of 2002*[4] charges the Secretary with the responsibility for coordinating federal operations within the United States to prepare for, respond to, and recover from terrorist attacks, major disasters, and other emergencies. The *Intelligence Reform and Terrorism Prevention Act of 2004* (IRTPA)[5] was signed into law in December 2004, and in 2005, *Executive Order 13388*[6] was issued by the President. These acts and the administrative direction require U.S. government organizations to strengthen the sharing of terrorism information between organizations and appropriate authorities of local and state governments and protect the ability of organizations to acquire this additional information.

4.2 The Evolution of NIEM

In the late 1990s, the state and local criminal justice community began to focus on sharing information rapidly and effectively to serve a variety of public safety needs. The advent of XML provided the technology with which information could be exchanged more efficiently and cost effectively. The **Global Justice XML Data Model (GJXDM)** vocabulary was derived from user requirements and was driven from the "bottom up" by active practitioners in the justice and public safety fields. The unique development approach taken with GJXDM provided an opportunity for national organizations to assist and support the process of sharing critical justice information where that information originates—at the state, local, and tribal levels.

GJXDM demonstrated the value of information sharing and helped promote the business case for NIEM, which now extends that concept on a national level. NIEM includes not only the Justice (JXDM) domain but also represents others, such as intelligence, emergency management, immigration, infrastructure protection, international trade, and screening. NIEM actively encourages federal agency participation while continuing to support state and local requirements and interoperability standards. NIEM provides component-based resources that are reusable and portable to any organization or platform.

Today, the stated objectives of the NIEM PMO are to:

- Bring stakeholders together to identify information sharing requirements for operational and emergency situations.
- Maintain a National Data Model and Reference Vocabulary containing common and domain-specific data components that pertain to agency information needs to facilitate development of discrete information exchanges.

[3] http://www.fas.org/irp/offdocs/nspd/hspd-5.html.

[4] http://www.dhs.gov/xlibrary/assets/hr_5005_enr.pdf.

[5] http://travel.state.gov/pdf/irtpa2004.pdf.

[6] http://www.fas.org/irp/offdocs/eo/eo-13388.htm.

- Develop standards, a common vocabulary, and an online repository of exchange standards to support information sharing.

Developing and implementing NIEM-based exchanges allows agencies to leverage existing investments in information systems by building the bridges to connect them. NIEM standards enable different information systems to share and exchange information, irrespective of the particular technologies in use in those information systems. Moreover, creating and adopting NIEM standards means that local, state, tribal, and federal organizations can reap significant cost benefits through adoption and reuse, rather than building proprietary, single-use software from scratch. The fact that NIEM requirements are driven from the user community rather than a Federal mandate paves the way for faster adoption, and more closely aligned outcomes between the NIEM PMO and its constituents.

4.3 NIEM Data Model

The NIEM data model provides the reference vocabulary for consistent and reusable intra- and interdomain information exchanges. The structure and meaning of NIEM data are defined by the model and dictionary and are represented as XML schema, thereby providing a common framework for information exchange. As part of the NIEM 2.0 release, the model can also be viewed as a spreadsheet[7] or in a database format.

The fundamental building block of NIEM is a data component. Data components are the basic business data items that describe common concepts used in general business activities.

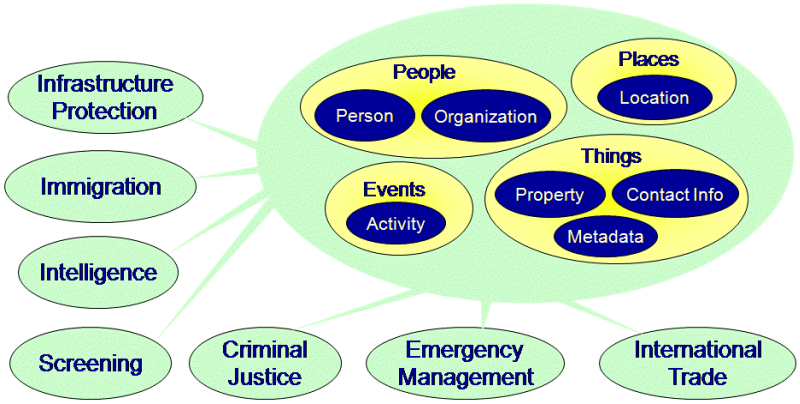


Figure 1: NIEM at 50,000 Feet.

Figure 1 illustrates that NIEM is modeled to be able to describe people, places, things, and events and the relationships between all of them at different points in time.

By far, activity makes up the bulk of the model, with person information coming in second. While each of these categories represents a stand-alone entity, each is structured such that it can also be associated with other categories.

[7] http://www.niem.gov/topicIndex.php?topic=spreadsheet.

The NIEM architecture consists of two sets of vocabularies—**NIEM Core** and the individual NIEM domains. NIEM Core includes **Universal** (U) and **Common** (C) components. The identities for U and C components in NIEM Core are maintained with metadata. *Universal* data components are concepts that are commonly understood across all business domains, such as dates, times, and locations. They do not have to appear in every exchange and do not have to apply all the time—they simply have to be well-defined and well-known enough to be understood by all (or the majority of) domains. *Common* data components, on the other hand, are used in exchanges between two or more domains but not universally shared.

By contrast, the individual NIEM domains contain domain-specific data components. As illustrated in Figure 2, the domains of Emergency Management, Justice, Infrastructure Protection, Intelligence, International Trade, and Immigration are currently participating in NIEM. Additional domains will be added as policy evolves and operational requirements emerge.

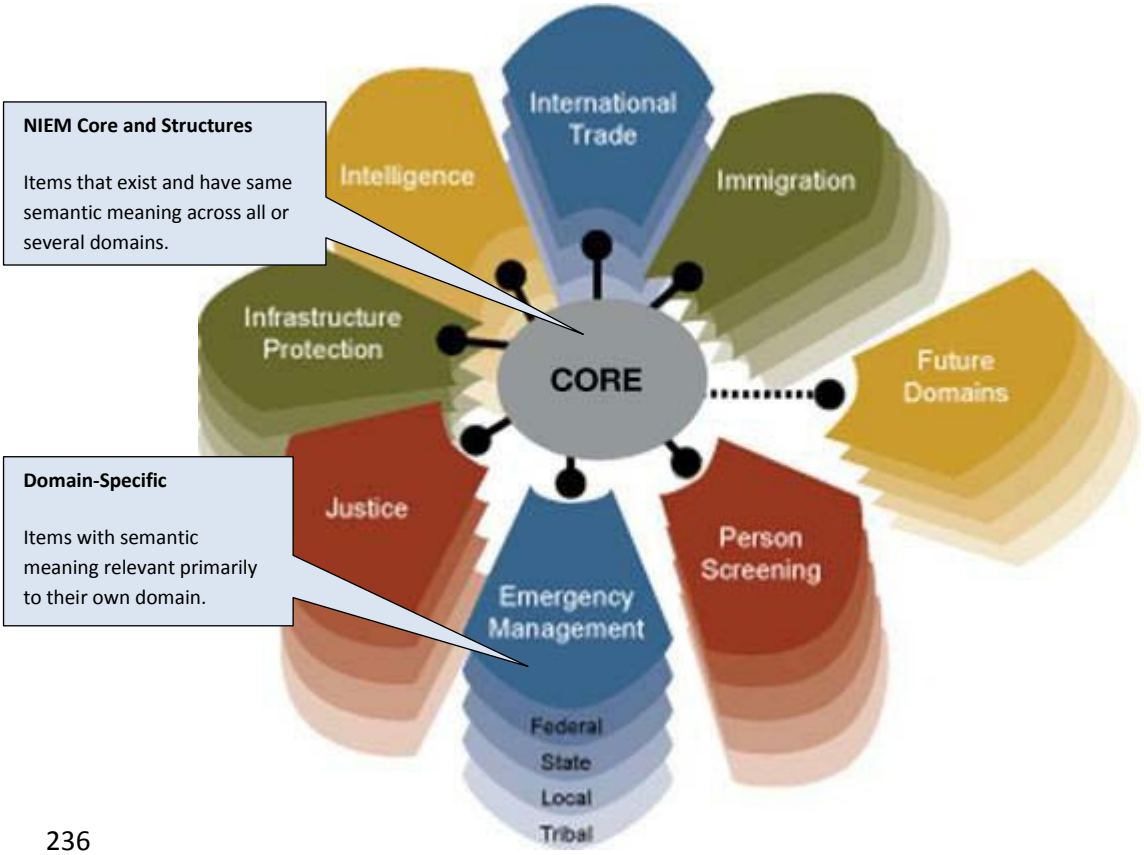


Figure 2: NIEM Core and Domains.

As of version 2.0, NIEM consists of 3,985 data elements and 777 data types. The elements are grouped into namespaces—NIEM Core or one of the seven domains.

These core components are commonly understood and their meanings are agreed to by many, if not all, domains. The standardization of these core components provides significant potential for increased interoperability among and between justice and public safety information systems. Standardization in this manner provides each of us with functionally equivalent or interchangeable components of the system or process in which they are used, regardless of our individual system differences.

The data model and dictionary are combined into one database—a component repository—which allows the consistent generation of several products that can be consumed by the sharing community:

- The NIEM schema
- Numerous external code table schemas
- A NIEM documentation spreadsheet

It is recommended that new users acquaint themselves with the NIEM Component Mapping Tool (CMT) spreadsheet,[8] which is provided as a Microsoft Excel file for easy navigation. The NIEM CMT spreadsheet provides all the element names organized hierarchically under the domains (NIEM Core, Emergency Management, Justice, etc.) with hyperlinks to related elements. The spreadsheet also provides information as to the type of data being represented (date, integer, Boolean, string, etc.) and a precise definition of each dictionary component. The definitions represent a commitment to provide reusable components that mean the same thing to all domains.

4.4 Design Criteria for NIEM

The primary goal for NIEM has been to develop a common set of reusable, extensible XML data components that could be combined in documents, transactions, and messages that are consistently structured to support interoperability between systems. The following design criteria were used in the development of NIEM:

- NIEM should be constructed from actual functional requirements, reference documents, use cases, and business-context components.
- An object-oriented data model, named types, and extensions are best suited to the goals of interagency information exchange.
- The composition of the data dictionary should be over-inclusive and optional to allow users to pick and choose appropriate building blocks for their data exchanges.
- NIEM element and attribute tag names should be based on relevant international standards for electronic data exchange, especially ISO/IEC 11179-5:1995—Specification and Standardization of Data Elements[9], as discussed in the NIEM Naming and Design Rules (NDR). Additional source standards include, but are not limited to:

[8] http://www.niem.gov/topicIndex.php?topic=spreadsheet.

[9] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=1758.

- W3C XML Schema Specification and RDF and RDF Schema Specification.[10]
- The Federal CIO Council Draft Federal XML Schema Developer's Guide.[11]
- UN/CEFACT ebXML Core Components Technical Specification 2.01.[12]
- Dublin Core Metadata for Documents.[13]
- U.S. Department of Defense 5015.02-STD Design Criteria Standard for E-RMS Applications.[14]
- The OASIS XML Common Biometrics Format.[15]
- The ASC X12 Reference Model for XML Design.[16]

♦ NIEM continues to evolve, so the data model must facilitate change and extension as required.

♦ Extension methods should comply with NIEM Naming and Design Rules (NDR)[17] to minimize the impact on prior schema and code investments by practitioners and developers.

♦ NIEM must provide migration paths for evolution to new technologies, such as Resource Description Framework (RDF) and Web Ontology Language (OWL).[18]

NIEM provides a mechanism through which standards for information exchange can be defined with a high degree of granularity.

---

[10] http://www.w3.org/XML/Schema#dev.

[11] http://www.xml.gov/documents/in_progress/developersguide.pdf.

[12] http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf.

[13] http://dublincore.org/documents/.

[14] http://www.dtic.mil/whs/directives/corres/pdf/501502std.pdf.

[15] http://www.oasis-open.org/committees/download.php/3353/oasis-200305-xcbf-specification-1.1.doc.

[16] http://www.x12.org/x12org/xmldesign/X12Reference_Model_For_XML_Design.pdf.

[17] http://niem.gov/topicIndex.php?topic=file-NDR-withoutLineNum.

[18] RDF and OWL are semantic Web standards that provide a framework for asset management, enterprise integration, and the sharing and reuse of data on the Web.

5 NIEM Data Model Concepts

5.1 An Introduction to Modeling Concepts

NIEM is a standardized data model and a reference vocabulary implemented in XML schema. The NIEM data model states exactly and explicitly the meaning of a given concept or relationship. Accordingly, an XML instance that conforms to the NIEM XML schema also has specific meaning. The purpose of NIEM is to provide a standard—but extensible—format for use in the exchange of information between information systems.

NIEM employs several constructs that address common concerns in the design of data models that represent information being exchanged between software systems.

- **Types and Properties:** Representations of the physical and conceptual things being communicated.
- **Container Elements:** Elements whose presence in types represents semantically weak relationships.
- **Content Elements and Reference Elements:** Two semantically equivalent ways to represent the properties of a type.
- **Associations:** Representations of the relationships that a type (e.g., "PersonType") has with other types (e.g., "VehicleType," "ActivityType") that do not create duplicate copies of the type in question ("PersonType").
- **Roles:** Representations of the different roles (e.g., "VictimType," "WitnessType") that a type (e.g., "PersonType") plays in its relationships with other types (e.g., "IncidentType," "CaseType") that do not create multiple, and possibly conflicting, specializations of the type in question ("PersonType").
- **Code Lists:** Generic representations of enumerated code values of a type.
- **Augmentation:** Representation of a reusable bundle of properties (e.g., "PersonAugmentationType" containing properties "DriverLicense," "PersonFootPrint," etc.) for the purpose of augmenting the definition of an existing type (e.g., "PersonType") that does not create multiple, and possibly conflicting, specializations of the type in question ("PersonType").
- **Metadata:** Representation of metadata of types in a flexible and extensible manner.
- **External Adapter Types:** Usage of non-NIEM types in a NIEM-conformant schema.

Each of the above-mentioned constructs comes with a prescribed mechanism to follow when designing NIEM-conformant XML schema types and when using elements of those types in XML instances. This chapter describes and exemplifies these constructs and mechanisms.

5.2 Expressing Object-Oriented Concepts in XML: Types and Properties

The NIEM data model consists of "*types*" (of things) that have "*properties*" and that participate in "*relationships*" with other "*types*" (of things).

A ***type*** is a description of a set of things that share the same properties, relationships, and semantics. For example in NIEM, "PersonType" and "VehicleType" represent persons and vehicles—kinds of things.

A ***property*** is a named characteristic of a type. For example, "PersonBirthDate" is a property of "PersonType." Furthermore, the property is of a specific type itself. For example, "PersonBirthDate" is itself of type "DateType."

A ***relationship*** may be modeled as either a type or a property. For example in NIEM, a relationship between persons and vehicles is represented by the type "PersonVehicleAssociationType."

An ***object*** is an instance of a type and is an abstraction of a specific physical thing or a conceptual thing. Also, in an object, the properties have values. For example, John Smith, a specific person, would be an object of type "PersonType" with the property "PersonBirthDate." Also, for John Smith, the property "PersonBirthDate" may have a value of "1970-01-01."

An object may have a unique ID within an XML instance, but it is not required to have a globally unique identifier. The presence of specific objects in an exchange makes the assertions that:

- Objects exist.
- Objects have properties.
- Objects participate in relationships.

The NIEM data model is explicit, not implicit. If the data says a person's name is John Smith, it is not implying that he does not have other names or that John Smith is his legal name or that he is different from a person known as Bob Jones. The only assertion being made is that one of the names by which this person is known is John Smith.

As shown in Table 2, types, properties, and objects in the NIEM data model have equivalent concepts in **XML Schema** and **Unified Modeling Language** (UML).

NIEM Data Model	XML Schema/XML Instance	UML
Type e.g., "PersonType"	Complex Type or Simple Type e.g., nc:PersonType	Class
Property e.g., "PersonBirthDate" of type "DateType"	Element or Attribute e.g., nc:PersonBirthDate of type nc:DateType	Attribute
Object e.g., "Person"	Element or Attribute e.g., nc:Person	Instance/Object

Table 2: Comparison of Terminology in the NIEM Data Model, XML, and UML.

In XML schema, a type is represented by a Simple Type or a Complex Type. A property is represented by an attribute or an element. An object is represented by an element in an XML instance fragment that conforms to the Simple Type or the Complex Type definition.

Consider the following fragment from the NIEM XML schema. The XML schema type *nc:PersonType* represents the NIEM Data Model type "PersonType." The element *nc:PersonBirthDate* represents the property "PersonBirthDate." Finally, the element *nc:AssessmentPerson* of *nc:PersonType* represents an object of "PersonType."

```
  <xsd:element name="PersonBirthDate" type="nc:DateType" nillable="true"/>

  <xsd:complexType name="PersonType">
   …
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          …
          <xsd:element ref="nc:PersonBirthDate" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

<xsd:element name="AssessmentPerson" type="nc:PersonType" nillable="true"/>
```

Figure 3: XML Schema Fragment Illustrating the Definition of nc:PersonType.

Next, consider the fragment below, which shows an XML instance containing nc:AssessmentPerson, where the element nc:PersonBirthDate has a value of "1970-01-01."

```
<nc:AssessmentPerson>
  …
  <nc:PersonBirthDate>1970-01-01</nc:PersonBirthDate>
  …
</nc:AssessmentPerson>
```

Figure 4: XML Instance Fragment Illustrating the Use of nc:PersonType.

In UML, a NIEM Data Model type can be represented by a UML class, a NIEM Data Model property by a UML attribute, and a NIEM Data Model object by a UML instance. For example, the NIEM Data Model type "PersonType" can be depicted as follows:

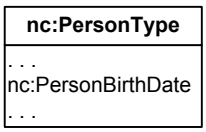


Figure 5: Diagram Depicting nc:PersonType.

In another example, the NIEM data model object “AssessmentPerson” containing the property “PersonBirthDate” with a value of “1970-01-01” could be depicted as in Figure 6.

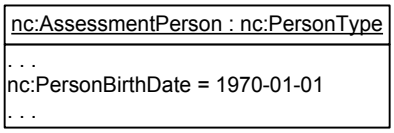


Figure 6: Diagram Depicting nc:AssessmentPerson.

5.3 Container Elements

There are two levels of semantics that can be associated with the presence of an element in a type—weak semantics and strong semantics. Consider for example, j:DriverLicenseDrivingIncidentAssociationType, which represents an association between a driver’s license and a driving incident and contains an element nc:Person of nc:PersonType. The presence of the nc:Person element does not establish what kind of relationship exists between j:DriverLicenseDrivingIncidentAssociationType and nc:PersonType, only that there is a relationship. This is an example of a semantically weak relationship. In such a case, the element nc:Person is called a “container element” because it only serves the purpose of containing an object of nc:PersonType, while leaving the exact meaning unstated.

```
  <xsd:complexType name="DriverLicenseDrivingIncidentAssociationType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/niem-core/2.0"
i:name="AssociationType"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="nc:AssociationType">
        <xsd:sequence>
          <xsd:element ref="nc:Person" minOccurs="0" maxOccurs="unbounded"/>
          <xsd:element ref="nc:DriverLicense" minOccurs="0" maxOccurs="unbounded"/>
          <xsd:element ref="j:DrivingIncident" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

Figure 7: XML Schema Fragment Illustrating j:DriverLicenseDrivingIncidentAssociationType.

If you contrast this situation with that of nc:AssessmentType, which represents an evaluation, appraisal, or assessment of something or someone and contains the element nc:AssessmentPerson of nc:PersonType, it is clear that the person referenced by the element nc:AssessentPerson was responsible for an assessment of some type, relevant to the exchange being modeled. The more descriptive name, nc:AssessmentPerson, makes the relationship between it and nc:AssessmentType a semantically strong relationship.

```
  <xsd:complexType name="AssessmentType">
```

```
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:name="ActivityType"/>
      </xsd:appinfo>
    </xsd:annotation>    <xsd:complexContent>
      <xsd:extension base="nc:ActivityType">
        <xsd:sequence>
          …
          <xsd:element ref="nc:AssessmentPerson" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

Figure 8: XML Schema Fragment Illustrating the Definition of nc:AssessmentType.

Note that the concept of "container element" is only notional. There are no formalized rules about what makes an element a container element. The distinction, however, between container and noncontainer elements is still useful in identifying the meaning that can be explicitly associated with the presence of the element in a type.

One caveat when working with NIEM—When looking for something, do not forget to look upward through all the parent elements for inherited properties.

5.4 Content Elements and Reference Elements

There are two forms in which an element may be present in a type—as a content element or as a reference element. A content element occurs in the definition of its containing type. For example, nc:PersonFullName element occurs as a content element in its containing element nc:PersonNameType in the following XML schema fragment.

```
<!-- targetNamespace="http://niem.gov/niem/niem-core/2.0" →

  <xsd:element name="PersonFullName" type="nc:PersonNameTextType" nillable="true"/>

  <xsd:complexType name="PersonNameType">
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
        …
          <xsd:element ref="nc:PersonFullName" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
        …
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="PersonName" type="nc:PersonNameType" nillable="true"/>
```

Figure 9: Use of nc:PersonFullName as a Content Element in PersonNameType.

The value (“John Smith”) of a content element (nc:PersonFullName) also occurs in-line in its containing element (nc:PersonName) in an XML instance. The following XML instance fragment shows this.

```
  <nc:PersonName>
    …
    <nc:PersonFullName>John Smith</nc:PersonFullName>
  </nc:PersonName>
```

Figure 10: XML Instance Showing the Use of Content Element.

A reference element, on the other hand, is an element that is defined to be of the type s:ReferenceType. For example, nc:PersonFullNameReference element occurs as a reference element in its containing type ext:AlternativePersonNameType in the following XML Schema fragment.

```
<!-- targetNamespace=”http://cjis.gov/extension/1.0” →

  <xsd:element name="PersonFullNameReference" type="s:ReferenceType"/>

  <xsd:complexType name="AlternativePersonNameType">
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          …
          <xsd:element ref="nc:PersonFullNameReference " minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
        …
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="PersonName" type="ext:AlternativePersonNameType" nillable="true"/>
```

Figure 11: Use of Reference Element nc:PersonFullNameReference.

The value (“N1”) of the reference element (nc:PersonFullNameReference) in an XML instance identifies the ID (s:id=”N1”) of the element (nc:PersonFullName) that contains the desired value (“John Smith”). The following XML instance fragment shows this.

```
  <nc:PersonFullName s:id=”N1”>John Smith</nc:PersonFullName>

  <ext:PersonName>
    …
    <ext:PersonFullNameReference s:ref=”N1”/>
  </ext:PersonName>
```

Figure 12: XML Instance Showing Use of a Reference Element.

In the NIEM data model, content elements and reference elements are semantically equivalent.

5.5 Associations

An association is a kind of relationship between two or more objects. The objects and the relationship must have the following characteristics for the relationship to be an association:

- The objects must be "peers" of one another. This means that no object is logically a part of another. In other words, each object can exist independently of others, and none of the objects lose meaning if separated from one another. This also means that each object has its own set of properties, which are independent of the properties of the other objects.
- The relationship between the objects may exist only if all the participating objects exist and it has its own set of properties separate from the properties of the participating objects.

For example in NIEM, a single or a set of related actions, events, or process steps is represented by an "ActivityType" and a person is represented by a "PersonType." Further, the relationship between an activity and a person, signifying the involvement of the person in the activity, is represented by "ActivityPersonAssociationType."

This can be depicted as shown in Figure 13.

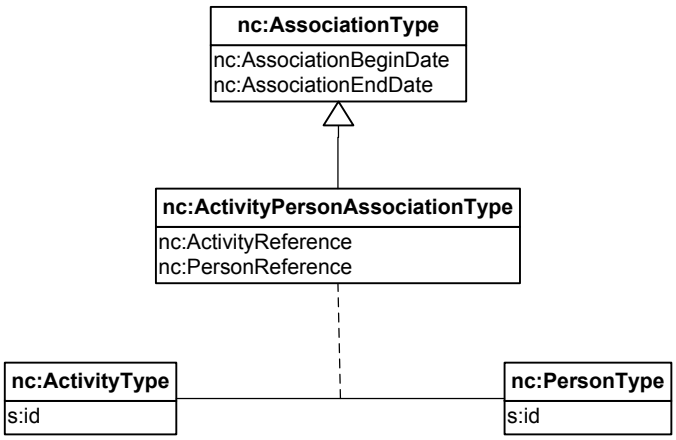


Figure 13: Diagram Illustrating the Definition of nc:ActivityPersonAssociationType.

The NIEM XML Schema represents an association as a type that extends nc:AssociationType. For example, the relationship "ActivityPersonAssociationType" is represented by nc:ActivityPersonAssociationType, which extends nc:AssociationType.

To demonstrate the definition and use of nc:ActivtyPersonAssociationType, you can examine the NIEM XML Schema fragments below, which include:

- The common components from the Structures namespace
- The "Activity"-related components from the NIEM Core namespace

- The “Person”-related components from the NIEM Core namespace
- The “ActivityPersonAssociation”-related components from the NIEM Core namespace
- The “IncidentInvestigatorAssociation” element from the Justice namespace

Following these is a fragment of an XML instance containing the j:IncidentInvestigatorAssociation element.

The first XML schema fragment shows the common components that are used directly or indirectly in the definition of nc:ActivityPersonAssociationType. Most complex types in NIEM are based on the abstract type s:ComplexObjectType, which contains three attributes.

1. The first attribute, ***s:id***, enables an element to identify itself uniquely within an XML instance.
2. The second attribute, ***s:metadata***, enables an element to point to metadata that affects itself.
3. The third attribute, ***s:linkMetadata***, enables an element to point to metadata that affects the relationship between itself and its context.

All reference elements within NIEM-conformant schemas are of the type s:ReferenceType. The s:ref attribute of s:ReferenceType enables an element of s:ReferenceType to point to another element of a different type.

```
<!-- Subset schema (Structures namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/structures/2.0"
…>
  …
  <xsd:attribute name="id" type="xsd:ID"/>
  <xsd:attribute name="ref" type="xsd:IDREF"/>

  <xsd:complexType name="ComplexObjectType" abstract="true">
    <xsd:attribute ref="s:id"/>
    <xsd:attribute ref="s:metadata"/>
    <xsd:attribute ref="s:linkMetadata"/>
  </xsd:complexType>

  <xsd:complexType name="ReferenceType" final="#all">
    <xsd:attribute ref="s:id"/>
    <xsd:attribute ref="s:ref"/>
    <xsd:attribute ref="s:linkMetadata"/>
  </xsd:complexType>
  …
</xsd:schema>
```

Figure 14: XML Schema Fragment Illustrating the Definition of s:ComplexObjectType and s:ReferenceType.

The next XML schema fragment shows the definition of the type nc:ActivityType and the element nc:Activity. It also contains the definition of the element nc:ActivityReference, which

will be used later by the type nc:ActivityPersonAssociationType to refer to the element nc:Activity.

```
<!-- Subset schema (NIEM Core namespace)  → 
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/niem-core/2.0"
…>
  …
  <xsd:complexType name="ActivityType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          …
          <xsd:element ref="nc:ActivityName" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="Activity" type="nc:ActivityType" nillable="true"/>

  <xsd:element name="ActivityReference" type="s:ReferenceType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:ReferenceTarget i:name="ActivityType"/>v
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>
  …
</xsd:schema>
```

Figure 15: XML Schema Fragment Illustrating nc:ActivityType and the Element nc:ActivityReference.

The following XML schema fragment shows the definitions of the type nc:PersonType and the element nc:Person of that type. It also contains the definition of the element nc:PersonReference of s:ReferenceType. This element will be used later by nc:ActivityPersonAssociationType to refer to the nc:Person element.

```
<!-- Subset schema (NIEM Core namespace)  → 
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/niem-core/2.0"
…>
  …
  <xsd:complexType name="PersonType">
```

```
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          …
          <xsd:element ref="nc:PersonName" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="Person" type="nc:PersonType" nillable="true"/>
  <xsd:element name="PersonReference" type="s:ReferenceType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:ReferenceTarget i:name="PersonType"/>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>
  …
</xsd:schema>
```

Figure 16: XML Schema Fragment Illustrating nc:PersonType and the Element nc:PersonReference.

The next XML schema fragment shows the definitions of nc:AssociationType and nc:ActivityPersonAssociationType. The type nc:ActivityPersonAssociationType has properties—nc:AssociationBeginDate and nc:AssociationEndDate inherited from nc:AssociationType—independently of the participating objects activity and person. In addition to these two properties, the association also has references—nc:ActivityReference and nc:PersonReference—to the activity and person objects participating in the relationship.

```
<!-- Subset schema (NIEM Core namespace)  →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/niem-core/2.0"
…>
  …
  <xsd:complexType name="AssociationType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Association"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          <xsd:element ref="nc:AssociationBeginDate" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:element ref="nc:AssociationEndDate" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
```

```
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:complexType name="ActivityPersonAssociationType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:name="AssociationType"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="nc:AssociationType">
        <xsd:sequence>
          <xsd:element ref="nc:ActivityReference" minOccurs="0" maxOccurs="unbounded"/>
          <xsd:element ref="nc:PersonReference" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  …
</xsd:schema>
```

Figure 17: XML Schema Fragment Illustrating nc:AssociationType and nc:ActivityPersonAssociationType.

The following fragment from the Justice domain namespace shows the definition of the element j:IncidentInvestigatorAssociation of type nc:ActivityPersonAssociationType.

```
<!– Extension schema →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
xmlns:j="http://niem.gov/niem/domains/jxdm/4.0"
targetNamespace="http://niem.gov/niem/domains/jxdm/4.0"
…>
  …
  <xsd:element name="IncidentInvestigatorAssociation"
type="nc:ActivityPersonAssociationType"
  nillable="true"/>  …
</xsd:schema>
```

Figure 18: XML Schema Fragment Illustrating j:IncidentInvestigatorAssociation.

Finally, the following fragment shows an XML instance that conforms to the definition of j:IncidentInvestigatorAssociation element.

```
<nc:Activity id="A1" >Some activity name</nc:Activity>

<nc:Activity id="A2" >Some other activity name</nc:Activity>

<nc:Person id="P1" >
  <nc:PersonName>
    <nc:PersonGivenName>John</nc:PersonGivenName>
    <nc:PersonSurName>Smith</nc:PersonSurName>
  </nc:PersonName>
</nc:Person>
```

```
<nc:Person id="P2" >
  <nc:PersonName>
    <nc:PersonGivenName>Jane</nc:PersonGivenName>
    <nc:PersonSurName>Doe</nc:PersonSurName>
  </nc:PersonName>
</nc:Person>

<j:IncidentInvestigatorAssociation>
  <nc:AssociationBeginDate>2007-12-28</nc:AssociationBeginDate>
  <nc:ActivityReference s:ref="A1"/>
  <nc:PersonReference s:ref="P1"/>
</j:IncidentInvestigatorAssociation>

<j:IncidentInvestigatorAssociation>
  <nc:AssociationBeginDate>2007-12-29</nc:AssociationBeginDate>
  <nc:ActivityReference s:ref="A1"/>
  <nc:PersonReference s:ref="P2"/>
</j:IncidentInvestigatorAssociation>

<j:IncidentInvestigatorAssociation>
  <nc:AssociationBeginDate>2007-12-30</nc:AssociationBeginDate>
  <nc:ActivityReference s:ref="A2"/>
  <nc:PersonReference s:ref="P2"/>
</j:IncidentInvestigatorAssociation>
```

Figure 19: XML Instance Fragment Illustrating the Use of j:IncidentInvestigatorAssociation.

Since the association is represented by a NIEM type, it (the association) may itself participate in another association with an object.

The nc:AssociationType and types derived directly or indirectly from it are collectively and generally referred to as association types.

Note that when associations are used, XML schema validation cannot guarantee a "valid" XML instance. This is because XML schema can neither ensure that an object reference is valid nor that it is the correct type. For example, the following XML instance is valid even though it contains two errors:

The nc:ActivityReference element in the j:IncidentInvestigatorAssociation element points to a nonexistent nc:Activity element.

The nc:PersonReference element points to an nc:Activity element instead of an nc:Person element.

XML schema validation can only ensure that s:ref attributes of nc:ActivityReference and nc:PersonReference elements contain valid XML ID values, not that they are the correct XML ID values.

```
<nc:Activity id="A1" >Some activity name</nc:Activity>

<nc:Person id="P1" >
  <nc:PersonName>
    <nc:PersonGivenName>John</nc:PersonGivenName>
    <nc:PersonSurName>Smith</nc:PersonSurName>
  </nc:PersonName>
</nc:Person>

<j:IncidentInvestigatorAssociation>
  <nc:AssociationBeginDate>2007-12-28</nc:AssociationBeginDate>
  <nc:ActivityReference s:ref="A2"/> <!-- Valid but incorrect: Referencing a non-existent
ID!! →
```

```
  <nc:PersonReference s:ref="A1"/>  <!—Valid but incorrect: Referencing the ID of a wrong type of element !! →
</j:IncidentInvestigatorAssociation>
```

Figure 20: Example of Valid XML Instance Containing Errors.

5.6 Roles

A role is a particular function, purpose, or use of an object. It may be specific to time, incident, employment, or other aspects of an activity or context. The object to which the role applies is called the "base object."

If the base object is referenced only by the role in the NIEM data model, and there are no additional properties of the role to be modeled, the simplest way to represent the role is to use an element. The following example represents the role of a person who performs an assessment.

```
<xsd:element name="AssessmentPerson" type="nc:PersonType" nillable="true"/>
```

Figure 21: Example of Person Role.

In many cases, however, there is a need to capture additional information about the role. In such cases, a new type is created to represent the role and its properties. For example in NIEM, a person whose whereabouts are unknown is modeled as j:MissingPersonType, which represents a particular role of nc:PersonType. Additional information about the person specific to his/her role as a missing person is modeled as the properties of j:MissingPersonType. Such information may include the date on which and the location at which the person was last seen, represented as the properties j:MissingPersonLastSeenDate and j:MissingPersonLastSeenLocation.

Figure 22 illustrates this.

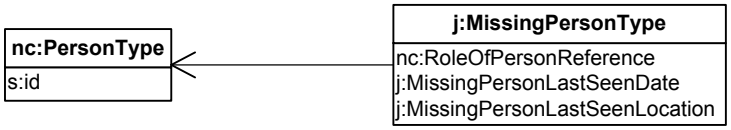


Figure 22: j:MissingPersonType as a Role of nc:PersonType.

To demonstrate the definition and use of j:MissingPersonType, review the NIEM XML Schema fragments below, which include:

- The common components from the Structures namespace
- The "Person"-related components from the NIEM Core namespace
- The "MissingPerson"-related components from the Justice domain namespace

Following these is a fragment of an XML instance containing the j:MissingPerson element.

The first XML schema fragment shows the common components that are used directly or indirectly in the definition of j:MissingPersonType.

```
<!-- Subset schema (Structures namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/structures/2.0"
…>
  …
  <xsd:attribute name="id" type="xsd:ID"/>
  <xsd:attribute name="ref" type="xsd:IDREF"/>

  <xsd:complexType name="ComplexObjectType" abstract="true">
    <xsd:attribute ref="s:id"/>
    <xsd:attribute ref="s:metadata"/>
    <xsd:attribute ref="s:linkMetadata"/>
  </xsd:complexType>

  <xsd:complexType name="ReferenceType" final="#all">
    <xsd:attribute ref="s:id"/>
    <xsd:attribute ref="s:ref"/>
    <xsd:attribute ref="s:linkMetadata"/>
  </xsd:complexType>
  …
</xsd:schema>
```

Figure 23: XML Schema Fragment Illustrating j:MissingPersonType.

The next XML schema fragment shows the definitions of the type nc:PersonType and the element nc:Person. It also contains the definition of the element nc:RoleOfPersonReference, which will be used later by the type j:MissingPersonType to refer to the element nc:Person.

```
<!-- Subset schema (NIEM Core namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/niem-core/2.0"
…>
  …
  <xsd:complexType name="PersonType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          …
          <xsd:element ref="nc:PersonName" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="Person" type="nc:PersonType" nillable="true"/>

  <xsd:element name="RoleOf" abstract="true"/>
```

```
  <xsd:element substitutionGroup="nc:RoleOf" name="RoleOfPersonReference"
type="s:ReferenceType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:ReferenceTarget i:name="PersonType"/>
        <i:Base i:name="RoleOf"/>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>
  …
</xsd:schema>
```

Figure 24: XML Schema Fragment Illustrating j:MissingPersonType.

The following fragment from the Justice domain namespace shows the definition of the element j:MissingPerson of type j:MissingPersonType.

```
<!-- Subset schema (Justice Domain namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:j="http://niem.gov/niem/domains/jxdm/4.0"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/domains/jxdm/4.0"
…>
  …
  <xsd:complexType name="MissingPersonType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          <xsd:element ref="nc:RoleOfPersonReference" minOccurs="0"
maxOccurs="unbounded"/>
          …
          <xsd:element ref="j:MissingPersonLastSeenDate" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:element ref="j:MissingPersonLastSeenLocation" minOccurs="0"
maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="MissingPerson" type="j:MissingPersonType" nillable="true"/>
  …
</xsd:schema>
```

Figure 25: XML Schema Fragment Illustrating j:MissingPersonType.

Finally, the following fragment shows an XML instance that conforms to the definition of j:MissingPerson element.

```
<nc:Person id="P1" >
  <nc:PersonName>
    <nc:PersonGivenName>John</nc:PersonGivenName>
    <nc:PersonSurName>Smith</nc:PersonSurName>
  </nc:PersonName>
</nc:Person>

<j:MissingPerson>
  <nc:RoleOfPersonReference ref="P1"/>
  <j:MissingPersonLastSeenDate>2007-12-31</j:MissingPersonLastSeenDate>
  …
</j:MissingPerson>
```

Figure 26: XML Instance Fragment Illustrating the Use of j:MissingPersonType.

As with associations, it should be noted that when roles are used, XML schema validation cannot guarantee a "valid" XML instance. This is because XML schema can neither ensure that an object reference is valid nor that it is the correct type. For example, the following XML instance is valid even though it contains an error, specifically, the nc:RoleOfPersonReference element in j:MissingPerson element points to a nonexistent nc:Person element. XML schema validation can only ensure that s:ref attributes of the nc:RoleOfPersonReference element contains a valid XML ID value, not that it is the correct XML ID value.

```
<nc:Person id="P1" >
  <nc:PersonName>
    <nc:PersonGivenName>John</nc:PersonGivenName>
    <nc:PersonSurName>Smith</nc:PersonSurName>
  </nc:PersonName>
</nc:Person>

<j:MissingPerson>
  <nc:RoleOfPersonReference ref="A2"/>
  <j:MissingPersonLastSeenDate>2007-12-31</j:MissingPersonLastSeenDate>
  …
</j:MissingPerson>
```

Figure 27: Valid XML Instance Fragment With Error.

5.7 Abstract Elements and Substitution Groups

Substitution groups, abstract elements, and element substitution are XML schema features that can be used as means to create an XML schema type that is extensible by an XML instance author.

A substitution group consists of two parts—a single group head element and one or more group member elements. In an XML instance, the group head element can be replaced by one of the group member elements. If the group head element has a type, then the group member elements must be of the same type or of a type derived from that same type.

When an element is declared to be abstract in XML schema, it cannot be used in an XML instance. Instead, a member of that element's substitution group must appear in the XML instance.

NIEM relies on the XML schema feature of element substitution for representing different kinds of enumerated code values for a type and for using those code values in an XML instance.

For example, consider the different kinds of representations and uses of code values denoting states or provinces such as Alberta or Minnesota. The NIEM XML Schema identifies a province through a value of nc:CanadianProvinceCodeType and a state through a value of nc:USStateCodeType. NIEM also defines the abstract type-less element nc:LocationState to represent the concept of a province or a state. Because it is abstract, the nc:LocationState element cannot appear in an XML instance. Rather, it heads a substitution group, which contains elements of eight different types, including the elements nc:LocationStateCanadianProvinceCode of the type nc:CanadianProvinceCodeType, and nc:LocationStateFIPS5-2AlphaCode of the type nc:USStateCodeType. In the XML instance, the nc:LocationState element can be replaced by the nc:LocationStateCanadianProvinceCode element or the nc:LocationStateFIPS5-2AlphaCode element. Finally, NIEM also defines the element nc:Jurisdiction of the type nc:JurisdictionType. The type nc:JurisdictionType contains the element nc:LocationState and represents a geopolitical area in which an organization, a person, or an object has a specific range of authority.

Figure 28 (below) illustrates this example.

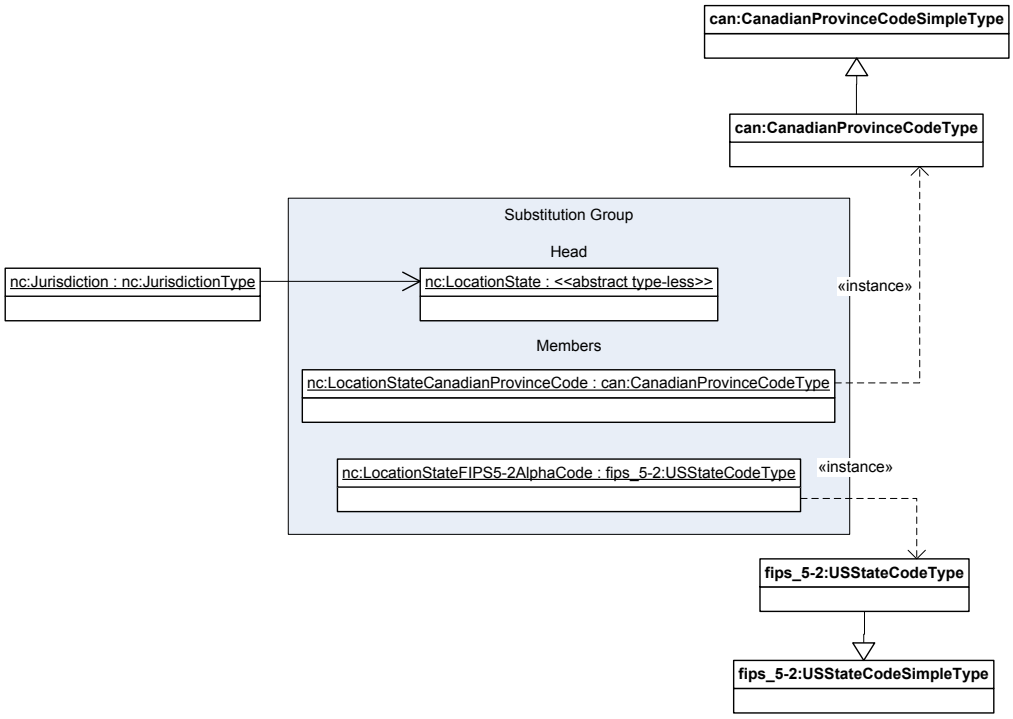


Figure 28: Abstract Type-Less nc:LocationState Element.

To demonstrate the definition and use of nc:CanadianProvinceCodeType and nc:USStateCodeType, consider the NIEM XML Schema fragments below, which include:

- The common components from the Structures namespace
- The "Jurisdiction"-related components from the NIEM Core namespace
- The "Province"-related components from the Post Canada namespace
- The "State"-related components from the FIPS 5.2 namespace

Following these are two fragments of XML instances containing the nc:Jurisdiction element.

The first XML schema fragment shows the common components that are used directly or indirectly in the definition of nc:CanadianProvinceCodeType and nc:USStateCodeType. These code types contain the attribute group s:SimpleObjectAttributeGroup, which, in turn, contains the attribute s:id. The attribute s:id enables an element to identify itself uniquely within an XML instance.

```
<!-- Subset schema (Structures namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/structures/2.0"
…>
  …
  <xsd:attribute name="id" type="xsd:ID"/>
  <xsd:attribute name="linkMetadata" type="xsd:IDREFS"/>
  <xsd:attribute name="metadata" type="xsd:IDREFS"/>

  <xsd:attributeGroup name="SimpleObjectAttributeGroup">
    <xsd:attribute ref="s:id"/>
    <xsd:attribute ref="s:metadata"/>
    <xsd:attribute ref="s:linkMetadata"/>
  </xsd:attributeGroup>  …
</xsd:schema>
```

Figure 29: XML Schema Fragment Illustrating s:SimpleObjectAttributeGroup.

The next XML schema fragment shows the definition of the abstract type-less element nc:LocatonState. Because it is abstract, nc:LocationState element cannot appear in an XML instance. Rather, it heads a substitution group, which contains the elements nc:LocationStateCanadianProvinceCode and nc:LocationStateFIPS5-2AlphaCode. The fragment also shows the definitions of the type nc:JurisdictionType and the element nc:Jurisdiction. The type nc:JurisdictionType contains the element nc:LocationState.

```
<!-- Subset schema (NIEM Core namespace)  →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/niem-core/2.0"
…>
  …
  <xsd:element name="LocationState" abstract="true"/>
  <xsd:element substitutionGroup="nc:LocationState"
name="LocationStateCanadianProvinceCode"
  type="can:CanadianProvinceCodeType" nillable="true"/>
  <xsd:element substitutionGroup="nc:LocationState" name="LocationStateFIPS5-2AlphaCode"
  type="fips_5-2:USStateCodeType" nillable="true"/>
  …
  <xsd:complexType name="JurisdictionType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
```

```
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          <xsd:element ref="nc:LocationCityName" minOccurs="0" maxOccurs="unbounded"/>
          …
          <xsd:element ref="nc:LocationState" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="Jurisdiction" type="nc:JurisdictionType" nillable="true"/>
  …
</xsd:schema>
```

Figure 30: XML Schema Fragment Illustrating nc:LocationState.

The following fragment from the Post Canada namespace shows the definition of the type can:CanadianProvinceCodeType.

```
<!-- Subset schema (NIEM Post Canada namespace)  →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:can="http://niem.gov/niem/post-canada/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/post-canada/2.0"
…>
  …
  <xsd:simpleType name="CanadianProvinceCodeSimpleType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:enumeration value="AB"/>
      …
      <xsd:enumeration value="YT"/>
    </xsd:restriction>
  </xsd:simpleType>
```

```
  <xsd:complexType name="CanadianProvinceCodeType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:simpleContent>
      <xsd:extension base="can:CanadianProvinceCodeSimpleType">
        <xsd:attributeGroup ref="s:SimpleObjectAttributeGroup"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
  …
</xsd:schema>
```

Figure 31: XML Schema Fragment Illustrating can:CanadianProvinceCodeType.

The next fragment from the FIPS 5.2 namespace shows the definition of the type fips_5-2:USStateCodeType.

```
<!-- Subset schema (NIEM FIPS 5.2 namespace)  →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:fips_5-2="http://niem.gov/niem/fips_5-2/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/fips_5-2/2.0"
…>
  …
  <xsd:simpleType name="USStateCodeSimpleType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:restriction base="xsd:token">
      <xsd:enumeration value="AK">
      …
      <xsd:enumeration value="WY">
    </xsd:restriction>
  </xsd:simpleType>

  <xsd:complexType name="USStateCodeType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:simpleContent>
      <xsd:extension base="fips_5-2:USStateCodeSimpleType">
        <xsd:attributeGroup ref="s:SimpleObjectAttributeGroup"/>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
  …
</xsd:schema>
```

Figure 32: XML Schema Fragment Illustrating fips_5-2:USStateCodeType.

Finally, the following fragments show XML instances that conform to the definition of the nc:Jurisdiction element. In the first XML instance, the nc:LocationState element has been replaced by the nc:LocationStateCanadianProvinceCode element—a member of the substitution group headed by nc:LocationState.

```
<nc:Jurisdiction>
  …
  <nc:LocationStateCanadianProvinceCode >AB</nc:LocationStateCanadianProvinceCode >
  …
</nc:Jurisdiction>
```

Figure 33: XML Instance Fragment Illustrating the Use of nc:LocationStateCanadianProvinceCode.

In the second XML instance, the nc:LocationState element has been replaced by the nc:LocationStateFIPS5-2AlphaCode element—also a member of the substitution group headed by nc:LocationState.

```
<nc:Jurisdiction>
  …
  <nc:LocationStateFIPS5-2AlphaCode>MN</nc:LocationStateFIPS5-2AlphaCode>
  …
</nc:Jurisdiction>
```

Figure 34: XML Instance Fragment Illustrating the Use of nc:LocationStateFIPS5-2AlphaCode.

5.8 Augmentation

Augmentation is a mechanism prescribed by NIEM to create a new NIEM-derived type, via the extension of an existing NIEM type, by adding a block of elements bundled together in another type called an "augmentation type." The practice of bundling the additional elements in an augmentation type is preferable to directly placing the elements in the new NIEM-derived type because it provides a reusable bundle of properties. The mechanism also prescribes that the augmentation type also extend the s:AugmentationType and that the element of the new augmentation type be made a member of the substitution group headed by the element s:Augmentation.

To illustrate the mechanism of augmentation, consider the following example. Suppose we wish to use an element of nc:PersonType in our schema, but that we also need to capture additional properties, such as the person's driver's license and place of birth, which nc:PersonType does not provide. NIEM provides several person-related properties in j:PersonAugmentationType, including nc:DriverLicense and j:PersonBirthPlaceCode, which match the two additional properties we want. Therefore, we create a new NIEM-derived type, ext:PersonType, which extends nc:PersonType by adding an element of j:PersonAugmentationType to nc:PersonType.

Figure 35 illustrates this.

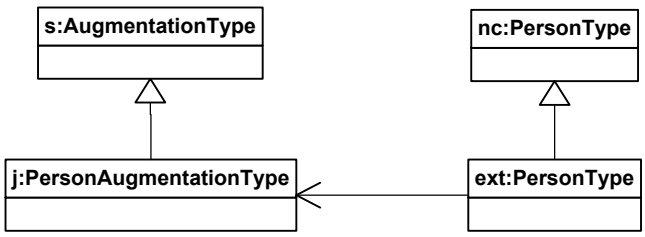


Figure 35: Use of j:PersonAugmentationType.

To illustrate the definition and use of j:PersonAugmentationType, consider the NIEM XML Schema fragments below, which include:

- The common components from the Structures namespace.
- The relevant components from the NIEM Core namespace.
- The relevant components from the Justice namespace.

- The relevant components from the Local Extension namespace.

Figure 36 is an XML instance fragment containing the ext:Person element.

We first show the relevant XML schema components from the NIEM Structures namespace that are used directly or indirectly in the definition of s:AugmentationType.

```
<!-- Subset schema (Structures namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/structures/2.0"
…>
  …
  <xsd:attribute name="id" type="xsd:ID"/>
  <xsd:attribute name="ref" type="xsd:IDREF"/>

  <xsd:complexType name="ComplexObjectType" abstract="true">
    <xsd:attribute ref="s:id"/>
    <xsd:attribute ref="s:metadata"/>
    <xsd:attribute ref="s:linkMetadata"/>
  </xsd:complexType>

  <xsd:complexType name="AugmentationType" abstract="true">
    <xsd:attribute ref="s:id"/>
    <xsd:attribute ref="s:metadata"/>
  </xsd:complexType>

  <xsd:element name="Augmentation" type="s:AugmentationType" abstract="true"/>
  …
</xsd:schema>
```

Figure 36: XML Schema Fragment Illustrating s:AugmentationType.

The following fragment shows the relevant XML schema components from the NIEM Core namespace.

```
<!-- Subset schema (NIEM Core namespace)  →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/niem-core/2.0"
…>
  …
  <xsd:complexType name="PersonType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          …
          <xsd:element ref="nc:PersonName" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
```

```
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="Person" type="nc:PersonType" nillable="true"/>
  …
</xsd:schema>
```

Figure 37: XML Schema Fragment Illustrating nc:PersonType.

The following fragment shows the relevant XML schema components from the Justice domain namespace.

```
<!-- Subset schema (Justice Domain namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:j="http://niem.gov/niem/domains/jxdm/4.0"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/domains/jxdm/4.0"
…>
  …
  <xsd:complexType name="PersonAugmentationType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0"
i:name="AugmentationType"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:AugmentationType">
        <xsd:sequence>
          <xsd:element ref="nc:DriverLicense" minOccurs="0" maxOccurs="unbounded"/>
          …
          <xsd:element ref="j:PersonBirthPlaceCode" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>  <xsd:element name="PersonAugmentation"
type="j:PersonAugmentationType"
  substitutionGroup="s:Augmentation">
    <xsd:annotation>
      <xsd:appinfo>
        <i:AppliesTo i:namespace="http://niem.gov/niem/niem-core/2.0"
i:name="PersonType"/>
      </xsd:appinfo>
    </xsd:annotation>
  </xsd:element>
  …
</xsd:schema>
```

Figure 38: XML Schema Fragment Illustrating j:PersonAugmentationType.

The following fragment shows the definition of ext:PersonType in the local extension namespace.

```
<!– Extension schema →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:j="http://niem.gov/niem/domains/jxdm/4.0"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
xmlns:ext="http://cjis.gov/extension/1.0"
targetNamespace="http://cjis.gov/extension/1.0"
…>
  …
  <xsd:complexType name="PersonType">
    <xsd:complexContent>
      <xsd:extension base="nc:PersonType">
        <xsd:sequence>
          <xsd:element ref="nc:PersonAugmentation" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="Person" type="ext:PersonType" substitutionGroup="nc:Person">
  </xsd:element>
  …
</xsd:schema>
```

Figure 39: XML Schema Fragment Illustrating ext:PersonType.

Finally, the following fragment shows an XML instance that conforms to the definition of ext:Person element.

```
<ext:Person>
  …
  <nc:PersonName>
    <nc:PersonGivenName>John</nc:PersonGivenName>
    <nc:PersonSurName>Smith</nc:PersonSurName>
  </nc:PersonName>
  …
  <nc:PersonAugmentation>
    <nc:DriverLicense>VA 1234</nc:DriverLicense>
    …
    <j:PersonBirthPlaceCode>VA</j:PersonBirthPlaceCode>

    …
  </nc:PersonAugmentation>
  …
</ext:Person>
```

Figure 40: XML Instance Fragment Illustrating the Use of ext:Person.

5.9 Metadata

Meta is generally used as a prefix to mean "one level of description higher." If *X* is a given concept, then *meta-X* is information about or processes operating on *X*. For example, a meta-syntax is syntax for specifying syntax, meta-language is a language used to discuss language, and meta-reasoning is reasoning about reasoning.

Likewise, **metadata** is data about data. It is information that is not descriptive of objects and their relationships but is descriptive of data itself. For example, NIEM provides j:EvidenceType, which represents an item received by or submitted to an agency for use in ascertaining the truth of a matter. It contains elements such as j:EvidenceAmount, which is an estimated or actual monetary value of a piece of evidence, j:EvidenceCollector, which is a person who collected a particular piece of evidence, and so on. In the case of a specific object of j:EvidenceType, these elements have values. These values constitute what is understood as data of the j:EvidenceType object. However, there is information such as whether or not j:EvidenceType object may be regarded as criminal or intelligence information (as may be the case if j:EvidenceType object was a document of some kind). Such information is considered to be metadata about the j:EvidenceType object.

Note that whether information is considered to be "metadata" or "data" is subjective or relative. It can be difficult to draw a clear dividing line between metadata and data.

NIEM prescribes a specific method for representing metadata. A type that represents metadata is called a metadata type. NIEM defines an abstract s:MetadataType to serve as the base type for all metadata types. The s:MetadataType contains a single attribute, s:id, the value of which uniquely identifies a metadata type element within an exchange. NIEM also defines an abstract element, s:Metadata, to serve as the head element of the substitution group in which all concrete metadata type elements should be placed.

The following XML schema fragment from the NIEM Structures namespace shows these definitions.

```
<!-- Subset schema (Structures namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/structures/2.0"
…>
  …
  <xsd:attribute name="id" type="xsd:ID"/>
  <xsd:attribute name="ref" type="xsd:IDREF"/>
  <xsd:attribute name="linkMetadata" type="xsd:IDREFS"/>
  <xsd:attribute name="metadata" type="xsd:IDREFS"/>

  <xsd:complexType name="MetadataType" abstract="true">
    <xsd:attribute ref="s:id"/>
  </xsd:complexType>

  <xsd:element name="Metadata" type="s:MetadataType" abstract="true"/>
```

```
  <xsd:attributeGroup name="SimpleObjectAttributeGroup">
    <xsd:attribute ref="s:id"/>
    <xsd:attribute ref="s:metadata"/>
    <xsd:attribute ref="s:linkMetadata"/>
  </xsd:attributeGroup>  …
</xsd:schema>
```

Figure 41: XML Schema Fragment Illustrating s:MetadataType.

As shown in the following fragment from the Justice domain namespace, j:JusticeMetadataType extends s:MetadataType and adds two additional elements, j:CriminalInformationIndicator and j:IntelligenceInformationIndicator. The namespace also defines j:EvidenceType, which can use the j:JusticeMetadataType element to capture its metadata information.

```
<!-- Subset schema (Justice Domain namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:j="http://niem.gov/niem/domains/jxdm/4.0"
xmlns:nc="http://niem.gov/niem/niem-core/2.0"
xmlns:s="http://niem.gov/niem/structures/2.0"
xmlns:i="http://niem.gov/niem/appinfo/2.0"
targetNamespace="http://niem.gov/niem/domains/jxdm/4.0"
…>
  …
  <xsd:complexType name="JusticeMetadataType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="MetadataType"/>
        <i:AppliesTo i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
        <i:AppliesTo i:namespace="http://niem.gov/niem/structures/2.0"
i:name="Association"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:MetadataType">
        <xsd:sequence>
          <xsd:element ref="j:CriminalInformationIndicator" minOccurs="0"
maxOccurs="unbounded"/>
          <xsd:element ref="j:IntelligenceInformationIndicator" minOccurs="0"
maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="JusticeMetadata" type="j:JusticeMetadataType" nillable="true"/>

  <xsd:complexType name="EvidenceType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:Base i:namespace="http://niem.gov/niem/structures/2.0" i:name="Object"/>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="s:ComplexObjectType">
        <xsd:sequence>
          <xsd:element ref="j:EvidenceAmount" minOccurs="0" maxOccurs="unbounded"/>
          <xsd:element ref="j:EvidenceCollector" minOccurs="0" maxOccurs="unbounded"/>
          …
```

```
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  <xsd:element name="Evidence" type="j:EvidenceType" nillable="true"/>
  …
</xsd:schema>
```

Figure 42: XML Schema Fragment Illustrating j:JusticeMetadataType.

Finally, the following fragment shows an XML instance containing a j:Evidence element that uses the element j:JusticeMetadata to represent its metadata information.

```
<j:JusticeMetadata s:id="M1">
  <j:CriminalInformationIndicator>true</j:CriminalInformationIndicator>
  <j:IntelligenceInformationIndicator>false</j:IntelligenceInformationIndicator>
</j:JusticeMetadata>

<j:Evidence s:metadata="M1">
  <j:EvidenceAmount>…</j: EvidenceAmount >
  <j:EvidenceCollector>… </j: EvidenceCollector >
  …
</j:Evidence>
```

Figure 43: XML Instance Fragment Illustrating the Use of j:JusticeMetadata

Figure 44 illustrates this example in a class diagram.

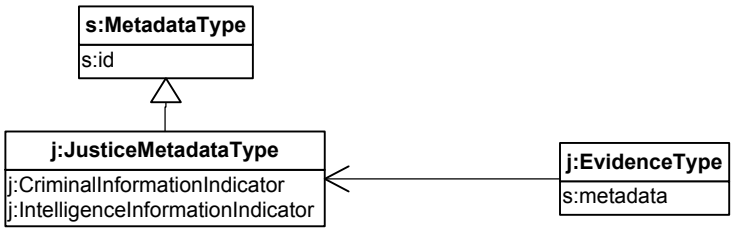


Figure 44: Use of j:JusticeMetadataType.

A metadata type can be defined to apply to specific types only. To do this, you can add an xsd:annotation/xsd:appinfo/i:AppliesTo element to the xsd:complexType definition of the metadata type. For example, j:JusticeMetadataType specifies that it can only be applied to types derived from Object and Association. This requirement is not enforced by XML schema, of course, and is the responsibility of the user's application.

5.10 External Adapter Types

While NIEM is a far-reaching standard, there are other information exchange standards used in other communities. To share information with these other communities, NIEM includes support for external standards. NIEM prescribes that the XML schema types from non-NIEM namespaces should be wrapped in NIEM-conformant types so they may be used in a NIEM-conformant schema. The main construct available in NIEM 2.0 for wrapping non-NIEM-conforming types is the external adapter type.

The external adapter type is a NIEM-conformant type that can contain:

- Attributes from external namespaces.
- Elements from external namespaces.

The subparts of that adapter type should correspond to a semantically meaningful concept. The adapter type may reference content from more than one external namespace, but all content must be from external namespaces.

There are some special importing and packaging requirements for an IEPD that accesses external adapter types. An IEPD that uses an external namespace through adapter components will require the import of both a schema that contains the NIEM-conformant components (adapter types) and the non-NIEM conformant external schemas. All the relevant schemas must be included in the IEPD. Aside from these requirements, however, external adapter types can be used in an IEPD just like standard NIEM types. Nothing special is required for designing schemas or instances that use external adapter types.

In the following examples, the schema import statements are removed for the sake of brevity. The Geospatial standard uses the prefix “geo:,” while the external content itself uses the prefix “addr:.”

First is an example of an external adapter type from the Geospatial external standard in NIEM 2.0 (geo:). The adapter type wrapping the nonconformant elements is geo:SingleSiteLandmarkAddressType.

```
<xsd:complexType name="SingleSiteLandmarkAddressType">
        <xsd:annotation>
              <xsd:appinfo>
                    <i:Base i:namespace="http://niem.gov/niem/structures/2.0"
                          i:name="Object"/>
                    <i:ExternalAdapterTypeIndicator>
                          true
                    </i:ExternalAdapterTypeIndicator>
              </xsd:appinfo>
       </xsd:annotation>
       <xsd:complexContent>
              <xsd:extension base="s:ComplexObjectType">
                    <xsd:sequence>
                          <xsd:element ref="addr:SingleSiteLandmarkAddress"
                                maxOccurs="unbounded"/>
                    </xsd:sequence>
              </xsd:extension>
       </xsd:complexContent>
</xsd:complexType>
```

Figure 45: Use of External Adapter Type geo:SingleSite LandmarkAddressType.

Note that the appinfo information states that this is an external adapter type. Other than this indicator, the XML schema for this type is much like any NIEM-conformant type. This external adapter type wraps the following external content from the URISA Street address namespace (addr:) but is simplified for this example:

```
<xsd:complexType name="PlaceName_type">
       <xsd:choice>
              <xsd:element name="MunicipalJurisdiction"
                    type="addr:MunicipalJurisdiction_type"/>
              <xsd:element name="USPSPlaceName"
                    type="addr:USPSPlaceName_type"/>
       </xsd:choice>
</xsd:complexType>
<xsd:complexType name="SingleSiteLandmarkAddress_type">
       <xsd:sequence>
              <xsd:element name="LandmarkName"
                    type="addr:LandmarkName type"/>
              <xsd:element name="PlaceName" type="addr:PlaceName_type"/>
              <xsd:element name="StateName" type="addr:StateName_type"/>
              <xsd:element name="ZipCode" type="addr:ZipCode_type"/>
       </xsd:sequence>
       <xsd:attribute name="action" type="addr:Action_type" use="optional"/>
</xsd:complexType>

<xsd:element name="SingleSiteLandmarkAddress"
       type="addr:SingleSiteLandmarkAddress_type"/>
```

Figure 46: Definition of External Type addr:MunicipalJurisdiction and addr:USPSPlaceName.

Note that this external content is not NIEM-conformant.

Finally, the resulting instance document pulls the two together, allowing for the use of the external standard elements within a NIEM-conformant exchange:

```
<geo:SingleSiteLandmarkAddress>
       <addr:SingleSiteLandmarkAddress>
              <addr:LandmarkName>Statue of Liberty</addr:LandmarkName>
              <addr:PlaceName>
                     <addr:MunicipalJurisdiction>
                            New York
                     </addr:MunicipalJurisdiction>
              </addr:PlaceName>
              <addr:StateName>NY</addr:StateName>
              <addr:ZipCode>10004</addr:ZipCode>
       </addr:SingleSiteLandmarkAddress>
</geo:SingleSiteLandmarkAddress>
```

Figure 47: XML Instance Showing the Use of an External Adapter Type.

Using external adapter types to wrap non-NIEM-conformant standards is a powerful method to leverage other standards from within NIEM without requiring those external standards to be made NIEM-conformant.

6 NIEM Data Model Content

6.1 Architecture of NIEM Model

NIEM contains many XML data types and properties. To ensure that the information being sent is understood clearly, it is important to avoid creating new data types and properties when similar ones exist within the data model. It is therefore important to develop an understanding of the content of NIEM. It is also important to develop a sense of how to go about navigating through the data model. This chapter presents an overview of NIEM with an eye towards aiding better navigation through the data model.

As mentioned previously, NIEM is a collection of data elements and data types, grouped logically into several kinds of XML schemas. The data elements and types can be classified into several broad categories:

- **Appinfo:** This schema provides support for high-level data model concepts and additional syntax to support the NIEM conceptual model and validation of NIEM-conformant instances.
- **Structures**: These elements enable consistent linking and description of information in NIEM. These elements are also used to connect metadata to objects.
- **NIEM Core components**: Basic NIEM objects—these describe core entities like Activity, Person, Document, etc. In addition, the NIEM Core components also describe more complex entities—such as drugs, vehicles, locations—that are useful across multiple domains.
- **Domains**: These represent specialized information models that represent information in verticals such as emergency management, justice, immigration, etc.
- **Standard Code Lists**: These data elements and types are not really a part of NIEM, but they are extremely useful in ensuring that information is described in a consistent manner.

In addition to the XML objects mentioned above, NIEM also provides mechanism to annotate individual data elements with metadata.

- In some cases, the metadata can be created as an entire XML object and be associated with the appropriate piece of information whose characteristics are being described. This is made possible by the XML object ***s:MetadataType***.
- In other cases, annotating individual elements of data is preferable. In these cases, you should use XML attributes provided by NIEM.
  - For example, when dealing with monetary information, a useful annotation would be the currency that the data represents.
  - Similarly, it might be desirable to explicitly specify the language of a piece of text.

A closer look at the XML Attributes in NIEM is presented towards the end of this section.

6.1.1 Relationships Between the Components

The data elements and types defined in each of the groups are interrelated in a specific manner. For instance, elements in the Structures depend only on basic XML constructs. Elements in NIEM Core depend only on each other and the elements in structures. As such, elements in the Structure groups can be considered to be primordial, while the data elements in the NIEM Core and the various domains represent increased specialization of information. Components in the domains depend on components in NIEM Core, Structures, and, in some cases, on each other.

For instance:

- NIEM Core defines an Activity—this is the object ***nc:ActivityType.***
- Justice domain models an Arrest as an Activity, i.e., extends ***nc:ActivityType***, and adds elements that are specific to the arrest of an individual—this is the object ***j:ArrestType.***
- Immigration domain extends the Justice domain's Arrest, ***j:ArrestType***, to describe the arrest of an alien—this is the object ***im:ImmigrationArrestType.***
- A reference schema could model the relationship between an illegal alien and a traffic stop, using constructs defined in the Structures namespace by simply connecting activities and persons.

Consider the following scenario in which a reference schema defines a traffic stop as an activity (see Figure 48, below). The schema also defines an association between a traffic stop and an illegal alien as an **ActivityPersonAssociation**. The definitions for IllegalAlien, TrafficStop, and TrafficStopIllegalAlienAssociation are contained within the reference schema. The object IllegalAlien is shown to be of type ***im:IllegalAlien*** (which, in turn, points to a person in NIEM Core). The object TrafficStop is modeled as an Activity (again in NIEM Core). Both Person and Activity objects are tagged with the s:id attribute discussed above. The TrafficStopIllegalAlienAssociation object, which in reality is an object of type **ActivityPersonAssociation**, contains two references—s:ref attributes discussed above with the ActivityReference containing the Activity object's s:id attribute and the PersonReference containing the Person object's s:id attribute.

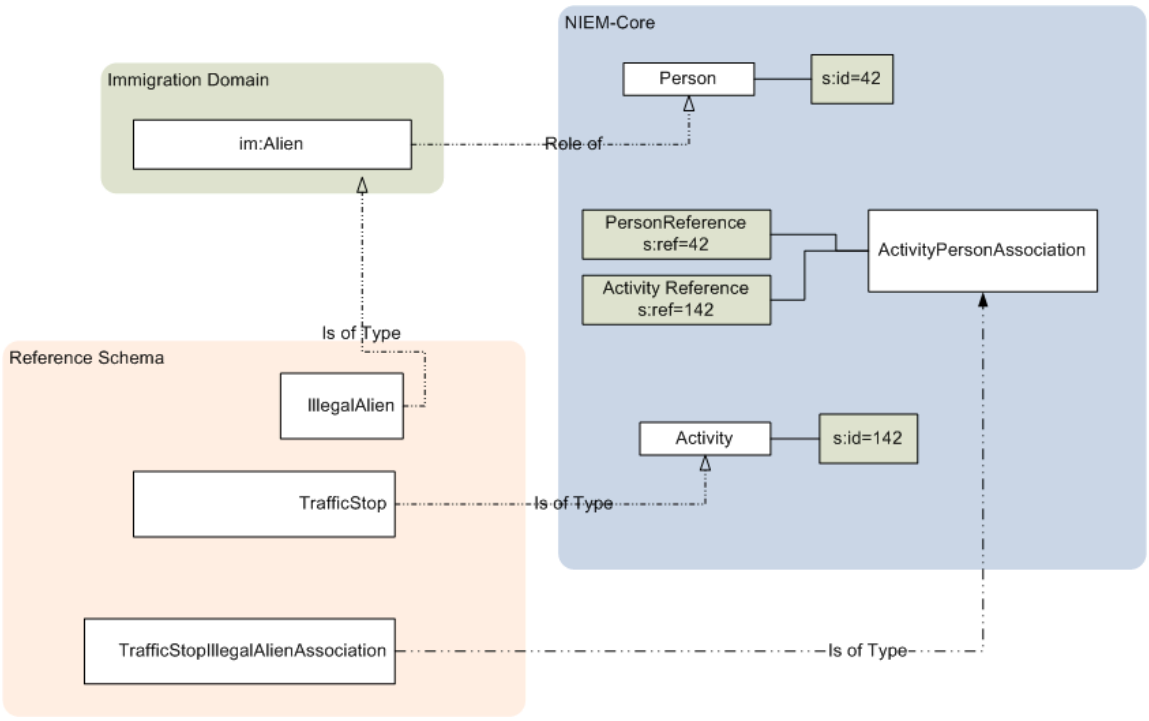


Figure 48: Use of Elements From Structures in Reference Schema Documents.

When you construct a reference schema that reflects a particular information exchange, you could use or extend the element from all of the namespaces. These relationships are illustrated in Figure 49. The components in the domains are shown to be dependent on the components in NIEM Core. Components in the external code sets are independent of the components in NIEM. Components in reference schemas can utilize components in the domains as well as in NIEM Core. External code sets are also available to the applications.

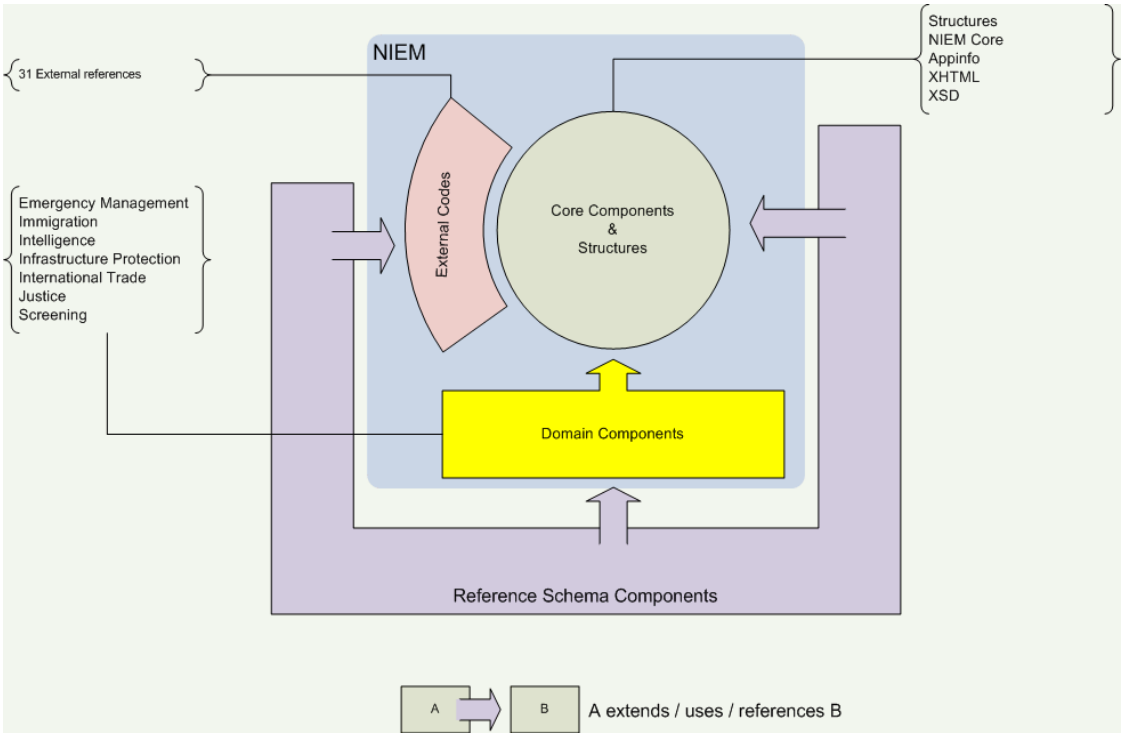


Figure 49: NIEM Components.

6.2 Namespaces

This section takes a closer look at the namespaces that constitute NIEM and provides a roadmap for navigating through the data model. The NIEM Web site (www.NIEM.gov) provides online tools for navigating the data model. In addition, the Web site provides the data model in the form of a spreadsheet. For some uses, the spreadsheet is easier to navigate. Refer to the online tools and the spreadsheet for a comprehensive listing of various components of NIEM.

The following subsections present an overview of the namespaces that comprise NIEM. Where appropriate, a few illustrative components are discussed individually.

6.2.1 Structures

The Structures namespace provides support for fundamental NIEM linking mechanisms, as well as provides base types for the definition of NIEM-conformant types. The namespace and the prefix are defined as follows.

Prefix: s	**http://niem.gov/niem/structures/2.0**
Schema constructs for use by NIEM-conformant schemas to provide consistent definitions and functionality.	

Table 3: Content Definition Namespace.

Some of the key components in this namespace are as follows:

- ***id***: The ***id*** attribute is used to define XML IDs for objects in NIEM and NIEM-conformant data models. These IDs could be associated with any data elements or even metadata elements.
- ***ref***: The ***ref*** attribute is used to refer to XML elements that have XML IDs, ***id*** attribute referenced above. The combined use of ***id*** and ***ref*** attributes makes it possible to refer to data elements/objects without having to contain them within objects.
- ***AugmentationType***: The ***AugmentationType*** type is a base type for all augmentations. An augmentation may have metadata and an ID. Typically, domains use domain-specific augmentations to extend the information contained in the base types. The use of augmentation has been covered in the previous sections.
- ***ReferenceType:*** The ***ReferenceType*** type is the type of all reference elements within NIEM-conformant schemas. This type provides only a reference attribute to reference an object defined elsewhere. It includes an ***id***, as the link established by a reference element may need to be identified, and it includes link metadata, as an element of this type establishes a relationship between its context and the referenced object. It does not contain metadata because it does not itself establish the existence of an object; it relies on a definition located elsewhere. This object is used when any object is to be referenced. For example, consider the following XML segment. This snippet is from the niem-core.xsd schema document and illustrates how a reference is created to an ***nc:Person*** object. It should be noted that the reference to the person is really of the type ***nc:ReferenceType***. The following XML snippet shows the definition of the ***nc:PersonReference*** from NIEM.

```
<xsd:element name="PersonReference" type="s:ReferenceType">
       <xsd:annotation>
              <xsd:documentation>A human being.</xsd:documentation>
              <xsd:appinfo>
                     <i:ReferenceTarget i:name="PersonType"/>
              </xsd:appinfo>
       </xsd:annotation>
</xsd:element>
```

Figure 50: Definition of nc:PersonReference in NIEM.

The following XML code segment shows the definition of the ReferenceType object. As can be seen, the ReferenceType object only contains the s:id and s:ref elements discussed above.

```
<xsd:complexType name="ReferenceType" final="#all">
       <xsd:attribute ref="s:id"/>
       <xsd:attribute ref="s:ref"/>
       <xsd:attribute ref="s:linkMetadata"/>
</xsd:complexType>
```

Figure 51: Definition of the s:ReferenceTypeNIEM Core.

The NIEM Core namespace contains more than 200 objects defined within it. These cover the gamut of content from representations of activities to vehicles. The namespace and the prefix are defined as follows.

Prefix: nc	http://niem.gov/niem/niem-core/2.0
NIEM Core includes both Universal (U) and Common (C) components. The identities for U and C components in Core are maintained with metadata.	

Table 4: NIEM Core Namespace.

While the NIEM Core namespace contains several objects, it is useful to consider the underlying concepts behind the objects. As has been mentioned before, the content in the NIEM Core namespace, for the most part, is used to model the following:

6.2.1.1 Activity

This provides the basis for representing a broad variety of content in several domains. It is useful to think of an activity as something that spans a period of time, i.e., actions, events, and processes. The following is an illustration of the content that is modeled as an activity.

- ***em:AlarmEventType*** (Emergency Management domain)
- ***im:TransferType*** (Immigration domain)
- ***it:ArrivalType*** (International Trade domain)
- ***j:ArrestType*** (Justice domain)
- ***nc:ProgramType*** (NIEM Core namespace)

6.2.1.2 Person

The ***nc:PersonType*** object represents a human being. The Person object includes several components that describe the various aspects of a person. For instance, NIEM provides a comprehensive description of a person's physical attributes such as eye color, hair color, race, ethnicity, physical features, etc. In most cases, the NIEM model supports the use of standardized code sets, such as FBI codes to describe eye color, to ensure a consistent description of a person regardless of the origin of the information. However, NIEM also permits the use of descriptive text in place of these code sets to support the rare cases in which the code sets are not adequate. Scenarios in which this is permitted can be easily identified by the specification of "abstract" for property types.

The ***nc:PersonType*** is used to represent an individual in several situations. For example:

- ***ip:Crew*** (Infrastructure Protection domain)
- it:CrewMember
- j:ActivityOfficial
- j:EvidenceCollector

In addition to being primary objects, the Person object is also used in several contexts as a reference to a person. For example:

- ***im:AlienType*** (Immigration domain) contains an element which is a reference to a person (discussed above) called the ***nc:RoleOfPersonReference***
- ***nc:LienHolder*** (NIEM Core namespace) also contains the person reference

The above examples are only illustrative and, as such, meant to draw out basic concepts and ideas. For a comprehensive listing of the object references, the reader is referred to the NIEM Web site and the documentation provided therein.

6.2.1.3 Document

A *document*, represented by ***nc:DocumentType*** object, is an extremely useful construct in the area of information exchange. This enables modelers to represent metadata related to business processes which, while being peripheral to the subject matter being discussed, are essential to the working of information systems in practice.

The ***nc:DocumentType*** object contains, among other things, information about the author (***nc:DocumentAuthor***), location of the document, ***nc:DocumentLocation*****,** and ***nc:DocumentLocationURI*** reference numbers (useful in tracing electronic documentations. In addition, metadata such as timestamps associated with the filing, transmission, and reception of documents are also represented by data elements in the ***nc:DocumentType*** object.

6.2.1.4 Item

An *item*, in NIEM parlance, refers to an *article* or *thing*. An item shows up in several situations. For example:

- nc:DrugType extends from nc:ItemType
- it:PackageItem is of nc:ItemType
- nc:VehicleType extends nc:ItemType

The item object is used to represent concepts such as ownership, possession, value, location, status, etc.

6.2.1.5 Location

There are numerous instances in which an accurate and precise description of location is required; to this end, NIEM provides several methods of describing a location. The ***nc:LocationType*** object includes the following representations of location:

- **Address location:** This is provided by the property ***nc:LocationAddress***, which supports structured description of a street address (***nc:StructuredAddressType***). In addition, a descriptive address location is also supported.
- **Highway location:** This is provided by the property *nc:LocationHighway.*
- **Latitude/Longitude:** The use of latitude/longitude, as specified by GPS systems, is supported through the use of ***nc:LocationTwoDimensionalGeographicCoordinate***
  http://niem.gtri.gatech.edu/niemtools/ssgt/SSGT-GetNamespace

.iepd?namespaceKey=np-5g; http://niem.gtri.gatech.edu/niemtools/ssgt/SSGT-GetProperty.iepd?propertyKey=np-xe property.

6.2.1.6 Organization

NIEM defines an organization as a data type for a body of people organized for a particular purpose. Organizations are useful in representing entities as broad ranging as criminal gangs and businesses. The examples of organization within NIEM are:

- j:CourtType
- ***it:MasterType*** (Master/operator of the vessel or other means of transport)

Organizations are also used to represent the senders of documents, owners of property, and parties in legal proceedings.

The ***nc:OrganizationType*** object contains data elements used to represent information, such as an organization's name, location, contact information, establishment and termination dates, description of the organization's activity, and category/classification of the organization.

6.2.1.7 Associations

NIEM provides over a hundred types of associations. Associations are used:

- to associate items to activities
- to associate activities to persons
- to associate persons with one another
- to associate documents with one another
- to associate documents with persons
- to associate items with documents (such as lien documents)
- to associate an organization with a location

Refer to the online tools and documentation at www.NIEM.gov for a complete listing.

6.2.1.8 Navigating through NIEM

Consider the following when looking for data types and objects to use in a reference schema:

- What information does a piece of data represent?
  - Is the underlying object a person, an organization, an item, an activity, or a document?
  - Does the information represent a relationship between two entities?
- Does the information represent metadata?

  Based on the above considerations, the user could start at the appropriate place within NIEM. For instance:
  - Documents are described by the ***nc:DocumentType*** object.
  - Human beings are described by the ***nc:PersonType*** object.
  - In a similar manner, items, activities, locations, and organizations are represented by ***nc:ItemType***, ***nc:ActivityType***, ***nc:LocationType***, ***nc:OrganizationType***, respectively.
  - When dealing with information that is specialized to a domain, the modeler should consider objects that are derived from the above objects in the NIEM Core namespaces. It is also permissible for domains to import and extend objects from other domains. In either case, the basic methodology should consist of trying to determine the underlying idea being represented and looking for the appropriate existing object for the purpose. Often, the spreadsheet referred to above is very useful, since it lists objects by inheritances clearly, and a modeler can determine at a glance all extensions (across all domains) of a given NIEM Core object.

6.2.1.9 Attributes in NIEM Core

The NIEM Core namespace also includes several attributes that provide precise metadata about the information being represented by the data object in question.

A few examples of the attributes are:

- ***nc:confidenceNumeric***: A decimal value that indicates belief in the accuracy of the tolerance.
- ***nc:currencyCode***: A unit of money or exchange. This is actually represented by the code set ***iso_4217:CurrencyCodeSimpleType***. This is typically available when monetary values, such as the value of cargo, or bail amounts are described.
- ***nc:partialIndicator***: This is a Boolean value and is used to indicate only partial information was available at the time this information was being discussed.
- ***nc:truncatedIndicated***: This is a Boolean value and is used to indicate that the system sending this information truncated the string in question.

6.2.2 Domains

NIEM contains seven domains that represent the specialization of information/content provided in the NIEM Core namespace within certain business segments. These are discussed as follows:

6.2.2.1 Emergency Management

Prefix: em	http://niem.gov/niem/domains/emergencyManagement/2.0
Emergency Management domain models information pertaining to emergency responders, alarms, hospitals, resources, etc.	

Table 5: Emergency Management Domain.

The objects in the Emergency Management namespace are used to model various aspects of emergency response. For instance:

- ***em:AlarmEventType***, an extension of ***nc:ActivityType***, is used to describe an alarm event, response to the alarm, etc.
- ***em:HospitalType***, an extension of ***nc:OrganizationType***, is used to describe the aspects of a hospital that are of interest to emergency response personnel (such as bed capacity, capability, etc).
- ***em:ResourceInformationType*** is used to model information pertaining to resources—requested/responding/dispatched—that are involved in handling an emergency.

6.2.2.2 Immigration

Prefix: im	http://niem.gov/niem/domains/immigration/2.0
The Immigration domain models information pertaining to aliens.	

Table 6: Immigration Domain.

The objects in the Immigration namespace represent content that describes, among other things, types of aliens (students, visitors), detention or processing of aliens, etc. For instance:

- ***im:AlienStudentAdmissionType*** describes the academic program for which an alien is being granted the visa, details about any dependents, etc.
- ***im:AlienType*** describes various attributes about an individual that are unique to the immigration domain, such as ***im:AlienIDDetails***, ***im:AlienDeportationIndicator***, etc.

6.2.2.3 Intelligence

Prefix: intel	http://niem.gov/niem/domains/intelligence/2.0
The Intelligence domain in NIEM contains extensions and augmentations that are useful in describing intelligence data.	

Table 7: Intelligence Domain.

The objects described in the Intelligence domain deal with the identification of individuals and information about cautions and/or reasons for which a person might be of interest to an agency.

For example:

- ***intel:AgencySubjectInterestType*** captures information about the agency that expressed interest in a given individual, the category of "interest," etc.
- ***intel:BiometricAugmentationType*** augments the nc:BiometricType.
- ***intel:IdentityAssociationType*** associates an authenticated identity of an individual with some primary identifier.

6.2.2.4 Infrastructure Protection

Prefix: ip	http://niem.gov/niem/domains/infrastructureProtection/2.0
Describes objects pertaining to protecting the infrastructure of the country.	

Table 8: Infrastructure Protection Domain.

This enables the modeling of information pertaining to threats to infrastructure facilities. This namespace contains detailed information only about air transportation infrastructure. In addition, objects that describe sectors to which an infrastructure belongs are also defined in this namespace. For instance, a bridge is considered to be a part of the road transportation infrastructure in the same manner in which an airport is a part of the air transportation infrastructure.

The following data elements illustrate the information content in this domain:

- ***ip:AirlineType*** describes an airline organization.
- ***ip:AssetType*** describes an asset that is a part of an infrastructure. For instance bridges, dams, and airports are all considered to be assets.

6.2.2.5 International Trade

Prefix: it	http://niem.gov/niem/domains/internationalTrade/2.0
The International Trade namespace contains objects that represent the various actors, items, goods, facilities, etc., that pertain to international trade.	

Table 9: International Trade Domain.

The objects and concepts utilized in this namespace model are buyers, sellers, consignments, shipments, customs declarations, exporters, importers, transportation means, etc.

Typical examples in this domain are as follows:

- ***it:CommodityType*** represents such aspects as cargo description, dangerous good identifiers, etc.
- ***it:DeclarationType*** describes an item's declared weight, invoice amount, customs identification, etc.

6.2.2.6 Justice

Prefix: j	http://niem.gov/niem/domains/jxdm/4.0
The justice namespace models various actors, events, and processes in the area of criminal justice.	

Table 10: Justice Domain.

There are numerous objects that represent various aspects of the criminal justice process. At a high level, the objects represent:

- **Individuals/Organizations:** For example:
  - j:EnforcementOfficialType
  - j:VictimType
  - j:SupervisionSubject
  - j:CourtOfficial
- **Activities:** For example:
  - j:ArrestType
  - j:CourtEventType
  - j:SentenceType
- **Associations:** For example:
  - j:ActivityArrestAssociationType
  - j:ActivityEvidenceAssociationType
  - j:SubjectPersonAssociationType

6.2.2.7 People Screening

Prefix: scr	http://niem.gov/niem/domains/screening/2.0
Namespace containing information related to the screening/processing of immigrants and nonimmigrants.	

Table 11: People-Screening Domain.

The People Screening domain provides harmonized information sharing content within the Screening Portfolio of DHS. The Screening namespace is initially being populated with person screening information for immigrant and nonimmigrant person types who have been encountered and identified by the Screening Portfolio Components. Screening expands on encounter-related NIEM elements currently included.[19]

The Screening namespace touches on immigration-related concepts because it represents arrival or departure processes pertaining to aliens. The screening domain also includes extensions/augmentations to biometrics.

Illustrative examples:

- ***scr:BiometricAugmentationType*** augments representations of biometric data in the NIEM Core domain with extensive source metadata, qualitative metadata, etc.
- ***scr:BenefitApplicationType*** extends the DocumentType object and describes an application for benefit such as Naturalization, Asylum, Permanent Residency, or Temporary Worker.
- ***scr:ChargeAugmentationType*** has extensions to indicate whether foreign authorities have been notified of the charge in question (an issue with offenders/accused persons who are not U.S. citizens).
- ***scr:DepartureType*** represents the exit of the person from the United States.

6.3 Standard Code Lists

NIEM contains 31 standard code lists borrowed from standards external to NIEM. These are imported into standard namespaces under NIEM through the use of proxies. The primary purpose of these code sets is to ensure that activities, items, and attributes are described in a consistent manner. This, in turn, will ensure that there is no ambiguity when different parties describe the same event, person, item, or location.

For example:

- The code sets from the FBI namespace (http://niem.gov/niem/fbi/2.0, referred to by the prefix ***fbi,*** in NIEM) contain codified values that describe a person's eye color, hair color, race, and ethnicity. They also contain codified representations of automobile makes, models, and styles, etc.
- The code sets from the USPS namespace (http://niem.gov/niem/usps_states/2.0, referred by the **usps** prefix) contain the familiar two-character codes for states in the United States of America.

[19] As defined in the NIEM spreadsheet.

- There are several code sets to denote countries. These ensure that countries are clearly and unambiguously specified. For instance, addresses that denote the destination points for cargo and citizenship information for aliens would utilize these code sets. These are found in following namespaces:
  - http://niem.gov/niem/fips_10-4/2.0 (prefix ***fips_10-4***)
  - http://niem.gov/niem/fips_5-2/2.0 (prefix ***fips_5-2***)
  - http://niem.gov/niem/fips_6-4/2.0 (prefix ***fips_6-4***)

7 Building NIEM-Conformant Data Exchanges

The goal of NIEM conformance is for the sender and receiver of information to share a common, unambiguous understanding of the meaning of that information. Conformance to NIEM ensures that a basic set of information (the NIEM components) is well-understood and carries the same consistent meaning across various communities. The result enables a level of interoperability to occur that would be unachievable with the proliferation of custom schemas and dictionaries.

In order to begin exchanging information, partners will need to develop data exchanges, also known as Information Exchange Packages (IEPs), which are then documented as Information Exchange Package Documents (IEPDs). An IEPD is a complete definition of an IEP. It is a compilation of documentation that can be understood both by the producer of the information exchange, as well as the receiver. Generally, it is composed of schemas (for data exchange) and documentation (for understanding the business context and usage). This section describes the process that can be used to guide the development of a NIEM IEP and associated IEPD.

The process described in this section is a guide or template for the development of IEPs and IEPDs and is intended to be customized as necessary. It provides a useful starting point in project planning and can help to set high-level expectations regarding milestones, resources, and timelines. Specific requirements to satisfying NIEM conformance are detailed in Appendix A.

The goals of the process are as follows:

- Communicate the specific requirements to building NIEM-conformant exchanges to promote compatibility and consistent development
- Ancillary artifacts that address the information needs of a broad range of project stakeholders, including project sponsors, business experts, business and IT managers, and technologists.
- Mechanism for synthesizing the domain/business knowledge of subject-matter experts.
- Artifact reuse across projects by improving artifact consistency.
- Leverage open industry standards that are familiar to most business analysts, architects, and other technology professionals.
- Work with standards-based tools that are readily available in the public domain or at low cost, allowing integration projects to avoid high licensing costs and vendor lock-in.
- Share valuable lessons learned and best practices from Reference IEPD development projects so that those lessons need not be relearned on future projects.

NIEM IEPs can be developed to share information within a single domain (intra domain exchange) or across multiple business domains (cross domain exchange).

The remainder of this section will discuss the process to create NIEM IEPDs. The following sections provide examples of some of the concepts using the Amber Alert, which is a partnership between law enforcement agencies, broadcasters, transportation agencies, and the wireless industry to activate an urgent bulletin in child abduction cases.

The activities and tasks outlined in the next few pages apply to both cross-domain and intra domain modeling unless otherwise noted.

7.1 NIEM Information Exchange Package Document (IEPD) Development Lifecycle

This subsection discusses the process for creating the IEP and IEPD for information exchanges. As previously mentioned, an IEPD is a complete definition of an Information Exchange Package (IEP).

Figure 52 illustrates the steps required to develop an IEPD.

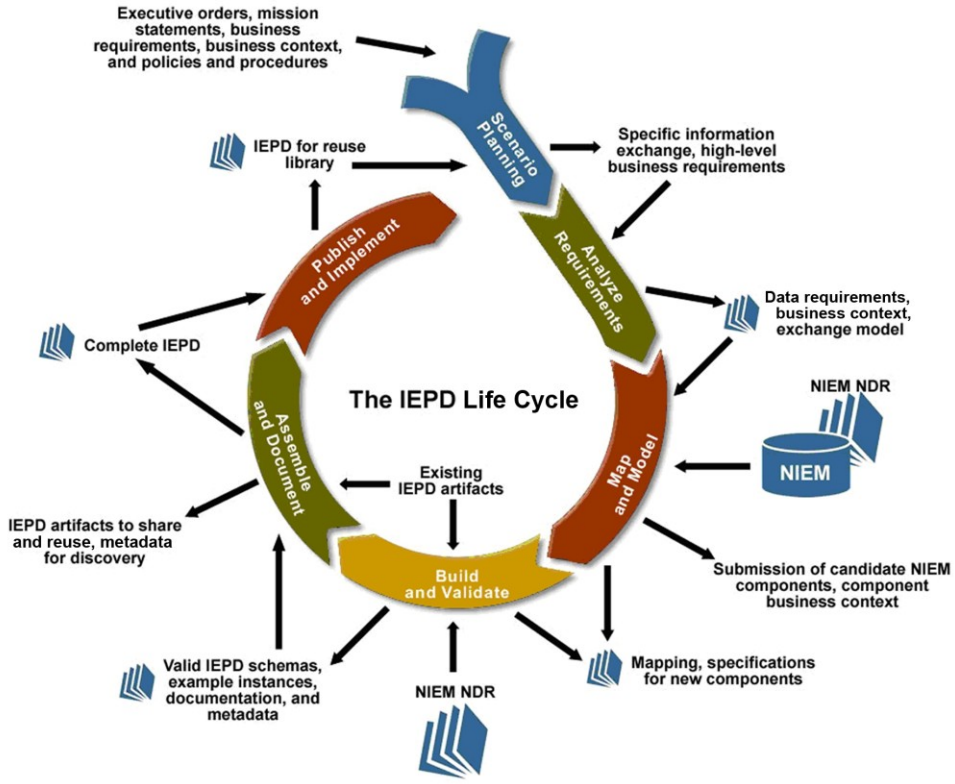


Figure 52: IEPD Lifecycle.

As presented above, the NIEM IEPD lifecycle has six major steps. These are:

- **Scenario Planning:** This step enables the identification of scenarios requiring exchange of information, business requirements, and business context.
- **Analyze Requirements**: This step defines the business and data requirements associated with an information exchange for which NIEM is being used.

- **Map and Model**: This step comprises activities that are focused on creating the exchange model and mapping the model to the NIEM model. During this step, components that are not found in the NIEM model are forwarded to the NIEM Committee for harmonization and promotion into the NIEM model.
- **Build and Validate**: The activities in this step focus on the creation of a set of NIEM-conformant XML schemas that implement the document structure identified in the previous steps.
- **Assemble and Document**: The activities in this step focus on assembling all artifacts of the IEPD and completing any remaining documentation.
- **Publish and Implement:** This step focuses on publishing the IEPD and using it in a production mode.

The next few pages discuss each of the steps above in greater detail. Additional guidance for this process has been released by the U.S. Department of Homeland Security.[20]

7.2 Step 1: Scenario Planning

The purpose of this step is to plan the project, establish the process, provide for human and technology resources, and identify information exchange business requirements. Identifying information exchange business requirements is best accomplished through identifying current and planned information exchanges, scenario-based planning, and information exchange mapping. An agency does not usually need to share all the information that the agency collects with other agencies or domains. Identifying precisely **what** information is exchanged between agencies will be determined by modeling relevant business practices of the domains through ***scenario-based planning*** and ***information exchange mapping***.

The first task in the scenario planning step is **establishing a clear vision** for the schema development project. The goal of a vision statement is to determine, at a high level, what the scope of the project is, who the stakeholders are, and what business-oriented results they should expect to achieve by the time the project has been completed. If the project has important contextual attributes, those should be noted as well. For example, a project may be a follow-on to a previous project, or it may leave important business objectives out of scope, with the intent to address those objectives on future projects.

The second task in this step is **establishing a process** to be followed on the project (such as the process described in this module, with appropriate modifications to address specific needs or risks on particular projects). The process should identify deliverable milestones (e.g., domain model, NIEM mapping, schemas, and sample instances) and target dates on which those milestones are expected to be reached. In setting target dates for each milestone, you should set proper expectations with stakeholders and other project participants. In particular, the dates should be viewed as reasonable targets rather than exact predictions, since as the domain model unfolds, hidden complexities may be uncovered that compel either an adjustment to project scope or to milestone dates.

[20] http://www.niem.gov/pdf/NIEM_Guidance_v1_0.pdf.

The third task in this step is **developing scenarios**. Scenarios describe the business context of events, incidents, or circumstances in which information must be exchanged between agencies and/or domains. The scenario may be a terrorist attack on a city, for example, and careful elaboration of that scenario will identify critical operational points at which information must be shared between two or more agencies for effective prevention, response, and remediation. Scenarios may be used to depict current (i.e., "as is") information exchange practices among involved agencies, thereby identifying gaps, impediments, and other flaws in business processes and data exchange. They may also be used to characterize potential future (i.e., "to be") environments that envision broader and more expansive information sharing, as well as changes in business practice. An example of a scenario is described from the justice domain below.

Sample Scenario from National Association of State Chief Information Officers (NASCIO)[21]

Functions appear in *italics*; **systems** appear in **bold**, and documents appear in underline.

1. A police officer submits a *query* to the **statewide warrant system** and discovers from the *response* that the subject of his car stop is wanted on an outstanding arrest warrant.
2. The police officer arrests the subject and completes and signs (digitally) an arrest report that describes the incident, offense, arrest circumstances, and the arrestee. The arrest report is stored in the **police information system**, which *pushes* either the full arrest report or certain segments and elements of information to the sheriff's **booking information system**.
3. The arrestee is taken to the sheriff's office to be booked. The sheriff's **booking information system** uses the arrest report number to *pull* the arrest report from the **police information system** and uses data from that report to (partially) complete the booking document.
4. The sheriff's **booking information system***,* using personal-description data in the arrest report and biometric identifiers, *pulls* information from the **state criminal history records repository**. Based on information from the criminal history record, the jailer makes a security decision and enters that decision into the sheriff's **booking information system**, which assigns an appropriate cell.
5. The sheriff's **booking information system** uses information from the arrest report and booking document to generate a standard press release and *pushes* it to the department's **Web page**, which posts information regarding arrests recorded over the past 24 hours.
6. The sheriff's **booking information system** uses information from the arrest report and booking document, together with the booking fingerprint images and mug shot, to *push*

[21] NASCIO IJIS ConOps http://www.nascio.org/nascioCommittees/ea/ConOps2003.pdf.

required identification and arrest information to the **state criminal history records repository**, where the arrest event information is *pulled* into the arrestee's criminal history record . . . .

The fourth task is **creation of the project work group**. The project work group should consist of the following members:

- **Business Subject-Matter Experts** who represent the interests of the stakeholders identified in the project's vision statement. These experts provide crucial business perspective on the information content of the exchange, as well as its context. They should have expertise in the business in general and the information exchange in particular. If existing enterprise software systems are involved in producing or consuming information in the exchange, it is useful if the users of these systems are represented on the work group.
- **XML Experts** who have an in-depth understanding of XML and XML schema technologies.
- **A Facilitator** with both business and XML expertise (though perhaps less of each of these than the other work group members) who can lead the work group through the process. The facilitator's responsibilities also include leading the domain modeling sessions, so it is important to choose a facilitator and a modeling technique that make this possible.

Finally, once work group members have been chosen and milestone dates established, the facilitator should arrange for meeting resources (to support in-person and remote/telephonic meetings, as appropriate) and other communication tools. A project Web site has proved useful for many work groups; the Web site should contain a list of work group members and their contact information, a project plan identifying milestones, and a repository for project artifacts.

Detailed tool needs will be identified in later sections within this module; however, the following basic tools will be needed:

- Tools to support domain modeling.
- Tools to support mapping of the domain model to NIEM; most work groups have found that a simple spreadsheet works well.
- Tools to support creation of valid schemas.

Table 12 provides a summary of the scenario planning tasks.

Inputs:	Mission Statements Business Context Policies and Procedures
Responsible Party:	IEPD Project Lead
Participants:	Business Subject-Matter Experts XML Experts Facilitator
Artifacts Created:	Project Charter Action Items Scenarios High-Level Business Requirements

Table 12: Scenario Planning Tasks.

7.3 Step 2: Requirements Analysis

The second step in the IEPD development process consists of requirements analysis. During this step, the selected IEPD is further elaborated to understand and document the business context and data requirements. This step concludes with the development of a domain model.

The first task in this step is **defining the context and content of the information**. The ***context*** of the exchange identifies who is involved in the exchange (agencies/partners), the events that trigger this exchange and under what conditions, and what happens after the exchange occurs (the next business process). The ***content*** of the exchange identifies the information (at a high level) that is part of the exchange. In addition to the context and content of information, critical policy requirements, such as privacy, security, priority, frequency, urgency, complexity, and confidentiality, should be captured and documented. A variety of tools and methodologies can and should be utilized to define the information requirements.

This task needs to be conducted using the work group defined in the earlier step. The output from this task provides the input for the next task—to build the domain model.

Domain modeling is an analysis activity through which business subject-matter experts reach agreement on the contents and structure of the exchange.

The output of the domain modeling step is, not surprisingly, a domain model. This model can take many forms, as discussed below. However, the form of the model is not as important as its ability to facilitate the building of consensus among the work group. That is, the domain model is primarily a communication device—not for communication between the business experts and the schema-building technicians as separate groups but, rather, for communication among the work group as a whole. The work group builds a domain model to represent, in a technology-agnostic way, what the information content of the exchange document needs to be.

The business subject-matter experts, in particular, need to be able to build consensus around the model. That is, the model needs to be something with which nontechnical participants can agree or disagree. This factor has important implications for the style and form of the domain model. In particular:

- The model should, in specifying information structures, use names and definitions that have meaning to the work group.

- The model should be built in a format and language that is easily understandable by everyone on the work group.
- The model should be easily consumable by the work group members, ideally without installation of special tools or specialized training.

The next few paragraphs describe the options to develop the domain model for the exchange.

7.3.1 Domain Modeling Options

On reference IEPD projects, work groups have had success building domain models in three formats:

- a "flat" textual model in the form of a spreadsheet
- an informal graphical model
- a more formal graphical model built with the Unified Modeling Language (UML)

Other model formats are certainly possible, and this is not meant to be an exhaustive list of the possibilities.

In choosing an option, facilitators and work groups should bear the following in mind:

- Choose an option with which the facilitator is familiar. An IEPD development project is usually not the place for someone to learn UML or any other technique.
- Be pragmatic rather than dogmatic about selecting the option. Choose something that works for the particular work group; there is no one right way to build a domain model.
- Measure the effectiveness of the selected modeling option early and often, and adjust as necessary. Avoid letting the option become a barrier to communication or consensus, and remember that the point of the domain model is primarily for communication *within the work group*.
- Also bear in mind the opportunities for reuse of the domain model in other contexts. For example, is the IEPD intended to be a statewide baseline, which will be further customized by county or municipal jurisdictions? If so, then closer adherence to open standard notations (like UML) and ubiquitous tools may be warranted.

Each of these options will now be examined in detail.

7.3.1.1 Spreadsheet Modeling

A spreadsheet domain model consists of a "flat" (one-dimensional) list of data elements, grouped into logical document sections or subject areas. Typically, the first column of the spreadsheet contains the subject area, and subsequent columns may contain more fine-grained subject areas. After the subject-area column(s), the name of the data element is listed, along with a definition.

Role	Is A	Has A	Is A	Description	Cardinality
MissingPerson	PersonType			Details about a person whose whereabouts are unknown	
		name	PersonNameType	A name by which a person is known.	1,1
		alternateName	PersonNameType	A name by which a person is known.	0,1
		dateOfBirth	Date	A date a person was born.	1,1
		race	String	A classification of a person based on factors such as geographical locations and genetics.	0,1
		ethnicity	String	A cultural lineage of a person.	0,1
		tribalAffiliation	String	An affiliation of a person to a tribe.	0,1
		caveat	String	A warning or caution.	0,1

Figure 53: An Example of a Domain Modeling Spreadsheet.

The ***advantages*** of spreadsheet modeling are as follows:

- There are no new tools to acquire and learn—almost everyone has access to Microsoft Excel or an equivalent spreadsheet tool.
- There is no modeling notation to learn—the model works by simply listing data elements and grouping them into logical subject areas.

The ***disadvantages*** of spreadsheet modeling are as follows:

- The spreadsheet structure is, in effect, a notation in and of itself that has to be learned (though it is quite simple).
- There is no universally agreed-upon heuristic for determining document sections (or how many section "levels" there should be) or for naming them.
- For large documents, the lack of a graphical presentation can result in "missing the forest for the trees."
- It is difficult to indicate reusable structures within the document unless you name the structures and reference the type names later in the modeling and/or create hyperlinks to predefined structures to prevent duplication.
- Relationships between entities are difficult to represent in a one-dimensional list.

7.3.1.2 Informal Graphical Modeling

An informal graphical model consists of a diagram that depicts domain entities (things) as symbols, with arrows drawn between entities to indicate relationships. These diagrams are essentially "concept maps" in which the concepts being linked are components or "sections" of an exchange document.

Interpreting the symbols on an informal graphical model is similar to interpreting classes and relationships on UML class diagrams.

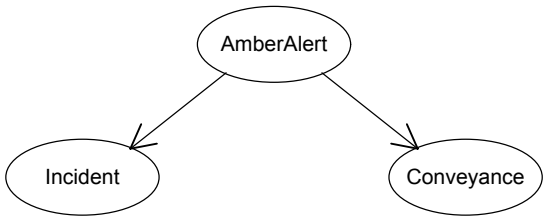


Figure 54: Informal Graphical Model.

The ***advantages*** of informal graphical modeling are as follows:

- This technique offers a graphical presentation that can improve communication of the context of each data element.
- There are generally no new tools to acquire and learn—Microsoft PowerPoint and Visio work well.
- This technique is very useful for high-level structural overviews, since it avoids inundating the reader in details.

The ***disadvantages*** of informal graphical modeling are as follows:

- This technique is not effective at documenting the fine details of document structure.
- Notation needs to be invented to document important concepts such as cardinality and inheritance.

7.3.1.3 UML Static Structure (Class) Diagrams

The Unified Modeling Language (UML) defines a diagram type, called a class or static structure diagram, which depicts domain entities and their attributes as well as the relationships between entities. This type of diagram has built-in facilities for documenting entities at high or low levels of detail and for documenting important concepts such as cardinality and inheritance.

The ***advantages*** of modeling with a UML class diagram are as follows:

- It offers a graphical presentation that can improve communication of the context of each data element.
- It offers a precise and formal notation for depicting document structure but, at the same time, is simple enough to be accessible to a wide range of stakeholders without requiring significant training or explanation.
- It supports object-oriented concepts inherent in NIEM and XML schema.
- It is supported by widely available, low-cost tools (as well as commercial tools that cost more but have more robust features).
- It has widespread adoption in the technology industry and is familiar to most analysts and developers.

The ***disadvantages*** of modeling with a UML class diagram are as follows:

- It requires the project to select a UML modeling tool.

- It requires that the facilitator be very familiar with both UML and the selected tool.
- Work group participants unfamiliar with UML will require coaching (though usually this is minimal).

If a work group elects to build its domain model using UML, choice of modeling tool becomes a critical factor in the long-term success of the project. It is recommended that users bear the following points in mind when selecting a tool:

- The modeling tool should be easy to use and familiar to the facilitator.
- The modeling tool should support creation of UML-compliant class diagrams.
- The modeling tool should support publishing of diagrams as ordinary image files (e.g., JPG or PNG).

The modeling tool should support exporting the model's structure in XML Metadata Interchange (XMI) format so that the structure can be exchanged with other modeling tools if necessary.

Figure 55 presents an example of an Amber Alert Domain model using a UML modeling tool.

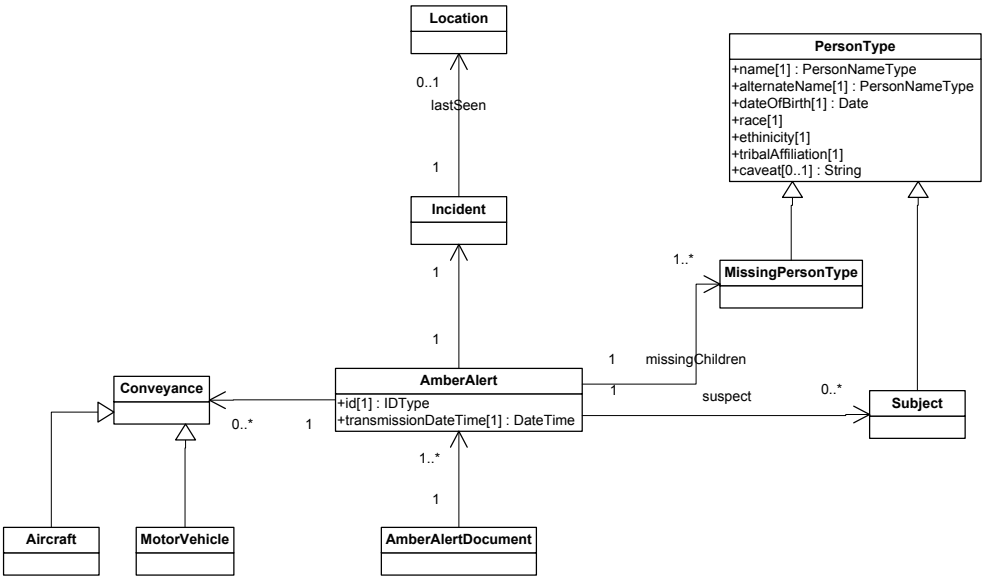


Figure 55: Extract From the Amber Alert Domain Model.

Table 13 provides a summary of the requirements analysis tasks.

Inputs:	Exchange Details High-Level Business Requirements
Responsible Party:	IEPD Project Lead
Participants:	Business Subject-Matter Experts System Users Facilitator

Artifacts Created:	Data Requirements Business Context for Data Domain Model

Table 13: Requirements Analysis Tasks.

7.4 Step 3: Mapping and Modeling

The third step in the IEPD development process involves associating domain model concepts and structures with types and elements in NIEM.

This process of associating domain model concepts and structures with types and elements in the NIEM model is called ***mapping***. During this task, each concept or class in the domain model, as well as each individual property or data element, needs to be associated with a particular type or element in the NIEM Schema. During the mapping exercise, there are three potential outcomes. These are:

- **Matches—**Matching components can involve those in which the component names may differ but in which the data components themselves are semantically and structurally equivalent, i.e., there is a one-to-one mapping between NIEM and the source component.
- **Partial Matches—**Partial matches can arise when there are similarities but also some differences between data components. These differences can include semantic or structural mismatches, element naming collisions, or mismatches at the value set, data type, or lexical levels. For partial matches, it is necessary to document the need for extension or refinement of existing data components.
- **No Matches—**Data components with no matching NIEM data components comprise a set of additional element types that are candidates for insertion into NIEM. Depending on the nature of the potential inclusion in the model, recommendations may include adding a new or subordinate type, adding an element, extending a value set, modifying a data type, or lexical representation, renaming data components, or revising a definition. For components that do not match at all, a NIEM-conformant component must be created, following the rules specified in the *NIEM Naming and Design Rules* (NDR).[22]

For partial matches or no matches, the extension techniques outlined in Sections 0 and 6 of this document will be used to add local extensions to the IEP. These may become candidates for later submission to NIEM. The mechanism to submit these extensions for inclusion into NIEM is described in detail in the Harmonization and Promotion task in Section 7.8 below.

The mapping artifact is designed to record these associations and extensions so that they can easily be input into the schema-building process.

[22] http://www.niem.gov/files/NIEM-NDRno_lines.pdf.

If a spreadsheet is used for the domain model, the mapping artifact will just be the addition of columns to identify the association of the business data element to an element in NIEM. There will generally not be a separate mapping artifact in this case.

To perform the mapping, it is necessary to be able to search quickly and efficiently through NIEM for types and elements that match the concepts in the domain model. Tools to accomplish these searches are available on the NIEM.gov Web site and the Wayfarer tool, available in both online and localized versions from the National Center for State Courts (NCSC), can be utilized to search NIEM. Please refer to Appendix B for a discussion on the NIEM tools. It is recommended that each work group and facilitator try each method to determine which method (or combination of methods) works best for its situation.

Class	Property or relationship	NIEM Path	Inherits From
AmberAlertDocument		AmberAlert	DocumentType
	AmberAlert		
AmberAlert	*AmberAlert*		
	ID	AmberAlert/DocumentIdentification/IdentificationID	
	transmissionDateTime	AmberAlert/DocumentPostDate	
	status	AmberAlert/DocumentStatus/StatusText	
	caveat	AmberAlert/CaveatText	
	alertLanguage	AmberAlert/DocumentLanguageCode	
	MissingPerson	AmberAlert/AmberAlertChild/RoleOfPersonReference@s:ref=<REF_TO_CHILD>	
	ShortMessage	AmberAlert/AmberAlertDeviceMessage	
	Subject	AmberAlert/AmberAlertSuspect/RoleOfPersonReference@s:ref=<REF_TO_SUBJECT>	
	Incident	AmberAlert/AmberAlertIncident	
	Conveyance	AmberAlert/Vehicle	
	PersonConveyanceAssocia	AmberAlert/PersonConveyanceAssociation	
	ChildSuspectRelation	AmberAlert/ChildSuspectRelationship	
AmberAlertIncident	*Incident*		IncidentType
	circumstancesDescription	AmberAlert/AmberAlertIncident/IncidentObservationText	
		AmberAlert/AmberAlertIncident/ActivityDate	
	lastSeenDateTime	AmberAlert/AmberAlertIncident/ActivityDateRange	
	directionOfTravel	AmberAlert/AmberAlertIncident/AmberAlertDirectionOfTravelText	
	Location	AmberAlert/AmberAlertIncident/IncidentLocation	
Subject	*Person*		PersonType
	id	AmberAlert/Person@s:id=<REF_TO_SUBJECT>	
		Person details omitted for clarity	
PersonConveyanceAssocia	*Association*		AssociationType
		AmberAlert/PersonConveyanceAssociation/AssociationBeginDate	
		AmberAlert/PersonConveyanceAssociation/AssociationEndDate	
		AmberAlert/PersonConveyanceAssociation/ConveyanceReference@s:ref=<REF_TO_VEHICLE>	
		AmberAlert/PersonConveyanceAssociation/PersonReference@s:ref=<REF_TO_SUBJECT>	
MissingPerson	*Person*		
	id	AmberAlert/Person@s:id=<REF_TO_CHILD>	
		Person details omitted for clarity	
Vehicle	*Vehicle*		
	id	AmberAlert/Vehicle@s:id=<REF_TO_VEHICLE>	
		Vehicle details omitted for clarity	
ChildSuspectRelationship	*Association*		AssociationType
		AmberAlert/ChildSuspectRelationship/AssociationBeginDate	
		AmberAlert/ChildSuspectRelationship/AssociationEndDate	
		AmberAlert/ChildSuspectRelationship/PersonReference@s:ref=<REF_TO_CHILD>	
		AmberAlert/ChildSuspectRelationship/PersonReference@s:ref=<REF_TO_SUBJECT>	

Figure 56: Example of an Amber Alert Mapping Document.

Figure 56 illustrates the manner in which a mapping document is created. The various columns identify the class, the property or relationship, the NIEM path, and the object from which this inherits.

As mentioned earlier, partial matches and no matches for components create the need for these components to be integrated into NIEM. Section 7.8 discusses the harmonization and promotion of these components into NIEM.

Table 14 provides a summary of the map and model tasks.

Inputs:	Data Requirements Business Context for Data Domain Model
Responsible Party:	XML Experts
Participants:	XML Experts
Artifacts Created:	Mapping Document XML Extensions

Table 14: Map and Model Tasks.

7.5 Step 4: Building and Validating

The next step in the IEPD development process is creating a set of exchange-specific NIEM-conformant XML schemas that implement the data model created for the exchange in the previous steps. The principal input into the schema-building process is the mapping artifact mentioned earlier. The output is a set of NIEM-conformant schemas.

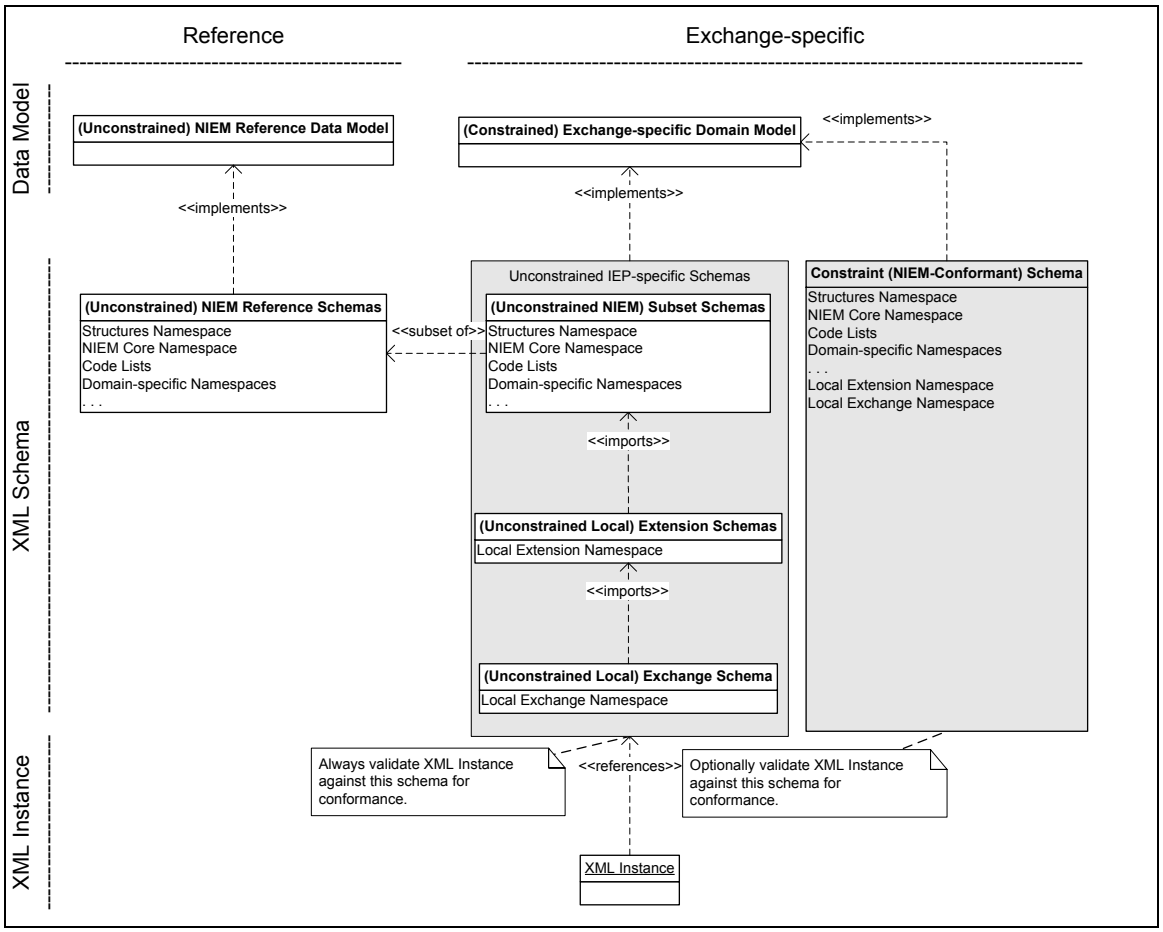
There are three main schemas involved. They are as follows, along with short descriptions. The following pages will address these matters in more detail.

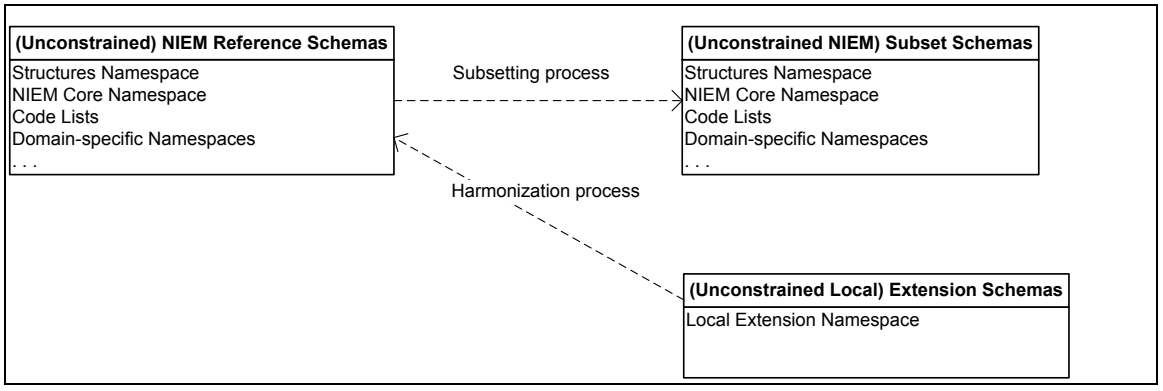
- **NIEM Reference Schemas**: These are the full unconstrained NIEM XML Schemas.
- **Unconstrained IEP Specific Schemas**: These are the unconstrained schemas specific to an exchange. They consist of three subschemas:
  - **Subset Schemas**: This unconstrained subset of the NIEM Reference Schema contains just those types and elements that are used in the exchange.
  - **Extension Schemas**: These schemas are optional and contain just those local types and elements that are used in the exchange.
  - **Exchange Schema**: This is the unconstrained schema containing the document type and element for the exchange.
- **Constraint schema**: This is the constrained version of the Unconstrained IEP-Specific Schemas. It contains additional constraints that capture the local business rules included in the data model for the exchange.

The output of the schema development process for an exchange is, at a minimum, the constraintless schema. Optionally, the constraint schema may also be created. The extension schema—which is a part of the constraintless schema—will usually also add to or modify the local reference schema. The schema development process for an exchange does not affect the NIEM reference schema in any way.

The following three-part diagram shows the various schemas involved in the schema development process and the relationships between them. NIEM prescribes a two-step process when validating an XML instance for conformance to the schema for an exchange. First, the XML

instance should be validated for conformance to the constraintless schema. Then, optionally, the XML instance may also be validated for conformance to the constraint schema.


Reference
Exchange-specific
Data Model
XML Schema
XML Instance
(Unconstrained) NIEM Reference Data Model
(Constrained) Exchange-specific Domain Model
<<implements>>
<<implements>>
<<implements>>
Unconstrained IEP-specific Schemas
(Unconstrained) NIEM Reference Schemas
Structures Namespace
NIEM Core Namespace
Code Lists
Domain-specific Namespaces
. . .
<<subset of>>
(Unconstrained NIEM) Subset Schemas
Structures Namespace
NIEM Core Namespace
Code Lists
Domain-specific Namespaces
. . .
Constraint (NIEM-Conformant) Schema
Structures Namespace
NIEM Core Namespace
Code Lists
Domain-specific Namespaces
. . .
Local Extension Namespace
Local Exchange Namespace
<<imports>>
(Unconstrained Local) Extension Schemas
Local Extension Namespace
<<imports>>
(Unconstrained Local) Exchange Schema
Local Exchange Namespace
Always validate XML Instance against this schema for conformance.
<<references>>
Optionally validate XML Instance against this schema for conformance.
XML Instance


(Unconstrained) NIEM Reference Schemas
Structures Namespace
NIEM Core Namespace
Code Lists
Domain-specific Namespaces
. . .
Subsetting process
(Unconstrained NIEM) Subset Schemas
Structures Namespace
NIEM Core Namespace
Code Lists
Domain-specific Namespaces
. . .
Harmonization process
(Unconstrained Local) Extension Schemas
Local Extension Namespace

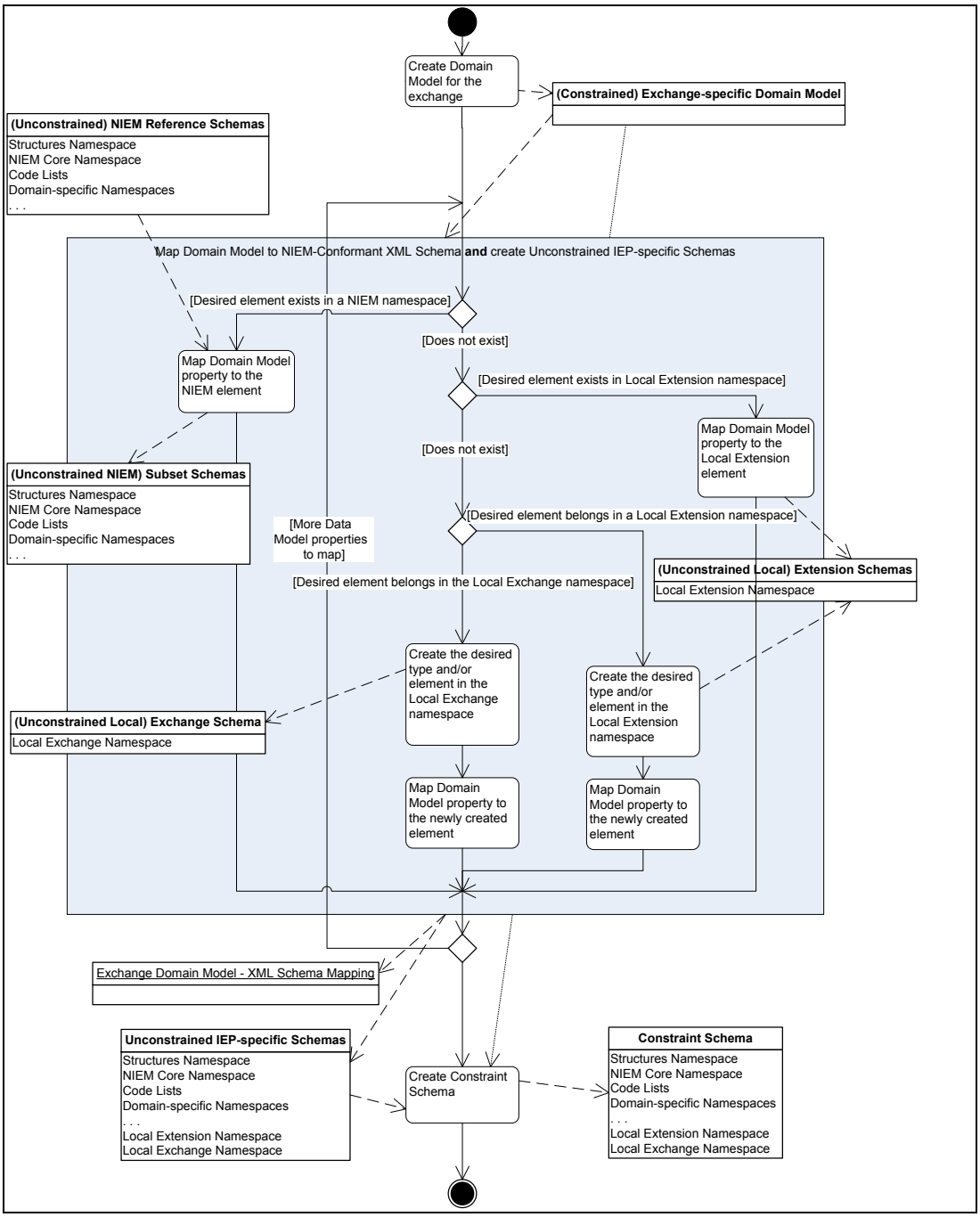


Figure 57: Example of Schema Development Process.

7.5.1 Subset Schema

The NIEM subset schema contains just those types and elements from the full NIEM schema that are needed for the exchange plus any types or elements used by those types, and so on.

The use of a subset schema, as opposed to the full reference schema, serves two purposes:

- It can improve performance when parsing and validating instances, since there is less schema information for the parser or other tool to process.
- It reduces the amount of information about the IEPD's data structure needed by developers and tools at design time.

There is a single fundamental rule to which all subset schemas must adhere, namely:

Instances that validate against a subset schema must also validate against the full NIEM schema.

In practice, this means that conformant subset schemas must have the following characteristics:

- They do not add types or elements beyond what is in NIEM.
- They do not change the types of elements or the base types of derived types from what is in NIEM.
- They do not change the name of any type or element in NIEM.
- They do not change the order of elements that occur within a type in NIEM.
- They are in the same namespace as the full NIEM.

The following actions are permissible in conformant subset schemas:

- Restriction of enumerations in code list schemas (for example, to restrict them to just a set of codes used in a jurisdiction).
- Removal of imports of unused schemas.
- Removal of unused attributes.
- Omission of documentation structures (i.e., annotation and documentation elements) from the full NIEM.
- Adjustment of cardinality constraints, as desired.

It is certainly possible to create conformant subset schemas by hand. However, for exchange documents of significant size, hand-crafting subset schemas that satisfy all the conditions would be tedious and error-prone. Consequently, the recommended approach for building subset schemas is to use the online Subset Schema Generation Tool (SSGT).[23]

The SSGT presents the schema designer with an interface that permits searching through NIEM for desired types and elements. When these types or elements are found, the user may

[23] http://justicexml.gtri.gatech.edu/subset_tool.html.

mark them for inclusion in the subset. When types or elements are marked for inclusion, the SSGT applies the appropriate rules and selects any dependent types and elements as well. This frees the designer from having to manage all of the dependencies.

After marking all of the desired elements and types, the designer can generate the subset. The result is a zip file containing the relevant NIEM schemas, which include NIEM Core, structures, and domain schemas, plus all of the code list schemas that are referenced in the subset.

7.5.2 Extension Schema

In many cases, an exchange document will require data structures that do not exist in NIEM. These structures will be identified in the mapping step, since they will not map to anything in NIEM. Such structures should be defined in extension schema.

Extension schemas are provided as a mechanism to create reusable local components in the exchange schema. If a local component is expected to be used only in a single exchange, it may be defined in the exchange schema. If the local component is expected to be used in multiple exchanges, it can be defined once in the extension schema and reused (by importing and referencing) in the various exchange schemas. This is often simpler than defining and keeping track of the component in each exchange schema in which it is used.

The extension schema defines an IEPD-specific namespace (sometimes called a "local" namespace). **Because the types and elements in an IEPD-specific namespace are not part of NIEM, there is no equivalent of the SSGT for extension schemas.** Extension schemas generally must be developed "from scratch," by writing XML schema constructs.

It is recommended that every type in an extension schema extend some type in NIEM (even if it only extends ComplexObjectType, ReferenceType, or AugmentationType.) Extending NIEM types fosters reuse of NIEM's semantics and also enforces consistency in use of metadata objects. To make NIEM namespace types (and elements) available in an extension schema, the extension schema must import the schemas for the appropriate NIEM namespaces. If the extension schema uses other namespaces within NIEM (e.g., a codelist namespace) or outside NIEM, it must import schemas that define those namespaces as well.

After reviewing the NIEM data model, you may find that the concept to be represented in the information exchange does not exist in NIEM. In this case, NIEM provides three techniques for creating new NIEM types to represent the new concept:

- Composing a new NIEM type from a collection of NIEM properties.
- Extending an existing NIEM type to create a new NIEM type.
- Augmenting an existing NIEM type to create a new NIEM-derived type.

These techniques are discussed in detail in GTRI's *Techniques for Building and Extending NIEM XML Components*[24] document.

[24] http://www.niem.gov/Techniques_for_Building_and_Extending_NIEM.txt.

7.5.3 Exchange Schema

An exchange schema is a schema that contains the root element and the root type for the IEPD, plus any local extensions that are not already defined in an extension schema. Since this schema is IEPD-specific, it must define an IEPD-specific namespace. The root type in this document schema defines the top-level structure of the instance document. In most cases, this root type will be an extension of the NIEM DocumentType, since DocumentType is intended to represent "documents." The exchange schema will import the extension schema (if it exists) and the subset schema.

7.5.4 Constraint Schema

The full NIEM reference schema provides a common language through which its users can communicate in a manner which is semantically consistent. However, because NIEM is defined for a large and varying group of users, it is impossible to embed all possible constraints and usages of that language into the reference schema. Therefore, the reference schema is unconstrained, very optional, and overinclusive. It defines the language but does not attempt to control exactly how people are going to use it.

As discussed in the previous section, the schema subset generated by the SSGT allows the user to identify only those types and elements required for the information exchange. However, the types and elements included in the subset still adhere to the NIEM philosophy of being "optional and overinclusive." In particular, the cardinality of all the elements is still "zero-to-many," meaning each element can occur zero, one, or many times within its parent structure. In many cases, the exchange needs to restrict this cardinality further. This kind of cardinality restriction is an example of a business rule that can be implemented in a constraint schema.

Constraint schemas are mechanisms to embed constraints and business rules so that they may be validated by an XML schema validator. Before they are described, however, it must be noted that the use of a constraint schema is completely optional; there are other ways of checking these business rules, and, in some cases, constraint schemas may be completely unnecessary. Business rules can be validated outside of XML schema by embedding them in applications, XML Stylesheets (XSLT), Schematron (an assertion language), or other methods. Alternatively, it may not matter whether the constraints are met or not. Systems can choose to parse out the valid portions of the data they receive and discard the rest. For example, suppose an organization requires the last name of a person to be no more than 30 characters. If it receives an instance document with a last name of 35 characters, it may choose simply to truncate the last name to its requirement rather than rejecting the instance document as invalid. This illustrates the notion that there are many different ways of dealing with constraints and business rules. An XML schema may not be the most powerful or rigorous method of defining such constraints, but it can be sufficient for validating common kinds of constraints. Furthermore, an XML schema precludes the introduction of new validators or other tools into the information exchange process.

A constraint schema is a simple way to define local business rules. Cardinality constraints, as discussed above, provide the primary constraint applied in constraint schemas. It is also possible to create further subsets in the constraint schema (e.g., removing elements, types, or enumeration facets), if that is desirable. However, usually the creation of any type or types of subset(s) is performed in the subset schema. It is important to note that the constraint schema

does not change the NIEM namespace. It also does not import the subset schema; rather, it replaces it.

The schema is defined in the same namespace as the NIEM reference schema and defines the same content but with the addition of constraints. Constraint schemas are often built beginning with a copy of a schema subset. From that starting point, the constraint schema is modified; for instance:

- Changes can be made to the default NIEM cardinality.
- Facets can be added that constrain allowable data values (e.g., maximum name length = 30 characters, minimum age value = 18, license plate number must match pattern "[A-Z]{3} d{4}"—three uppercase alpha characters followed by a space and four digits).
- Choice blocks can also be inserted (e.g., either a person's social security number or both the name and the date of birth must appear in the instance).
- Types can be constrained differently based on how they are used in the document (e.g., changes can be made to a constraint schema such that only a person's name and badge number can be used with an enforcement official but a full set of person descriptors can be used with a subject).

The constraint schema does not add or change the semantics defined in NIEM. It is not the place to add local extensions or content.

The NIEM reference schema and/or schema subset still defines the language being used. The constraint schema further defines local business rules about the NIEM content that can appear in the instances.

The primary rule that must be followed when building constraint schemas is:

Instances that validate against a constraint schema also validate against the full NIEM schema.

This means that the only changes one can make to a constraint schema are those that do not prevent instances from validating against the full NIEM reference schema or a valid subset. Things that one cannot do in a constraint schema include changing element names, modifying the order or hierarchy in which elements appear, and modifying the definitions or semantics of NIEM content. For example, changing the name of NIEM element "PersonGivenName" to "firstName" in a constraint schema is not allowed. Any instance that appears with element "firstName" replacing element "PersonGivenName" because of changes made to the constraint schema would not be a valid NIEM instance.

To ensure that invalid changes are not made to the constraint schema, even unintentionally, it is important that instances be validated against the full reference schema or schema subset to check for NIEM language consistency, in addition to validating against the constraint schema, which only checks for local business rules. This concept of making two passes to validate, whereby each pass checks for different constraints, is called multipass schema validation.

The only change made during the different validation passes is to the schemaLocation attribute—the reference schema or the schema subset and the constraint schema will have different file names (and possibly different paths). When an instance is validated against both the reference or schema subset and the constraint schema, it is not necessary to check the same thing twice. Anything that has already been checked by the reference or schema subset validation pass can be dropped by the constraint schema. For example, it is not necessary to validate VehicleMakeCode twice in an instance. The reference to the large NCIC code set can be dropped from the constraint schema.

It is important to note, again for emphasis, that the constraint schema has the same NIEM namespace as the full reference schema or the subset. It does not import the subset or reference schema; it is a local copy of NIEM that users can modify to add constraints to NIEM content.

7.5.5 Validating IEP Schemas

To validate the IEP schemas, the IEPD developer can use an XML validator tool to ensure that the example XML instances and stylesheets validate the schemas according to the NIEM reference architecture. The validator tool can be used to ensure that both conformance and constraint validation, if applicable, are accomplished.

Inputs:	Domain Model Mapping Document Extensions
Responsible Party:	XML Experts
Participants:	XML Experts
Artifacts Created:	XML Schemas

Table 15: Build and Validate Tasks.

7.6 Step 5: Assembling and Documenting

To further define the IEPD, additional documentation including business rules, change log, and metadata is also needed. The outputs of this step are the valid schemas, example instances, documentation artifacts, and metadata.[25]

Once all of the schemas, documentation, metadata, and other files have been captured, the IEPD can be generated based on the NIEM IEPD specification format. The NIEM IEPD tool can assist with this process.

The assembly step prepares and packages all required files for this IEPD into a single self-contained, self-documented, portable archive file. Included in this archive are all schemas (subset, extension, exchange, code lists, etc.), sample instances (XML), stylesheets (XSLT), and documentation (business requirements, diagrams, etc.). The archive also contains a metadata file prepared to an XML specification for NIEM IEPD metadata and an XHTML catalog file that opens in a standard browser and indexes the contents of the archive. These IEPD artifacts are discussed more fully in Section 8 of this document. By unpacking the archive and opening the catalog file, a user can browse through the entire package. Furthermore, the specification for

[25] January 9, 2007, Page 37 of 78 NIEM Concept of Operations Version 0.5.

the catalog is formal enough that the format and purpose of each file in the IEPD can be distinguished. This means that a NIEM IEPD could be machine-processed for various automated purposes.

The output of this step is a complete IEPD that provides reference for other users. An IEPD is considered to be NIEM-conformant if it:

- Imports and references a NIEM namespace or a correct subset.
- Uses the appropriate NIEM data component (i.e., does not create a duplicate of one that already exists).
- Is semantically consistent (i.e., uses NIEM data components in accordance with their definitions and does not use an element to represent data other than what its definition describes).
- Applies the NIEM architecture and constructs (i.e., NIEM NDR) correctly and consistently.

NIEM conformance allows stakeholders to share accurate and reliable information that has the same meaning for the receiver as for the sender.

Inputs:	All Schemas, Mapping Document, Scenario(s), Requirements
Responsible Party:	IEPD Project Lead
Participants:	Business Subject-Matter Experts Technical Staff
Artifacts Created:	Required IEPD Artifacts

Table 16: Assemble and Document Tasks.

7.7 Step 6: Publishing and Implementing

The final output of the IEPD lifecycle is an IEPD that is published and available for search, discovery, and reuse. IEPD developers have the option to publish their IEPDs to their own repository; to an industry repository, such as the IEPD Clearinghouse;[26] or, preferably, to register and publish them through NIEM. Details on how to publish to the IEPD Clearinghouse or NIEM may be found on their respective Web sites. Nevertheless, all IEPDs are portable and self-documented and can be registered anywhere.

The NIEM PMO and the NIEM Communications and Outreach Committee (NC&OC) will promote awareness and encourage use of IEPDs through direct outreach with stakeholders, as well as by developing a strategy for interfacing with government IEPD registries. IEPDs being promoted by the NIEM PMO will conform to the NIEM NDR and will align to strategic priorities, including national priority information exchanges identified and designated by the National Priority and Exchange Panel and those sponsored by an authoritative source (e.g., Global Rap Sheet).

[26] http://www.it.ojp.gov/iepd/.

Inputs:	IEPD Artifacts
Responsible Party:	IEPD Project Lead
Participants:	All Users and Project Stakeholders
Artifacts Created:	Required IEPD Metadata (See Table 20: IEPD Metadata)

Table 17: Publish and Implement Steps.

7.8 Data Harmonization and Refactoring

As new data requirements are identified through business needs for information exchanges, NIEM may expand to incorporate those requirements. In many cases, new requirements are represented by some mixture of existing components and new components. Integration of these new components into NIEM, identified during the previous step, occurs through a process called ***harmonization***.

In NIEM, harmonization is a process for modeling, adding, and integrating new data components in ways that minimize differences, remove duplication, resolve conflicts, reduce the degree of variation, and achieve consistency across all existing components. Harmonization seeks to bring new content into NIEM while reestablishing or maintaining standardization and uniformity across all parts of the data model under the NDR.

The submission of candidate NIEM components should occur as soon as the new components are identified. Often, these new data requirements are identified during the mapping and modeling processes. Once components are submitted, a process will review any updates received by the NIEM community. Also, this may give the identifier of the candidate NIEM components the possibility of incorporating the updates into future IEPDs.

Harmonization guidelines provide direction for evolving the stock of NIEM data components in alignment with other NIEM principles and rules of this NDR. During harmonization, it may be necessary to ***refactor*** some components.

Refactoring is a technical process that applies sets of atomic transformations to existing components to change their structure for the purpose of improving or reestablishing model integrity, consistency, or harmony. This process usually occurs during harmonization.

Inputs:	Data Requirements Business Context for Data Domain Model
Responsible Party:	IEPD Project Lead
Participants:	XML Experts
Artifacts Created:	Mapping Document

Table 18: Data Harmonization and Promotion.

8 IEPD Artifacts

An IEPD is a set of artifacts consisting of normative exchange specifications, examples, metadata, and documentation encapsulated by a catalog that describes each artifact. The entire package is archived as a single compressed file. When uncompressed, the catalog is a hyperlinked index into the IEPD and can be opened in a standard browser. The user may use the catalog to overview the IEPD contents or to open each individual artifact, provided the appropriate software required to open a given artifact is installed. Assembling the artifacts into a final IEPD using NIEM tools is discussed in Appendix B.

IEPD Artifact	Description	File Type/ Examples	Required/ Optional
Exchange Files (normative XML)			
Subset schema	A directory structure containing the IEP-specific subset of the full NIEM schemas.	xsd	R
Wantlist	User requirements—an SSGT-generated XML file containing user-selected NIEM components specific to an IEP. It saves the current state of a NIEM subset schema so that it can be later modified and/or regenerated.	xml	R
Exchange schema	Base document schema that defines the XML root element and is generally named after the IEPD itself. Also known as the document schema, reference schema, or root schema.	xsd	R
Constraint schema	Constraints for separate constraint validation path.	xsd	O
Extension schema	Specification for extended components—separate local namespace of components not contained in NIEM.	xsd	O
Sample XML instance	Example instance—may be multiple and may reference optional stylesheet.	xml	O
Sample stylesheet	Example stylesheet for display of instances, which may be multiple.	xsl	O
Documentation			
Master documentation	May include purpose, business requirements, what, when, why, how to, etc. Guidelines are needed for master documentation content, and the following indented items are possible documents that can be contained within the master documentation or broken out as individual files.	txt, doc	R
Business requirements	Itemized descriptions that may also contain business rules.	txt, doc	O
MOUs	Memoranda of understanding among participating agencies.	txt, doc	O
Endorsement letters	Documentation from professional or governmental organizations that confirm support. Refer to *Endorsement* in metadata.	txt, doc	O
Methodology and tools	Used to build IEPD and may contain URLs or references to tools, methodology, or documentation.	txt, doc	O

IEPD Artifact	Description	File Type/ Examples	Required/ Optional
Change log	Record of cumulative changes from previous IEPD versions. The initial IEPD simply records its creation date.	xml, txt, doc	R
Testing and conformance	Description and results of validation and conformance testing performed—may include testing output or products.	txt, doc	O
Domain model	Domain model in standard open format (xmi, vsd, zargo) and standard open graphic (jpg, pdf, etc.) that is likely a Unified Modeling Language (UML) model.	vsd, xmi, zargo, jpg, pdf, etc.	O
Use case model	Use case diagram in standard open format and standard graphic, likely UML.	vsd, xmi, zargo, jpg, pdf, etc.	O
Business rules	May be (1) plain or structured English, (2) written into master documentation, (3) Schematron or other formal business rule language, or (4) generated by a development tool.	xml, txt, doc	O
Mapping (to NIEM components)	Mapping of domain components to NIEM components; tagged with constraints (cardinality, etc.); prefer Component Mapping Tool (CMT).	xls, csv	O
Extended components	Components created because they were not in NIEM—may be part of mapping spreadsheet and include structure and definitions of new components. Prefer CMT.	xml, xls, csv	O
Catalog Files			
Catalog	List of artifacts in the IEPD that is machine-readable; in an open, portable format; and browser displayable.	xml, xhtml,	R
Metadata	All metadata registered with the IEPD.	xml, xhtml,	R

2731 **Table 19: IEPD Artifacts.**

9 IEPD Metadata

The metadata artifact contains all metadata that the authoritative source wishes to register with an IEPD. This metadata should be specified by an XML schema so that an instance for a given IEPD can be parsed, loaded into a registry, and used to search, discover, and harvest business context and metrics on IEPDs and their artifacts.

Metadata Item	Description	Req/ Option
Descriptive		
URI Universal Identifier	Each IEPD version will have a distinct URI. NIEM will provide a suggested strategy for URIs, but use of strategy is NOT mandatory.	R
Name	Title of this IEPD (e.g., Amber Alert, Prosecutor Arrest Warrant).	R
Summary	Brief summary of this IEPD for short display purposes—maximum of 160 characters including spaces.	R
Security	Security label to indicate treatment/distribution of this IEPD; e.g., for official use only (FOUO), classified, sensitive but unclassified (SBU), public. The default is public, unless otherwise noted.	R
Description	Narrative description of this IEPD—may contain as much detail as you think useful to those with a potential interest in this IEPD.	O
Web site	URL of Web site where this IEPD and related artifacts (e.g., XML schema, documentation, mapping spreadsheets) are posted.	O
Change log data (must be consistent with change log artifact)		
Creation date	Project start date—YYYYMM that planning or work on this IEPD started. Do NOT confuse with date on which you submitted this IEPD information.	R
Version	Version of this IEPD.	R
NIEM version	NIEM version used for this IEPD.	R
Last revision date	Year and month (YYYYMM) this IEPD information was last revised. Do NOT confuse with the date on which the IEPD itself was last revised, generating a new IEPD version.	O
Next revision date	Year and month (YYYYMM) this IEPD information is expected to be revised.	O
Status		
Maturity	State of development: 1. Entry level; under development with minimum documentation (see artifacts). 2. Complete; being tested and in limited use with draft documentation. 3. In production; fully documented and endorsed for use in official exchanges.	R
Status	Description or additional information related to current state of this IEPD.	O
Schedule	Information about the development schedule for this IEPD; e.g., "Development started YYYYMM; draft planned YYYYMM; completion planned YYYYMM."	O
Endorsements	Names and acronyms of professional or governmental organizations that support this IEPD as official business information exchange package.	O

Metadata Item	Description	Req/ Option
Sponsors	Name of organization(s) that sponsored, contributed, or participated in the development of the IEPD.	O
Navigation		
Lineage	IEPDs from or with which this IEPD was derived or built, identified by URI. This is not normal version control.	O
Relationships	URIs of other IEPDs and their relationship to this IEPD; should not duplicate other attributes such as Lineage, LoB, Organization, etc.	O
Keywords	Search terms that would not otherwise be in other metadata attributes (e.g., Georgia's Levi's ***Call for an Amber Alert***).	O
Business Context		
Domains	Primary domains or line(s) of business (LoB) that this IEPD covers.	R
Purpose	A short description of the business reason for using this IEPD; may include brief statement of scope.	R
Message exchange patterns	Category of transaction for which this IEPD is designed and used: query/response, message, publish/subscribe, document, etc.	O
Communications environment	Description of the primary communications environment(s) for which this IEPD was designed; for example, wireless, satellite, broadband, T1.	O
Exchange partner categories	Types of organizations that would use this IEPD.	O
Exchange partners	Names of the organizations that are using this IEPD.	O
Process	Business process(es) during which this IEPD is exchanged.	O
Triggering event	Event(s) that cause this IEPD to be exchanged.	O
Conditions	Condition(s) under which this IEPD is exchanged.	O
Authoritative Source		
Authoritative source organization name	Organization responsible for owning and maintiaining the IEPD or Information Exchange related artifacts and metadata; includes both full name and acronym, as appropriate, to enhance discovery.	R

2738 **Table 20: IEPD Metadata.**

Appendix A: Data Model Conformance Guidelines

Introduction

NIEM is a data model and reference dictionary. This means it is not a rigid standard that must be used exactly as it is in its entirety. NIEM was designed as a core set of building blocks that are used as a consistent baseline for creating exchange documents and transactions across government. While an XML schema rendering of the entire model exists, it is not a requirement for NIEM conformance that this entire schema be used for validation. Nonetheless, there are several informal conformance requirements.

The goal of NIEM conformance is for the sender and receiver of information to share a common, unambiguous understanding of the meaning of that information. Conformance to NIEM ensures that a basic core set of information (the NIEM components) is well-understood and carries the same consistent meaning across various communities. The result enables a level of interoperability that would be unachievable with the proliferation of custom schemas and dictionaries.

These conformance rules serve as guidelines for any agency utilizing NIEM to implement its information sharing exchanges. Grantees that are developing interagency XML-based exchanges must comply with the special condition language contained in the grant and follow the associated NIEM implementation guidelines outlined below.

It is important to understand that NIEM conformance is intended for the XML-based exchange. It is not intended to place any conformance standards on legacy databases or database design.

Conformance Rules

The rules for NIEM conformance are as follows:

- Schema instances must validate against the set of NIEM reference schemas. Schemas conformant to NIEM must import and reference the NIEM Schema namespace they need to use (NIEM Core, Justice, etc.) or a conformant NIEM Schema subset. Note that importing the NIEM Justice Domain namespace will cascade to importing NIEM Core. Also, note that if an instance validates against a correct subset of the NIEM reference schemas, it will validate against the NIEM reference schemas.
- If the appropriate component (type, element, attribute, etc.) required for an IEPD exists in NIEM, use that component. Do not create a duplicate component of one that already exists.
- Be semantically consistent. Use NIEM components in accordance with their definitions. Do not use a NIEM element to encapsulate data other than what its definition describes.

- Follow the Information Exchange Package Documentation (IEPD) Development Lifecycle as described in the *IEPD Requirements*[27] and define all required artifacts at each step.
- Adhere to the *NIEM Naming and Design Rules*[28] (NDR) to ensure correct, consistent schema development.

Assistance in Developing NIEM-Conformant Schemas

Further guidance on the proper development of conformant exchange schemas is provided in part by the *NIEM Concept of Operations*[29] (ConOps) and the NIEM NDR. These concepts are still being developed as NIEM continues to grow and mature.

In addition to document support, tools are provided to help simplify conformance when developing exchanges. The Schema Subset Generation Tool[30] (SSGT), along with others, is built to ensure conformant subsets and development without requiring implementers to have detailed knowledge of the formal *Naming and Design Rules*. The NIEM *IEPD Lifecycle* and other best-practice models for developing exchanges take full advantage of these tools to help ensure consistent design and development.

Additional Remarks About Conformance

Information Exchange Packages (IEPs) and the IEPDs that define them conform to NIEM—systems, however, do not. The way data is labeled or used in one system does not affect NIEM conformance. Conformance depends on how data is packaged as XML for an information exchange to be shared between two or more systems.

Use of some NIEM components to exchange information with other agencies does not guarantee conformance to NIEM. Users should be careful to avoid violating conformance Rule 2, listed above. An information exchange either conforms to NIEM or it does not.

Grant Recipients

To support governmentwide information sharing, all recipients of grants from certain government agencies for projects implementing information exchange capabilities using XML technology are required to use NIEM in accordance with the NIEM Implementation Guidelines.[31] These grantees are further required to assemble, register, and make available without restriction all IEPDs and related artifacts generated as a result of the grant to the component registry. Assembly of NIEM IEPDs within the NIEM IEPD Tool is optional. However, NIEM IEPDs must be assembled in accordance with IEPD requirements as specified by the NIEM PMO and must be registered in the IEPD Clearinghouse.[32]

---

[27] http://www.niem.gov/files/NIEM_IEPD_Requirements_v2_1.pdf.

[28] http://www.niem.gov/topicIndex.php?topic=file-NDR-withoutLineNum.

[29] http://www.niem.gov/topicIndex.php?topic=file-conops.

[30] http://niem.gtri.gatech.edu/niemtools/ssgt/index.iepd.

[31] http://www.niem.gov/implementationguide.php.

[32] http://it.ojp.gov/iepd/.

Organizations not receiving federal funding to use NIEM are also encouraged to register their IEPDs in the IEPD Clearinghouse. This will facilitate interoperability of information systems and promote awareness to enhance effective sharing of critical information.

Appendix B: NIEM Tools

Introduction

In developing NIEM exchange specifications, certain tools come into play at various stages of the IEPD development lifecycle. The order in which topics are introduced in this document generally coincides with the order in which the tools will be used during the development of a NIEM exchange and mirrors the IEPD development lifecycle.

NIEM Tools	Description
Universal Modeling Language (UML) Tools	UML tools are used to provide an efficient way of modeling data object classes and components and their attributes and dependencies.
NIEM Data Model Browser	NIEM Data Model Browser enables the user to graphically explore the NIEM model and relationships between data classes (i.e., data types) and data elements.
NIEM Wayfarer	NIEM Wayfarer is a non-NIEM.gov application. It was developed as an alternative to the SSGT for exploring the NIEM model; its elements, attributes, and data types; and the relationships between them.
Subset Schema Generation Tool (SSGT)	SSGT has a strong set of search features that helps map exchange data elements to NIEM and create exchanges.
Component Mapping Template (CMT)	Component Mapping Template (CMT) helps facilitate the mapping of the exchange elements to the equivalent NIEM terms and identifies mapping gaps which form the basis of the extension schema.
Code List Schema Tool	Code List Schema Tool is used to create a NIEM-conformant schema enabling an application to validate XML data against a list of restricted values.
Migration Assistance Tool (MAT)	NIEM Migration Assistance Tool helps convert GJXDM 3.0.x or NIEM 1.0 wantlist to a NIEM 2.0 wantlist.
IEPD Tool	IEPD Tool allows the user to store IEPD for future editing and sharing the IEPD with the public to view the IEPD.
Justice Information Exchange Model (JIEM) Tool	The JIEM Modeling Tool is a non-NIEM.gov application that helps model business processes with best practices, documenting requirements for electronic information sharing, capturing both the information content and business context of information exchanges.

Table 21: List of NIEM Tools.

Help Documentation on NIEM.gov

With the release of Version 2.0 of the NIEM tools, the help documentation has been extensively revised. Almost every screen includes a description of the features of the currently selected tool and provides links to more extensive help documentation. Clicking **help** at the top of the page brings up the **NIEM Tools Help and Features** page. (See Figure 58.) Each tool, along with each of the features, is hot-linked to its own help page with more information on the features, functions, and capabilities.

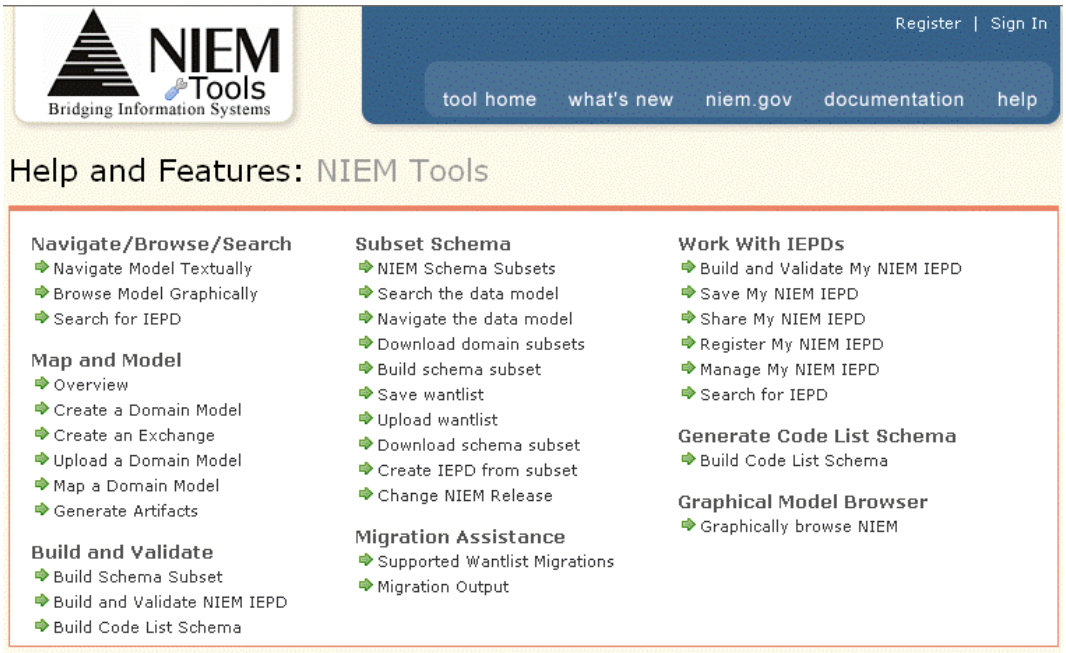


Figure 58: The Tools Help Feature Has Been Greatly Enhanced With NIEM 2.0.

Registering on NIEM.gov

It is not required that you register on NIEM.gov to use the tools; however, if you register, an account is created which provides storage space allowing you to permanently save your IEPD artifacts on the Web site. Registering on NIEM.gov is relatively easy and does not require a lot of personal information.

Justice Information Exchange Model (JIEM) Tool

The Justice Information Exchange Model (JIEM)[33] is a non-NIEM.gov application developed by SEARCH, in partnership with the Bureau of Justice Assistance. It comprises a reference model, methodology, and online tool designed to facilitate integrated justice information systems planning and implementation.

The JIEM consists of four components:

[33] http://www.search.org/programs/info/jiem.asp.

- A conceptual framework for understanding justice system information exchanges.
- A methodology for analyzing current information exchanges and for reengineering information exchanges in an integrated justice environment.
- The JIEM Modeling Tool—a software package to assist justice system practitioners in applying the model to their jurisdictions.
- The JIEM Adult Felony Reference Model—a set of information exchanges common to most jurisdictions.

The **conceptual framework** for understanding justice system information exchanges can be described in five dimensions—process, event, agency, condition, and information. The information dimension includes documents and data elements and is the foundation for information exchanges in NIEM.

The **JIEM methodology** is a structured, formally documented approach for capturing information exchange requirements. It includes both the content of the exchange (the information) and the context (the business processes). In addition, JIEM captures critical policy requirements such as the privacy, security, priority, frequency, and urgency of the exchange.

The **JIEM Modeling Tool** helps model business processes with best practices, documenting requirements for electronic information sharing, capturing both the information content and business context of information exchanges. Leveraging the JIEM Adult Felony Reference Model (described below), the JIEM Tool helps users perform JIEM analysis much more quickly with results more consistent with those of other jurisdictions.

The **JIEM Adult Felony Reference Model** is a set of standard information exchanges that occur in the adult felony environment and are common to most jurisdictions. The reference model has been developed and refined by other JIEM users and provides a common framework that others can build on to model the business processes and information exchanges relevant to their jurisdictions.[34]

Universal Modeling Language (UML) Tools

Information exchange modeling is a way of describing the data components required for an exchange and the hierarchical relationship between those components in a graphical format. A graphical representation of your exchange model makes it easier to share ideas with other working group members and facilitates the collaborative development process of identifying the appropriate subset of data for the exchange. The most popular class of tools used for this process is Universal Modeling Language (UML) tools. The value of using a UML modeler in developing NIEM information exchanges is that it provides an efficient way of modeling data object classes and components and their attributes and dependencies. A number of UML tools on the market provide the functionality needed to model information exchanges.

A full discussion of UML modeling is beyond the scope of this document; however, a Web search on UML modeling tools will yield a wealth of resources on tools, books, and tutorials.[35]

[34] More information about the JIEM Tool is located at: http://www.search.org/programs/info/jiem.asp/.

[35] For example, a number of UML resources are shown at http://www.uml.org/ and http:// umlcenter.visual- paradigm.com/.

The *Objects by Design*[36] Web site lists a wide variety of UML tools with various features, functionality, and price points as well as what features to look for in a UML modeling tool.[37]

Not all UML modeling applications are compatible with the NIEM tools.

As explained in the ***Map Information Exchange*** section beginning on page 98, not all UML tools export the necessary format required by NIEM. Research potential tools carefully before choosing one for developing IEPDs.

Searching and Navigating the NIEM Model

As you begin building information exchanges and go through the process of mapping your data elements to NIEM, you will likely need to search through the model to identify semantic equivalent elements between your data set and the NIEM model. There are a number of ways to explore the model to aid you in this process—the Data Model Browser, several alternate model formats, such as spreadsheets and a database, the NIEM Wayfarer Tool, and the Subset Schema Generation Tool. Each of these tools is described below in more detail.

NIEM Data Model Browser

For a visual and interactive representation of the NIEM model, the NIEM Data Model Browser enables you to graphically explore the NIEM model and relationships between data classes (i.e., data types) and data elements.

The Data Model Browser allows you to:

- Browse different parts of the model to see how properties, types, associations, and their relationships are connected in NIEM
- Visualize the model from a number of key starting points

To access this tool from the NIEM Tools page on www.NIEM.gov, roll over **Search/Navigate Model**, then select **Search/Navigate Model Graphically**.

To use the Data Model Browser, you must have Java Runtime Environment (JRE) installed on your computer.

---

[36] http://www.objectsbydesign.com/tools/umltools_byProduct.html.

[37] http://www.objectsbydesign.com/tools/modeling_tools.html.

The Data Model Browser allows you to view the various object classes contained in NIEM Core, including person, vehicle, organization, location, contact information, document, person associations, and activity associations (see Figure 59). Use the Data Model tab to view the data object and its associated properties. The Type Inheritance tab shows the parent object of the selected object, and the Associations tab shows the derived types of a parent association type (i.e., person or activity association).

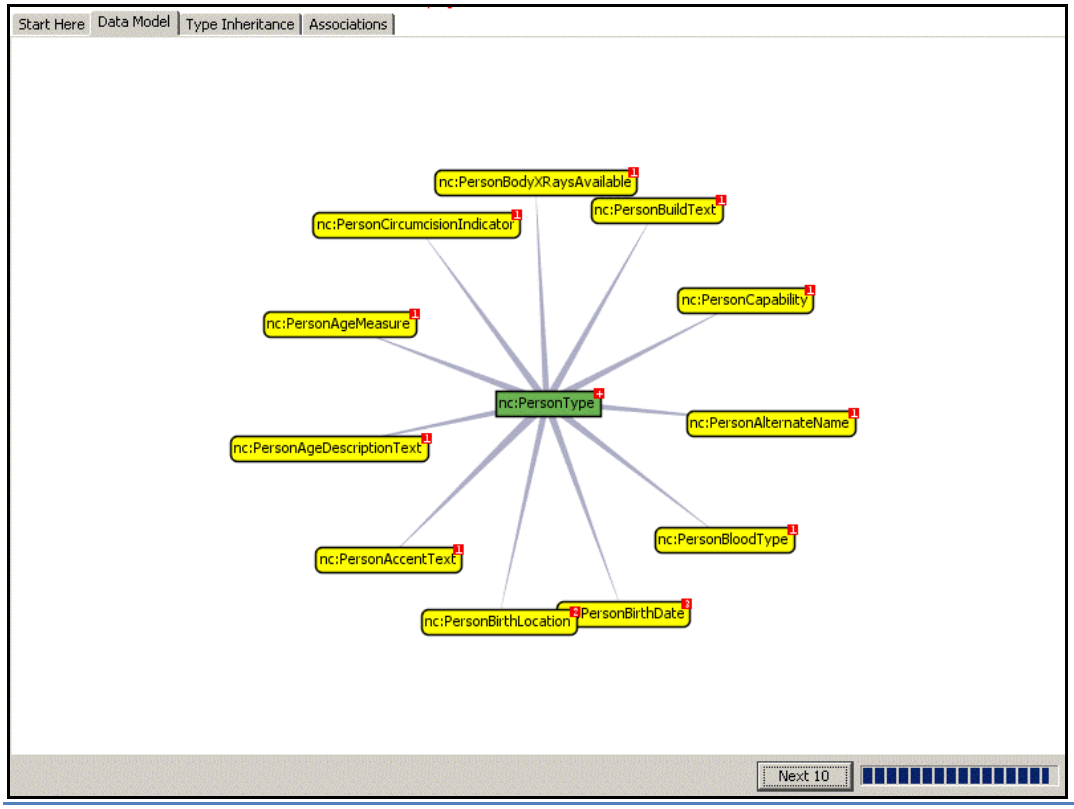


Figure 59: Data Model Browser.

While the Data Model Browser will not help create your schemas, it is a useful tool for gaining a deeper understanding of the model, visualizing the model hierarchy, and identifying dependencies between data objects.

Alternate Model Formats

While the NIEM schemas are considered the authoritative version of the model, unless you are a hard-core coder, you will probably find it easier to search the model using one of several alternative formats available for downloading on NIEM.gov. Go to the NIEM.gov home page and click the **downloads** link to go to the **Downloads** page. On that page you will find the link **Other database formats**. Clicking that link will allow you to download and save a .zip file containing several Excel spreadsheets and an MS Access database containing the full NIEM model. These files are simple to use and are handy references while you are going through the process of mapping your exchange elements to NIEM.

NIEM Wayfarer

Like the JIEM Tool, NIEM Wayfarer is a non-NIEM.gov application. It was developed by the National Center for State Courts (NCSC) as a lightweight alternative to the SSGT (described below) for exploring the NIEM model; its elements, attributes, and data types; and the relationships between them.

Searching

Searching is the main entry point into NIEM Wayfarer. You begin by entering one or more terms into the search box and viewing the search results. The search results page presents elements, types, and code table entries that match the search terms entered. Results can be narrowed or broadened by changing the searching options (detailed below). The results page shows matching elements in the left-hand column, matching types in the middle column, and matching code table entries in the right-hand column. With most browsers, placing the mouse pointer over the hyperlinked text will result in a pop-up containing the definition of the element or type.

Search Options

Searches can be narrowed or broadened by changing the search options. The default is to search both names and definitions. The full range of options is described below.

- **Search both names and definitions.** This option searches both names and definitions for matches. It provides a good balance between getting too many results and getting too few.
- **Search names, definitions, and more.** This option includes additional search fields, such as keywords (synonyms for NIEM terms), examples, and other additional usage information. Including these fields may yield too many results to be useful. On the other hand, as there is no "police officer" in the NIEM data model, it is currently the only way to search on "police officer" and have its semantic equivalent "j:EnforcementOfficial" returned as a result.
- **Search names only.** This option searches element and type names only. This is best used when searching on a common term that will appear in many definitions.
- **Search definitions only.** This option searches element and type definitions. It is used primarily to filter out names containing common terms.
- **Search exact name match.** This option searches for an exact match with an element or type name. For example, searching on "person" will result in dozens of hits. To quickly get to nc:Person or nc:PersonType, do an exact name search on "person."

Contextual Search

Contextual searching is an indirect search that takes element inheritance into account. This capability is best explained by example. For instance, if we wanted to include an "arrest date" element in our exchange, and we did a standard name and definition search, Wayfarer would not return any matches because there is no ArrestDate element in NIEM. However, the concept of an "arrest date" can be derived in NIEM because it is represented through inheritance. The

j:Arrest object is of type j:ArrestType and j:ArrestType is derived from nc:ActivityType. Since nc:ActivityType contains the abstract element nc:ActivityDateRepresentation and nc:ActivityDate can be substituted for it, j:ArrestType can, through inheritance, contain an nc:ActivityDate. In NIEM Wayfarer, a contextual search will make this connection and display the following suggested solution:

ArrestDateRepresentation: A date an Arrest occurs.

Represented by nc:ActivityDateRepresentation in the context of the j:Arrest property.

It is interesting to note that, in the example output above, "ArrestDateRepresentation" is not actually part of NIEM but a suggested solution by the tool as to how to handle "arrest date" in an extension schema.

Search Results

Clicking on one of the resulting hits will take you to a page that provides more information about the data element or data type, including the element name and definition, keywords, example content, additional usage information, and information about the namespace in which the element resides. Clicking the link **Graphical View** will display the data object and its hierarchy in a graphical format. You will need the Adobe SVG Plug-in installed on your computer to use this feature. A link to download the plug-in is displayed along with the link to the Graphical View page. (See Figure 60.) For more information on NIEM Wayfarer and its capabilities, and to access the tool, go to http://www.ncsconline.org/niemwayfarer/.

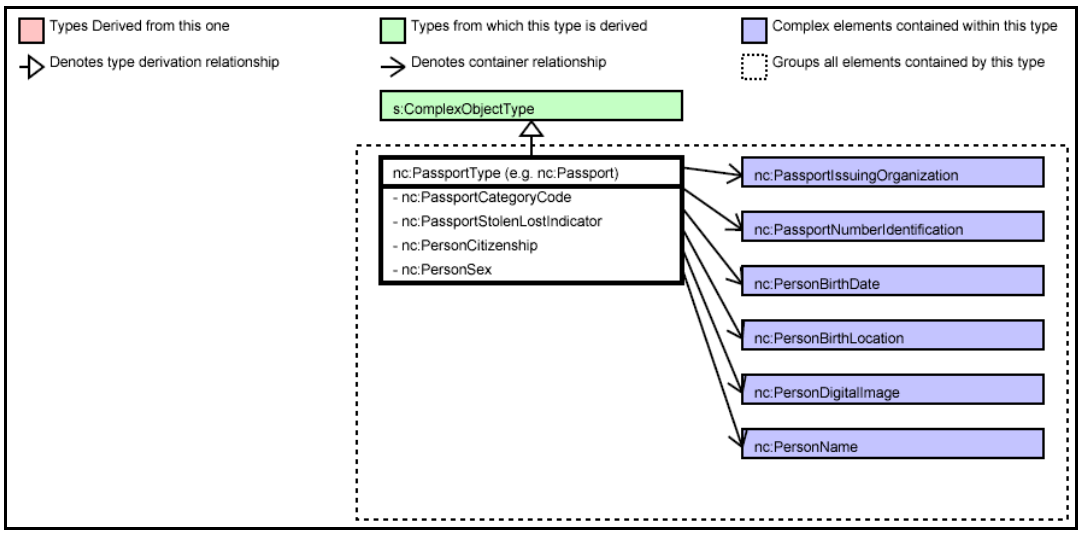


Figure 60: NIEM Wayfarer Tool With Example Graphical Output.

Subset Schema Generation Tool (SSGT)

The SSGT has a strong set of search features to help you map your exchange data elements to the NIEM model and, once you have identified the NIEM elements you need, the schema subset features will help you build a custom subset of the NIEM schemas to suit your application.

To access the Subset Schema Generation Tool from the NIEM Tools page, you can either roll over **Search/Navigate Model** and then select **Search/Navigate Model Textually** or roll over **Build Schema Subset** and then select **Build Schema Subset.** (See Figure 61.)

SSGT Search function

The SSGT search function allows you to:

- Enter search terms and view matching results in a data hierarchy format.
- Navigate through the various data types, properties, and facets and their relationships.
- Select advanced search options to refine your search results.

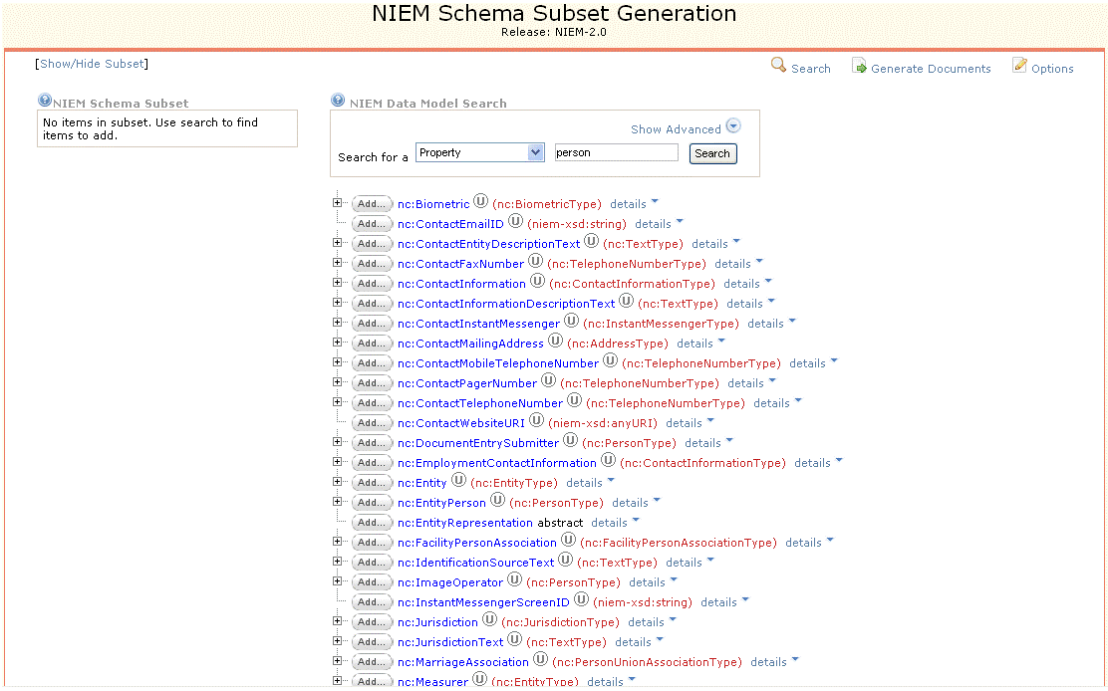


Figure 61: The SSGT Is the Primary Entry Point Into Building NIEM Subset Schemas.

Standard Search Options

The default search option "Property" will search for individual data elements within the NIEM model. You can change the search parameters by clicking the "Search for a" drop-down and selecting either "Property," "Type," "Namespace," "Facet," "External," or "Association." In NIEM, a property would normally hold the actual data in an XML instance. A "Type" is a class of data that contains any number of properties normally associated with its data class. For example, nc:BiometricType contains properties, such as nc:BiometricImage, nc:BiometricEncodingMethodText, etc.

A "Namespace" is a logical grouping of data types, properties, and facets associated with particular domain. If you select "Namespace" in the search drop-down, the application will search the namespace prefixes, fully qualified namespace URIs, and namespace descriptions for

the matching search term. It will return the namespace or hyperlinked list of namespaces that contain the matching term.

In XML terms, a "Facet" is a code list value. For instance, if you have a code list for automobile manufacturers, "Volvo" would likely be a facet in that list. The SSGT will search all the code lists within NIEM for the matching facet term either as its code list value or definition. It will return all matching code list schemas, data types, code list values, and definitions.

Selecting the search parameter "External" will search for properties and data types within external namespaces and adapter types for the search term. Selecting "Association" will limit the search to those properties and types that include the word "Association" in the name or definition. See Section 5.5 for a detailed discussion about associations within NIEM.

Advanced Search Options

To open or close the advanced search options, click the toggle link **Show Advanced**/**Hide Advanced** (or optionally, the arrow next to the link) on the search page. There are a number of user-selectable options to help refine your search. For multiterm searches, you can limit the search to the exact phrase by entering your search terms in the textbox next to "with the exact phrase." A logical OR search would be conducted by placing your search terms in the textbox next to the phrase "with at least one of the words." A logical NOT search would be conducted by placing your search terms in the textbox next to the phrase "without the words" and would exclude those terms in the results.

NOTE: This last option is not as useful as the other textbox options since it is possible to enter a term that is not included in any property or data type, resulting in a display of the entire model. Selecting or unselecting the checkboxes next to Names, Definitions, Keywords, Usage, and Example Content will either target or limit your search for terms included in the selected fields.

To clear all the textboxes, click the **Clear search terms** link. To clear all checkboxes, click the **Clear All Checkboxes** link. To reset all of the advanced search options to their default state (on), click the **Reset All Checkboxes to Defaults** link. Both the clear checkboxes and reset checkboxes links can be found at the bottom of the "Search Preferences" section of the page. You can also limit searches to specific domains by either checking or unchecking the checkbox next to the domain name in the "Search Preferences" section. (See Figure 62.)

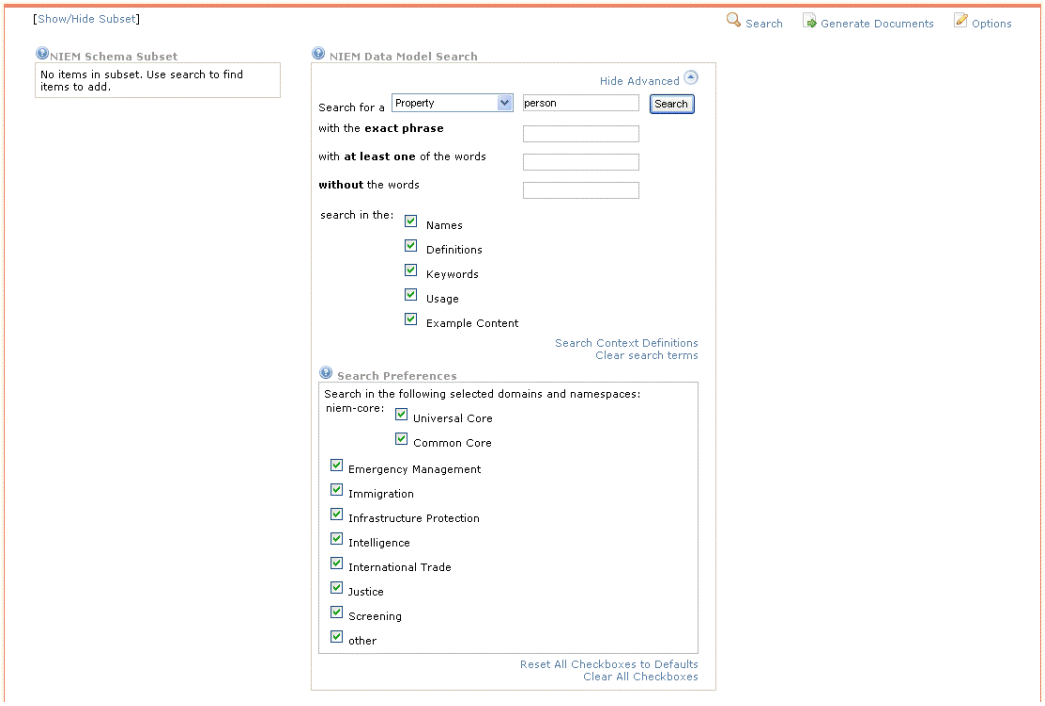


Figure 62: The SSGT Includes a Number of User-Selectable Options to Help Refine Your Search.

Search Context Definitions

Another really useful search capability is the context search function. Some derived data types inherit properties from their parent data types, which allows for better reuse of elements; however, it can also make searching for properties more difficult. Similar in function to the NIEM Wayfarer Tool, the Subset Schema Generation Tool has a context search capability that makes it easier to search for derived properties because these kinds of indirectly related properties can be found only through a context definition search. For example, a standard search on "BailStatus" will not return any results; however, a context definition search of "BailStatus" will indicate that "Bail inherits ActivityStatus from ActivityType."

NOTE: Context definition searches work only with multiple terms in the search box. You can enter the search terms in the textbox with spaces between the words or you can type a single word as UpperCamelCase or lowerCamelCase—either way, the application recognizes these variations as multiple terms.

Therefore, in our example above, if you put "bailstatus" in the search term textbox, the application will return "No search results found." However, if you put "bailStatus," "bail status," or "BailStatus" in the textbox, the application will still return "No search results found" although you will now be provided with the additional link **Search Context Definitions**. Clicking the link will run the search in context definition mode and will return a number of potential matches for "bail status."

Exploring the Model

The properties and types displayed in the search results are hyperlinked to display more detailed information about that component. This helps users navigate through the NIEM data model to find associated properties and to explore the model hierarchy. Search results are displayed as trees and can be expanded by clicking the ⊞ icon next to a result. Expanding a node in the tree will display all of the properties contained in a type. In an expanded node, clicking the link **show inheritance** will show the parent type of the current type. Clicking on the hyperlinked property or type name will display the details page for that property or type.

Property Details Page

The property details page will display the definition of the property along with any keywords, usage information, or example text that may be available. The details page will also display the property type (parent) and any other types that contain that property. In addition, if the property is an abstract element, it will display all the properties that are substitutable for that property. For more information on abstract elements, see Section 5.7.

Type Details Page

The type details page will display the definition of the type along with its content style ("Complex with Complex Content," "Complex with Simple Content," or "Simple"). This page will also display the properties contained within the type and other properties that are of the type. If the type has any base types that it inherits from or derived types, this information will also be displayed.

Selecting Properties and Types

The second major functionality of the tool is that it allows you to select properties and types to add to your subset. As you search and explore the model, you will identify elements you will need to use in your exchange. By clicking the Add... icon next to the type or property name, you select the item and add it to your subset schema list. This feature is explored in more detail in the "Build Schema Subset" section below.

Map Information Exchange

The **Map Information Exchange** is a tool introduced with NIEM 2.0. It provides a new entry point for mapping your exchange requirements to the NIEM model. As mentioned in the introduction above, the starting point for the exchange mapping process in NIEM 1.0 was the Component Mapping Template (CMT), a spreadsheet for aligning the data element terms from your data set to the equivalent NIEM XML terms. Although the CMT is still a viable option and relatively easy way to map your exchange components to NIEM (see *Component Mapping Template (CMT)* beginning on page 1057), the new Map Information Exchange Tool is integrated with both the SSGT and IEPD tools and adds additional capabilities that make the IEPD development process much easier.

You will use the Map Information Exchange Tool to create an exchange, associate an exchange model with the exchange, map the data objects within the exchange model to the equivalent NIEM data elements, and generate IEPD artifacts, such as mapping reports, wantlists,

and schemas. To access the Map Information Exchange Tool from the NIEM Tools page, roll over and select **Map Information Exchange** from the list of tools.

Creating an Exchange

If you have registered on NIEM.gov, you should log in now before you create your exchange. If you do not have an account, you can still create exchanges, store the exchange models you upload (temporarily, at least), and map your exchange elements to NIEM. However, once you exit the Web site or leave the Web site inactive for two hours, all the work you have done will be lost unless you first download the files to your computer. Registering on NIEM.gov is free, quick, and easy, and it provides you with storage space on the server to permanently save your IEPD artifacts.

To use the Map Information Exchange Tool, you must have a UML model of your exchange (see *Universal Modeling Language (UML)* Tools, beginning on page 90). You begin the mapping process by creating an exchange. Click **create an exchange** to add a new exchange to your list of exchanges. (See Figure 63.)

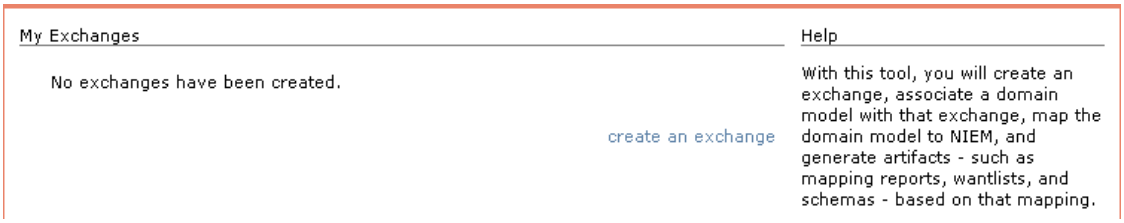


Figure 63: Create an Exchange.

Highlight the name of the exchange text in the box to change the default name "Exchange created on [DATE] at [TIME]" to something more meaningful to your project. (See Figure 64.)

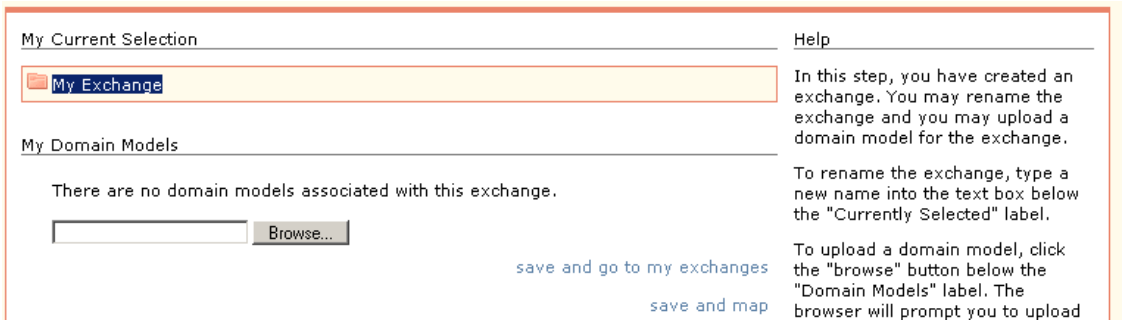


Figure 64: Change the Default Exchange Name.

Next, click the **Browse…** button to select a UML file in .xmi format for your exchange model. The tool will upload and attempt to parse the file. If the parse is successful, the name of the exchange model will appear below "My Domain Models." (See Figure 65.)

NOTE: Currently, the tool is limited to importing UML models in XMI version 1.0 or XMI 1.2 format. If you are looking to purchase a UML modeler that is compatible with the NIEM tools, a tool that exports an XMI 1.0 or 1.2 representation of a UML 1.4 metamodel should be considered a primary consideration in your tool choice.

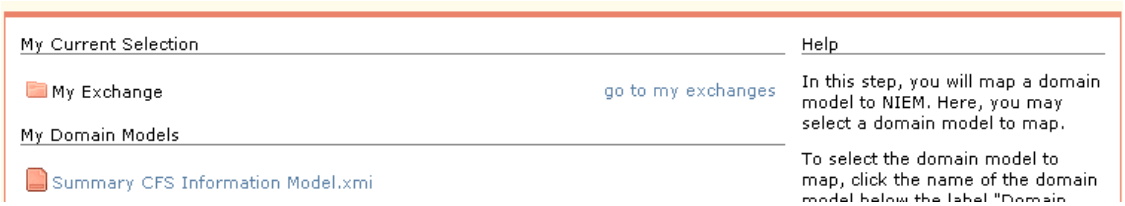


Figure 65: Uploading Data Model to Exchange.

Once you have uploaded the data model for your exchange, click the hyperlinked model name. The tool will extract the class diagram from the file and display the data elements under the heading "My Data Elements" organized according to the class hierarchy of the model. If a data class has elements associated with it, a small ⊞ symbol will appear to the left of the class name. Clicking the ⊞ symbol will expand the data class and show each of the elements associated with the class (see Figure 66).

Very large UML models could overwhelm the tool, causing the processing speed to slow down considerably. It is best to break up large UML models into a number of smaller, more manageable parts. A reasonably sized model would include around 30–50 elements.

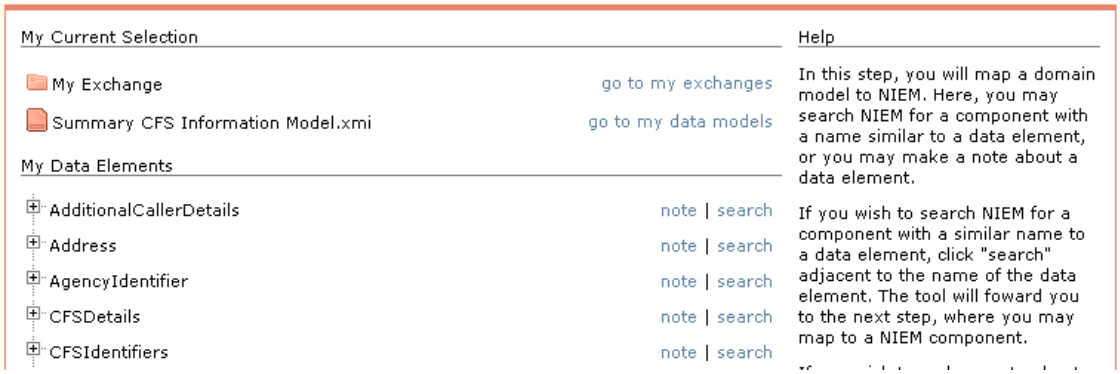


Figure 66: Process to Begin Mapping Exchange Elements to NIEM.

Mapping an Exchange

You may now begin the process of mapping your exchange data elements to NIEM. If your exchange element has a ⊞ symbol to the left of the element name, clicking the symbol will expand that data class and display all the elements within it. Click the **search** link to the right of the exchange element to search the NIEM model for elements with similar names to your element. (See Figure 67.)

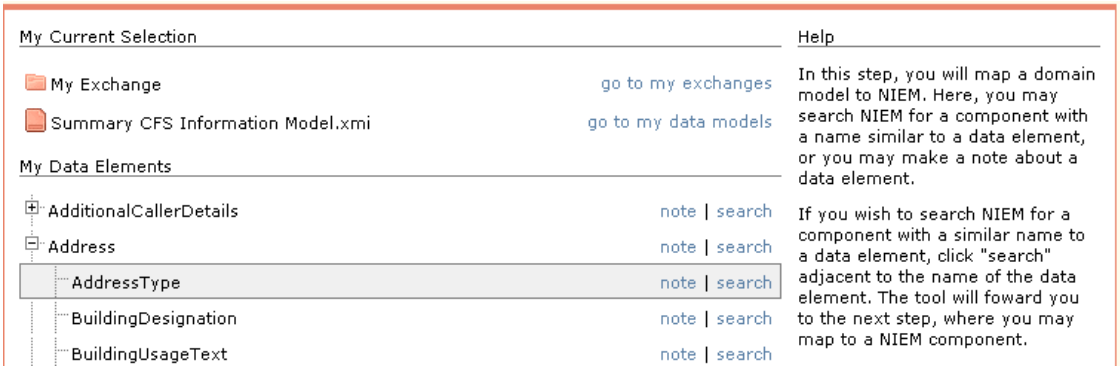


Figure 67: Find Equivalent NIEM Terms.

The search return will list all the NIEM elements that are potential matches for your exchange element. To display more information about a specific component in the search return list, click **show** beside the name of the NIEM component. The tool displays the component definition and other information related to the properties and data types that contain that component. To close the information window, click **hide**.

> In the search return, some components may include one or more star symbols by the element name. The number of stars signifies the degree of confidence the application calculates that the related NIEM element is a good match for your exchange element. (See Figure 68.)

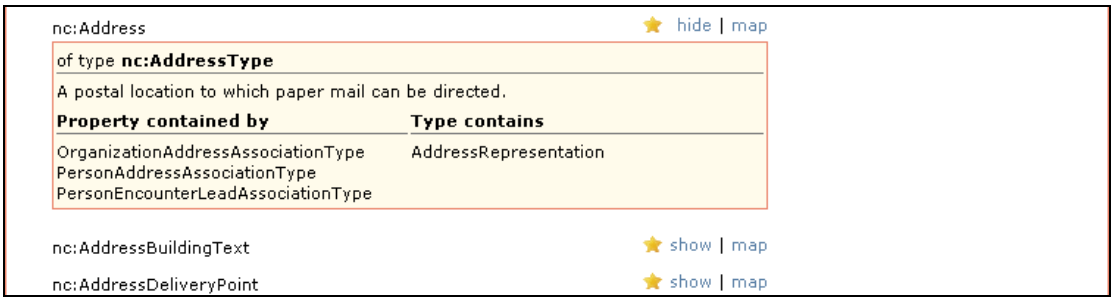


Figure 68: More Information Regarding Selected Components.

If a search fails to return a suitable match for your exchange element, you can perform a search based on one or more terms you select by typing your search term(s) in the "Search for more NIEM Components" textbox at the bottom of the search return screen, then clicking **search**. (See Figure 69.)

Figure 69: Search for Additional Matches.

When you have identified a good match within NIEM for your exchange element, map the selected component to your exchange element by clicking **map** beside the name of the NIEM component.

- You may include a note about the data element, if you wish, by typing your comment in the "Mapping Notes" field.
- If you wish to identify the degree of matching between your exchange element and the selected NIEM component, click one of the radio buttons under the "Mapping Category" heading. Both the mapping notes and the mapping category information are included in the mapping report you will generate later.
- Select "Equivalent" if the semantics and structure map appropriately. The NIEM element name and definition do not have to be the same as your exchange element name and definition, but they should have the same semantic meaning.
- Select "Partial Match" if the data element definition somewhat matches the NIEM term. Although there may be some degree of disparity between your exchange element and the matching NIEM element you select, there should be no semantic mismatch or structural conflict.
- Finally, click **save** to save your mapping choice and metadata. (See Figure 70.)

The options "Semantic Conflict," "Structural Conflict," and "No Match" should not be used.

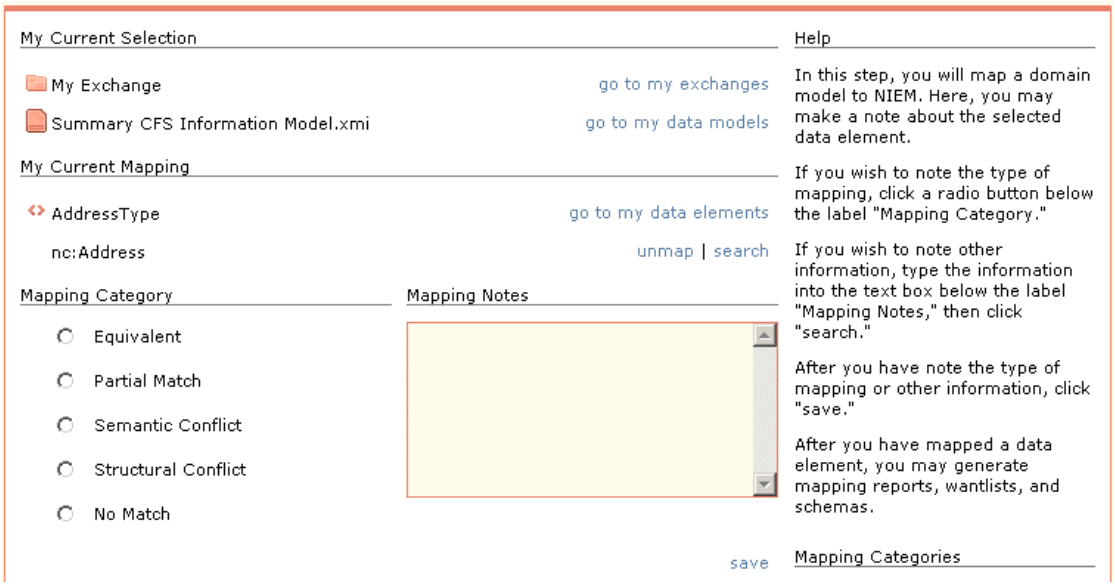


Figure 70: Comments and Mapping Categories.

Continue the above process with the remaining elements—searching, selecting, and mapping, as appropriate.

> After you complete the mapping process, any remaining exchange elements that have no equivalent in the NIEM model will be included in the extension schema when you generate your exchange artifacts (as explained below).

Once you have mapped your exchange elements to the equivalent NIEM components, you may generate mapping reports, wantlists, and schemas.

- Click **go to my exchanges** to return to the top-level list of your exchanges.
- You have several options at this point:
  - You can continue mapping additional exchange elements to the equivalent NIEM terms by clicking **map** and following the process described above.
  - You can permanently delete the exchange from your list of exchanges by clicking **delete.**
  - You can generate reports, wantlists, and schemas based on your exchange by clicking **artifact** next to the exchange you wish to use. Generating artifacts is described below.

Generating Artifacts

The **Artifacts** page provides you with several options for generating and downloading your exchange artifacts.

- To generate a mapping report in Excel format, click **generate a mapping report**.

- To generate a mapping set (the mapping report in XML format), click **generate a mapping set**.
- The current state of the mapping process can be saved to a "wantlist" file and used as the starting point for further refinement or editing at a later point in time.
- To generate a wantlist, click **generate a wantlist**.
- To generate NIEM subset schemas, click **generate a subset schema**.
- To generate a constraint schema, click **generate a constraint schema**.
- To generate both an exchange and extension schema,[38] click **generate an exchange schema.** (See Figure 71.)

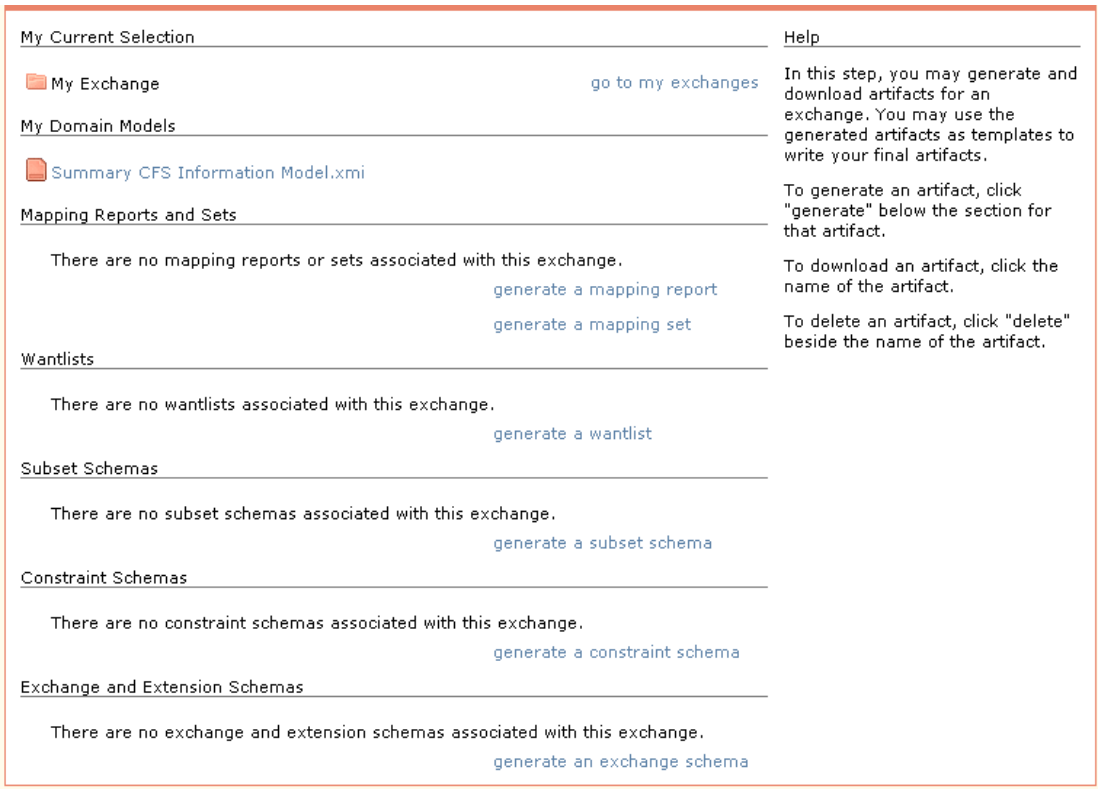


Figure 71: Generate Artifacts.

There is no limit to the number of times you can generate a given artifact.

To save multiple versions of an artifact, return to the Generate Artifacts page whenever you need to create an additional version of an artifact. Once you have generated your artifacts, you can download and save or open your artifacts on your computer by clicking the hyperlinked

[38] Although the tool will generate exchange, extension, and constraint schemas for you, these files will need to be modified to better fit your needs. The schema creation process is not completely automated, but the generated files provide a good starting point for additional refinement.

artifact name. You may also delete your artifact at any time by clicking **delete** next to the artifact name. To include your artifacts in an IEPD, see the section *Working With IEPDs* beginning on page 109.

Component Mapping Template (CMT)

A Component Mapping Template (CMT) has been developed to facilitate the mapping of your exchange elements to the equivalent NIEM terms and for identifying mapping gaps that form the basis of your extension schema. The template is an Excel spreadsheet and provides a convenient format for capturing the results of the mapping process. The template can be used as is or modified to meet specific mapping needs.

The NIEM component mapping process involves identifying and characterizing gaps at the entity (class), element (database attribute), and value (literal) levels. Component mapping categorizes data-source components at each level as matching (equivalent), partially matching, or not matching any component within NIEM. Matching components include those in which the component names may differ but in which the components themselves are semantically and structurally equivalent. Partial matches can arise when there are similarities but also some differences between components. These differences can include semantic and/or structural mismatches, naming collisions, and mismatches at the value set, datatype, and/or lexical levels. Exchange data elements with no matching NIEM term comprise a set of additional entities and element types to be included in the extension schema. In addition, these elements may be evaluated by the appropriate NIEM governance bodies for inclusion in an update to the NIEM model.

The CMT is self-explanatory and has column definitions, instructions, and examples to facilitate the mapping process. The CMT can be downloaded from the NIEM.gov.

Building Schema Subset

The NIEM Data Model consists of thousands of data types, properties, and code lists throughout more than 60 namespaces. Typically, only a very small fraction of the content of the model is ever used in any given exchange, which is why NIEM provides tools to facilitate the creation of NIEM-conformant schema subsets based. In addition to the search capability described above, the SSGT allows you to select the data components from which your schema subset will be built.

Although the SSGT is really two tools in one—a search tool and a selection tool—in building a schema subset, the search and selection process usually happens in an iterative fashion. First, you search for the NIEM component to map to your exchange element, and then you select the component to be added to your schema subset. This section will discuss the selection and subset schema-building features of the SSGT.

The SSGT has a number of features that help make the subset schema-creation process relatively easy. You can use the tool to:

- Download the entire dataset of a selected domain.
- Build a schema subset containing only the components required for a particular exchange.
- Download a wantlist of selected components for later use.

- Download a schema subset based on a wantlist of selected components.
- Upload a wantlist as a starting point for further refining an exchange specification.
- Select the NIEM version used to build a schema subset (NIEM 1.0 or 2.0).
- Transfer your completed schema subset directly to the IEPD tool (discussed below) to build an IEPD from your schema artifacts.

To access the SSGT from the NIEM Tools page, you can either roll over **Search/Navigate Model** and then select **Search/Navigate Model Textually** or roll over **Build Schema Subset** and select **Build Schema Subset**.

Add Components to Schema Subset

As you search for the components you need for your exchange, select those to be included in your subset by clicking the Add... button next to the property or type name. A component will be added to your subset list in the context in which it is displayed. For instance, if a property is displayed within a type, clicking the Add... button will add it to your subset list within that type. To add a property as a top-level component, either click the Add... button next to the property when it is the top level in the tree, or click the hyperlinked property name to open the details page for that property and click the Add... button from there. To add a property in a type, click the Add... button where the property is displayed in that type.

When components are selected, they are displayed within the "NIEM Schema Subset" section on the left side of the page. The components you explicitly selected are highlighted in bold. To ensure that the resulting schema subset will validate correctly, the application automatically adds all component dependencies to your selection. For example, if you add the component ActivityDescriptionText to your subset list, the SSGT will also include nc:TextType, niem-xsd:string, and xsd:string as dependencies of ActivityDescriptionText. (See Figure 72.)

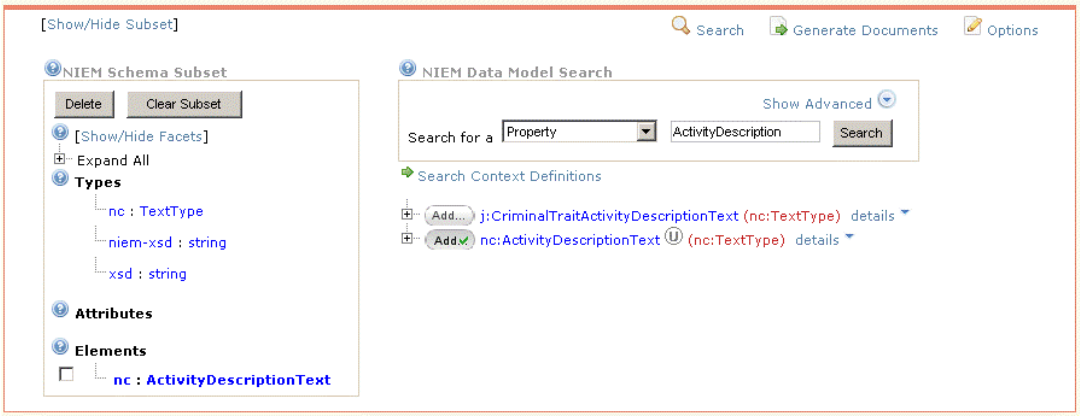


Figure 72: Select Component for Subset.

The components you explicitly select will include a checkbox to the left of the component name that allows you to delete the component if you decide you no longer need it for your exchange. As long as your subset list includes at least one explicitly selected component, the component dependencies of those components will also remain on your subset list. To delete an explicitly selected component, select the checkbox next to the component name and click the

Delete button at the top of the NIEM Schema Subset section. To clear all selected components and start over, click the **Clear Subset** button at the top of the section.

Generating Documents Page

Save a Wantlist

Once you have added components to your subset list, the user requirements can be saved in a wantlist file for later use. To save a wantlist, click **Generate Documents** from the search page and then click **Save current wantlist to a file** in the Generate Wantlist section to open or save the wantlist file to your computer. (See Figure 73.)

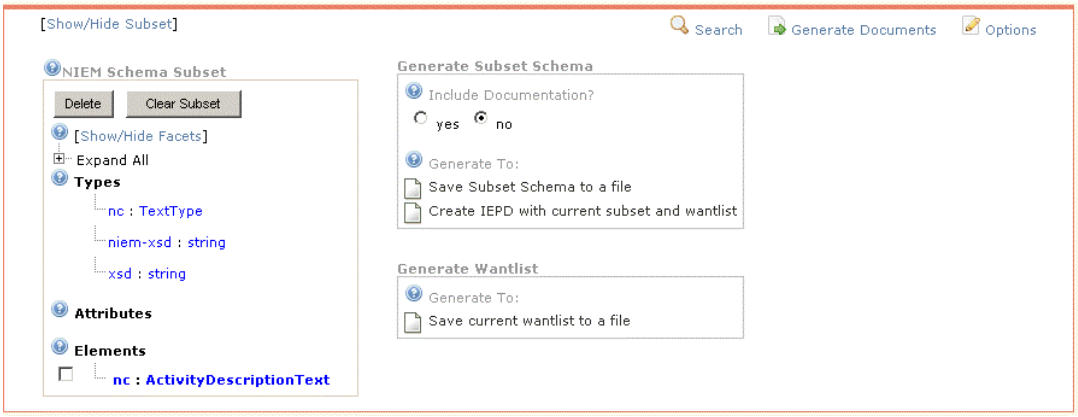


Figure 73: Generate Documents Page.

Generate Subset Schema

To generate a schema subset from the list of selected components, first choose whether to include the component definitions within the schema annotation.

- Select **yes** to include documentation or **no** to omit it.
- Finally, click **Save Schema Subset to a file** to open or save the .zip archive containing the schema files on your computer.
- Alternately, to transfer your work directly to the **IEPD Tool**, click **Create IEPD with current subset and wantlist** to open the IEPD tool application with your current subset and wantlist. You must be registered and logged in to use the IEPD tool.
- If you have not already registered, see the section "Registering on NIEM.gov" above. If you have registered but are not logged in, you will be asked to provide your username (i.e., your e-mail address) and password to log in.
- Once you are logged in, the tool will take you through a series of steps that will enable you to build an IEPD.

SSGT Options Page

Within the SSGT, the Options page allows you to:

- Select the NIEM version used to create your schema subset.
- Load a wantlist file from your computer.
- Download the NIEM Wantlist Schema Specification.
- Download a schema subset based on an entire NIEM domain namespace.

Change Release

You can choose which NIEM version (1.0 or 2.0) will be used to create your subset schema by selecting the release in the "Change release to" drop-down list in the Change Release section.

> If you wish to save your current component selection to a wantlist, do so before changing releases. The selection list is cleared when the release is changed. (See Figure 74.)

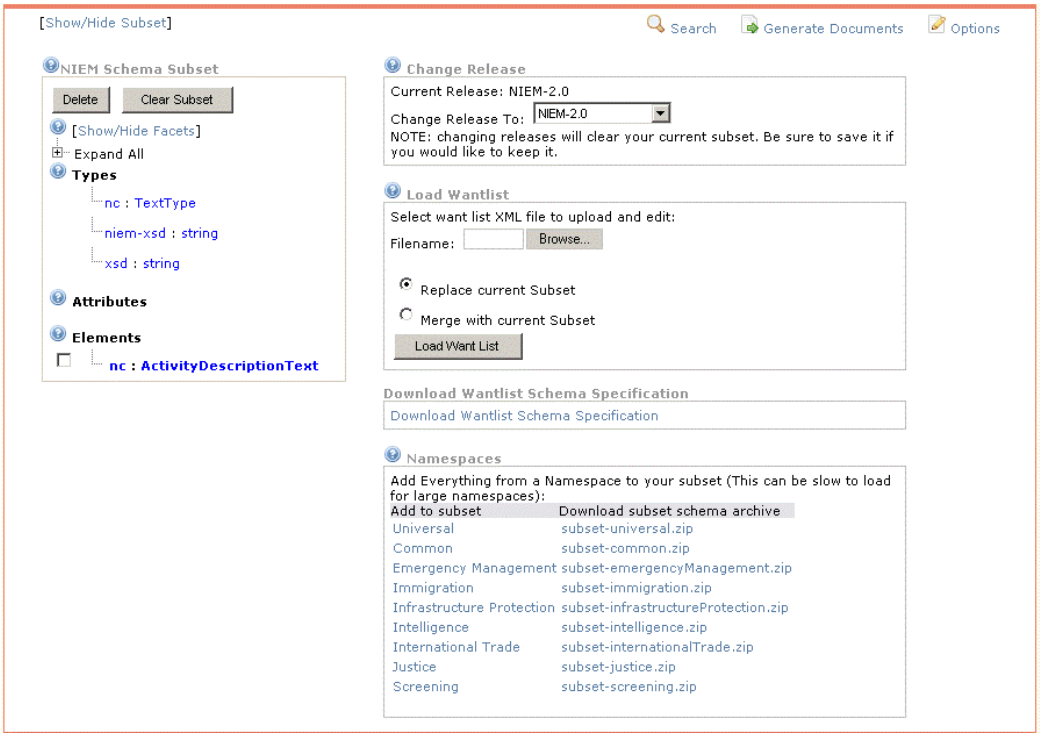


Figure 74: Load a Wantlist or Change Release Versions on the SSGT Options Page.

Load Wantlist

To import a wantlist as a starting point for further refining your subset, click the **Browse…** button in the Load Wantlist section and then select the file from your computer. To clear any currently selected components as you load the new wantlist, choose the option **Replace current Subset**. To continue with the currently selected components and merge the new wantlist components to your existing subset, choose the **Merge with current Subset** option. Finally, click the **Load Want List** button to upload the file and update your selected subset in the NIEM Schema Subset box.

Download Wantlist Schema Specification

The SSGT automatically keeps track of the list of components you select in your subset. As explained above, you can retain this selection list for later use by downloading and saving the wantlist to your computer. At a later point in time, you can upload the wantlist in the SSGT and continue building your subset. An alternative method of building your subset is to construct your wantlist manually and then upload the wantlist file to the SSGT. The Wantlist Schema Specification is the set of "instructions" that allows you to build a valid wantlist without the repetitive search and select process of the SSGT. This approach is useful if you are a "NIEM XML Power User" and you need a quicker and more efficient way of building subsets. To download a copy of the Wantlist Schema Specifications, click **Download Wantlist Schema Specification** in the Download Wantlist Schema Specification section to save the schema file to your computer.

You can also download this file from the **Documentation** page of the Tools Web site.[39]

Namespaces

You can download a NIEM schema subset that includes all the components of a selected domain namespace.[40] By clicking Universal, Common, Emergency Management, Immigration, Infrastructure Protection, Intelligence, International Trade, Justice, or Screening, you can open or save the associated .zip archive file on your computer for use in your exchange.

XML Development Tools

A complete IEPD includes several schemas, including an exchange schema, an extension schema, a constraint schema, and a number of NIEM subset schemas. Except for the NIEM subset schemas that can be developed using the SSGT, all schema development is done outside the NIEM.gov Tools site using either freely or commercially available XML development tools. As with the section on UML tools above, a full discussion of XML tools and resources is beyond the scope of this guide, but a good starting point for your research would include a search for "XML tools" using your favorite search engine.[41]

Working With IEPDs

The final activity in the IEPD lifecycle is packaging your schemas and other documentation into a .zip archive and publishing your work to the IEPD repository. As mentioned in the introduction above, there are several starting points for building your exchange specifications, including the JIEM Tool, the Component Mapping Template, and the Migration Assistance Tool. Regardless of your starting point, each path leads to the same end—the ***Work With IEPDs*** tool described here. To access the Work With IEPDs tool from the NIEM Tools page, roll over and select Work With IEPDs from the list of tools on the left. Using this tool, you can search existing

[39] http://niem.gtri.gatech.edu/niemtools/resources/schemas/wantlist/niem-1.xsd.

[40] http://niem.gtri.gatech.edu/niemtools/ssgt/help.iepd#download_domains.

[41] A list of commercially available XML editing tools can be found at http://www.google.com/Top/Computers/Data_Formats/Markup_Languages/XML/Tools/Editors/.

IEPDs on the repository, create a new IEPD, modify your existing IEPDs, or upload an IEPD. Each function is described in more detail below. (See Figure 75.)

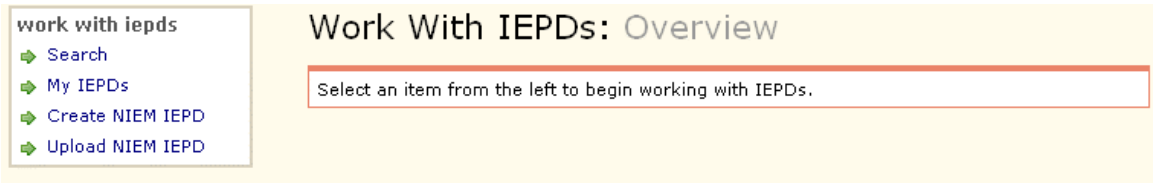


Figure 75: Search, Edit, Create, or Upload an IEPD From the Overview Screen.

Searching the IEPD Repository

This screen allows you to locate "shared" IEPDs on the NIEM.gov repository (instructions on setting the sharing attribute on your IEPDs are described below). A link is also provided to take you to the IEPD repository search page on http://it.ojp.gov. View the listing of shared IEPDs or enter a term in the search field to narrow your search for IEPDs containing the specific search term in the name or description.

My IEPDs

This function allows you to edit your existing IEPDs.

- Click **My IEPDs** to view the list of IEPDs in your account and then click the hyperlinked IEPD name to view the details of that IEPD. (See Figure 76.)

Figure 76: Review and Edit Artifacts.

- On the details page, click **Download** to download and save, or open the .zip archive on your computer.
- Click **New Version** to create a new IEPD using a copy of your IEPD artifacts and metadata as a starting point for changes and updates.
- Click **Edit** to go to the **IEPD Edit Options** page.
- On this page, click **Edit Metadata and Artifacts** to go back through the previous artifacts and metadata pages to make changes and updates to your IEPD.
- Click **Delete** to go to the **Delete IEPD** page, where you can download and save or open the IEPD on your computer and verify that you wish to delete the selected IEPD.
- Click **Register** for information about sharing and registering your IEPD on the OJP IEPD Clearinghouse.

- Click **Edit Visibility/Sharing** to go to the **Edit Artifact Visibility** page, where you can change the sharing/visibility attribute from the default **Not Shared** to **Shared** so that other NIEM.gov users will be able to see and access the IEPD you have created.
- Click the **Update Visibility** button to commit your change to the sharing attribute.

Creating a NIEM IEPD

This option will take you through the steps of uploading artifacts and adding metadata to your IEPD. The meaning and use of each of the files and pieces of metadata are described below.

Upload Artifacts Page

Exchange Files

- **Subset Schema:** Subset of the full NIEM schema—a compressed directory of schemas (to distinguish from other schema sets).
- **Wantlist:** User requirements (distinguishes user data components required by the user from components that the user depends on for conformance);—generated by and uploaded to the Schema Subset Generator Tool (SSGT); this is an open spec; the SSGT is not required to create a wantlist (though it is easier).
- **Exchange Schema:** Base document schema that defines the XML root element, generally named after the IEPD itself—also known as the document schema, reference schema, and root schema.
- **Constraint Schema:** Constraints for separate constraint validation path—a compressed directory of schemas (to distinguish from other schema sets).
- **Extension Schema:** Specification for extended components—separate local namespace; components not contained in NIEM.
- **Sample Style Sheet:** Example style sheet for display of instances—may include several files.
- **Sample XML Instance:** Example instance—may be multiple; may reference optional style sheet.

Master Documentation Files

- **Main Master Documentation:** May include purpose, business requirements, what, when, why, how to, etc.
- **Business Requirements:** Itemized descriptions—may also contain business rules.
- **Memos of Understanding:** Memoranda of understanding among participating agencies.
- **Endorsement Letters:** Documentation from professional or governmental organizations that confirm support—refer to Endorsement in metadata.

- **Methodology and Tools:** Used to build IEPD—may contain URLs or references to tools, methodology, documentation.
- **Testing and Conformance:** Description and results of validation and conformance testing performed—may include testing output or products.

Other Documentation

- **Domain Model:** Domain model in standard open format (xmi, vsd, zargo) and standard open graphic (jpg, pdf, etc.)—likely a Unified Modeling Language (UML) model.
- **Use Case Model:** Use case diagram in standard open format and standard graphic—likely UML.
- **Business Rules:** May be (1) plain or structured English, (2) written into master documentation, (3) Schematron or other formal business rule language, or (4) generated by a development tool.
- **Mapping to NIEM:** Mapping of domain components to NIEM components—tagged with constraints (cardinality, etc.); prefer Component Mapping Tool (CMT).
- **Extended Components:** Components created because they were not in NIEM—may be part of mapping spreadsheet; include structure and definitions of new components; prefer Component Mapping Tool (CMT).
- **Change Log:** Record of cumulative changes from previous IEPD versions—initial IEPD simply records its creation date.

Click the "Next" button to enter metadata on the next page.

Enter Metadata Page

IEPD Metadata

- **IEPD Name:** Title of this IEPD (e.g., Amber Alert, Prosecutor Arrest Warrant).
- **Short Summary:** Brief summary of this IEPD for short display purposes.
- **Detailed Description:** Narrative description of this IEPD—may contain as much detail as you think useful to those with a potential interest in this IEPD.
- **Creation Date:** Year and month (YYYY-MM) that planning or work on this IEPD started (do NOT confuse with date you submitted this IEPD information). Click the  icon to pop up an interactive calendar to select the date.
- **Last Revision Date:** Year and month (YYYY-MM) this IEPD information was last revised (do NOT confuse with the date the IEPD itself was last revised). Click the  icon to pop up an interactive calendar to select the date.
- **Next Revision Date:** Year and month (YYYY-MM) this IEPD information is expected to be revised. Click the  icon to pop up an interactive calendar to select the date.
- **Security:** Security label to indicate treatment/distribution of this IEPD (default is public, unless otherwise noted). Select either "Public," "FOUO," "Classified," or "SBU."

- **NIEM Version:** NIEM version used for this IEPD. Select either "1.0" or "2.0."
- **Maturity:** State of development. Select either "1" Entry level; under development; minimum documentation; "2" Complete; being tested; in limited use; draft documentation; or "3" In production; fully documented; endorsed for use in official exchanges.
- **Version of This IEPD Version:** Description or additional information related to current state of this IEPD.
- **Schedule:** Information about the development schedule for this IEPD (e.g., Development started (YYYY-MM); draft planned (YYYY-MM); completion planned YYYY-MM).
- **Endorsements:** Names and acronyms of professional or government organizations that support this IEPD as an official business information exchange package.
- **Sponsors:** Name of organization(s) that sponsored, contributed, or participated in the development of the IEPD.
- **URI:** Universal Identifier (each IEPD version will have a distinct URI).
- **Web Site URL of IEPD:** URL of Web site where this IEPD and related artifacts (e.g., XML schema, documentation, mapping spreadsheets) are posted.
- **Message Exchange Patterns:** Category of transaction for which this IEPD is designed and used. Select "query/response," "message," "publish/subscribe," or "document."
- **Lineage:** IEPDs from or with which this IEPD was derived or built; identified by URI.
- **Relationships:** URIs of other IEPDs and their relationship to this IEPD (should not duplicate other attributes such as Lineage, LoB, Organization, etc.).
- **Keywords:** Search terms that would not otherwise be in other metadata attributes.
- **Purpose:** A short description of the business reason for using this IEPD (may include brief statement of scope).
- **Communications Environment:** Description of the primary communications environment(s) for which this IEPD was designed (wireless, satellite, broadband, T1, etc.).
- **Exchange Partners:** Names of the organizations that are using this IEPD.
- **Domains:** Primary domains or line of business (LoBs) that this IEPD covers.
- **Exchange Partner Categories:** Types of organizations that would use this IEPD.
- **Process:** The business rules and activities associated with this IEPD.
- **Triggering Event:** Event(s) that cause this IEPD to be exchanged.
- **Conditions:** Circumstances under which this IEPD is exchanged.
- **Organization Name:** Include both full name and acronym (as appropriate) to enhance discovery.
- **Address 1, Address 2, City, State, Zip, Country:** The full address of the organization.

- **Website:** URL of the Web site of the authoritative source organization.
- **Category:** Type of authority to create IEPD. Select either "statutory," "policy," "both," or "none."
- **Point of Contact Name:** Person designated as the POC for the authoritative source who can provide information, effect change, etc.
- **Address 1, Address 2, City, State, Zip, and Country:** The full address of the contact person within the organization.
- **E-mail:** E-mail address of contact person within the organization.
- **Phone:** Phone number (xxx-xxx-xxxx) of contact person within the organization.
- **Fax:** Fax number (xxx-xxx-xxxx) of contact person within the organization.

Click the "Next" button to review your artifacts and metadata before finalizing the IEPD.

IEPD Details Page

On this page, verify that you have included all the artifacts you want to incorporate into your IEPD. To make changes, click **Edit** in the section that includes the artifact or metadata you wish to update.

- Click the **Validate IEPD** button to get a report of any missing required artifacts or metadata.[42]
- Click the **Create IEPD** button to create a .zip archive of your artifacts and store it within your account space.
- Once you have successfully created your IEPD, click **Download** to download and save, or open the .zip archive on your computer.
- Click **New Version** to create a new IEPD using a copy of your IEPD artifacts and metadata as a starting point for changes and updates.
- Click **Edit** to go to the **IEPD Edit Options** page. On this page, click **Edit Metadata and Artifacts** to go back through the previous artifacts and metadata pages to make changes and updates to your IEPD.
- Click **Delete** to go to the **Delete IEPD** page, where you can download and save or open the IEPD on your computer and verify that you wish to delete the selected IEPD.
- Click **Register** for information about sharing and registering your IEPD on the OJP IEPD Clearinghouse.
- Click **Edit Visibility/Sharing** to go to the **Edit Artifact Visibility** page, where you can change the sharing/visibility attribute from the default **Not Shared** to **Shared** so that other NIEM.gov users will be able to see and access the IEPD you have created.

[42] The Validate IEPD function will only report whether the minimum required artifacts and metadata are included in your package. It will not validate whether your schemas are NIEM-conformant. For more information on NIEM conformance, see Appendix A: Data Model Conformance Guidelines.

- Click the **Update Visibility** button to commit your change to the sharing attribute.

Uploading a NIEM IEPD

This screen will take you through the steps needed to upload an existing IEPD. Although you can create an IEPD without the use of the IEPD Tool,[43] the Upload NIEM IEPD function will work properly only with IEPDs created with the IEPD Tool.

- From the **Upload an IEPD** screen, click the **Begin** button to start the process.
- On the following screen, click the **Browse…** button to locate and select an IEPD to be uploaded.
- Click the **Next** button to upload the file and review the artifacts extracted from the .zip archive.
- You can add artifacts to the IEPD by clicking the appropriate **Browse…** button and then locating and selecting the artifact file from your computer.
- You can delete artifacts from the IEPD by clicking **remove** next to the artifact to be deleted.
- From this point forward, the tool works the same as the Create NIEM IEPD and Edit IEPD functions. Refer to the instructions above for more information.

Generating a Code List Schema

In XML specifications, a code list schema allows you to restrict the permissible values that a particular data entity can contain within an instance document. In NIEM, you can use the Generate Code List Schema Tool to create a NIEM-conformant schema enabling an application to validate XML data against a list of restricted values. To access the tool from the NIEM Tools page, roll over and select Generate Code List Schema from the list of tools on the left.

The Generate Code List Schema Tool is relatively simple to use. Start by downloading the Excel template file to your computer. Then modify the spreadsheet to suit your needs and upload it to the tool to generate your code list.

To download the template, click the link **template.xls** and save the file to your computer. To modify the spreadsheet, open the Excel file to the first tab. The first line of the spreadsheet contains the code list name. The second line contains the code list definition. Rename the default code list name and definition to something appropriate to your exchange.

Code List names, like data element and data type names, should conform to the NIEM *Naming and Design Rules.*[44]

[43] An IEPD is basically a .zip file containing IEPD artifacts. You can create one manually using a commercially available archiving tool such as WinZip, WinRAR, WinAce, etc.

[44] http://www.niem.gov/topicIndex.php?topic=file-NDR-withoutLineNum.

The code list values and descriptions start on Line 4 of the spreadsheet and continue, one line per code value, to the end of the list. Additional code lists can be created in the same spreadsheet by adding additional workbook tabs. (See Figure 77.)

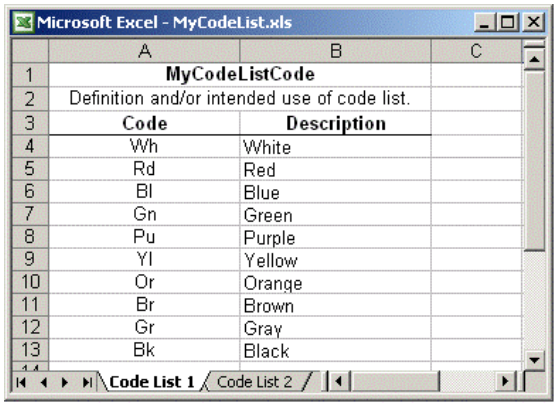


Figure 77: Code List Template.

When you finish modifying the code list template, enter the namespace prefix, namespace URI, and version in the appropriate textboxes on the screen. Click the **Browse** button to locate and select the spreadsheet from your computer. Finally, click the **Build Schema Code List** button to generate your code list schema and save the resulting file to your computer. (See Figure 78.0)

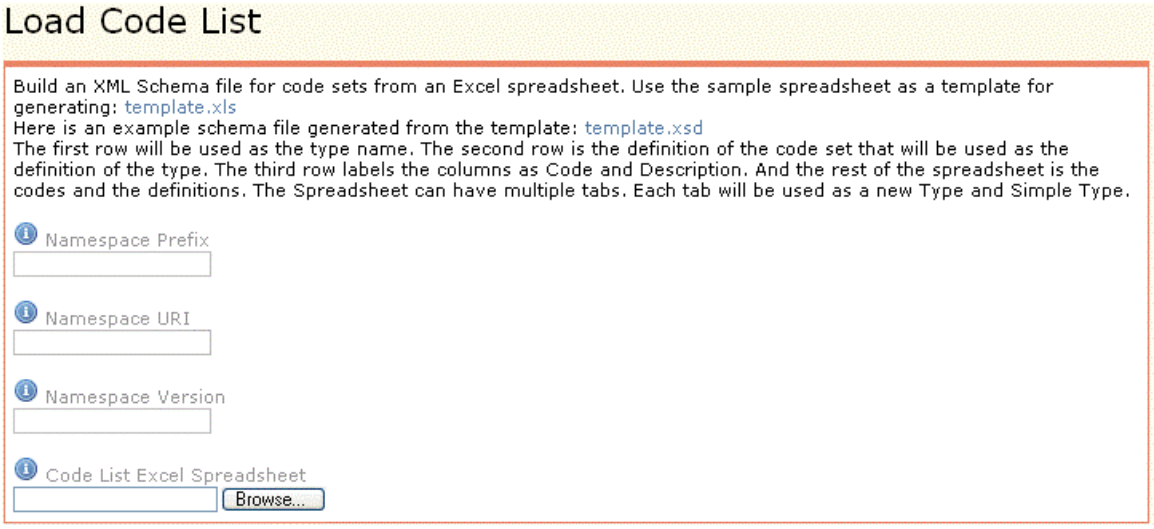


Figure 78: Code List Generation Tool.

Migration Assistance Tool (MAT)

You can use the NIEM Migration Assistance Tool to help convert your GJXDM 3.0.x or NIEM 1.0 wantlist to a NIEM 2.0 wantlist. The Migration Assistance Tool is fairly simple and straightforward and has only one option—identifying the version of the wantlist used as input.

To access the tool from the NIEM Tools page, roll over and select **Migration Assistance** from the list of tools on the left.

To use the tool, click the **Browse…** button to select the wantlist to be converted and then click the drop-down button to select the version of the wantlist. If you are not sure whether you are starting with a NIEM 1.0 or GJXDM 3.0.3 wantlist, open the XML file in a text browser and read the second line of code. If it is a NIEM 1.0 wantlist, the code will read:

```
<w:WantList w:release="1.0" w:product="NIEM xmlns:w="http://niem.gov/niem/wantlist/1">
```

If it is a GJXDM wantlist, the code will read:

```
<w:wantList w:release="3.0.3" xmlns:w="http://gjxdmtools.gtri.gatech.edu/wantList/1">
```

To complete the conversion process, click the **Migrate Wantlist** button. (See Figure 79.)

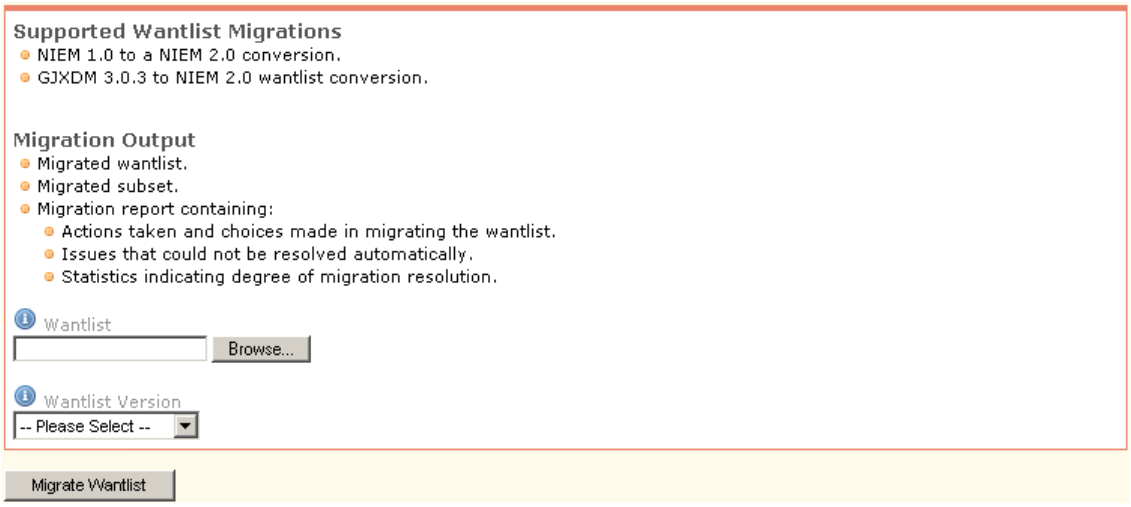


Figure 79: Select a Wantlist File and Version to Migrate.

The results page will display the outcome of the conversion process, including the number and percentage of components that converted automatically as well as the number and percentage of components that did not migrate automatically and require further action. On this page, you can download the converted wantlist, schema subset, or migration report in either HTML or Excel format. To download and save or open the wantlist file to your computer, click **Download Wantlist**. To download and save or open the subset schemas to your computer, click **Generate Subset**. To download an Excel version of the migration report to your computer, click **Download Spreadsheet Report**. To open an HTML version of the migration report in a new window, click **Download HTML Report.** (See Figure 80.)

The migration report is useful for displaying the detailed listings of each individual type or element, whether they were successfully migrated or manual intervention is needed. In the

migration report, the list of components is broken down into two categories—Items Requiring Further Action and Items Not Requiring Further Action. As with any automated tool, users must review the result to ensure the mapping is appropriate and, in some instances, may need to make changes because of tool limitations.

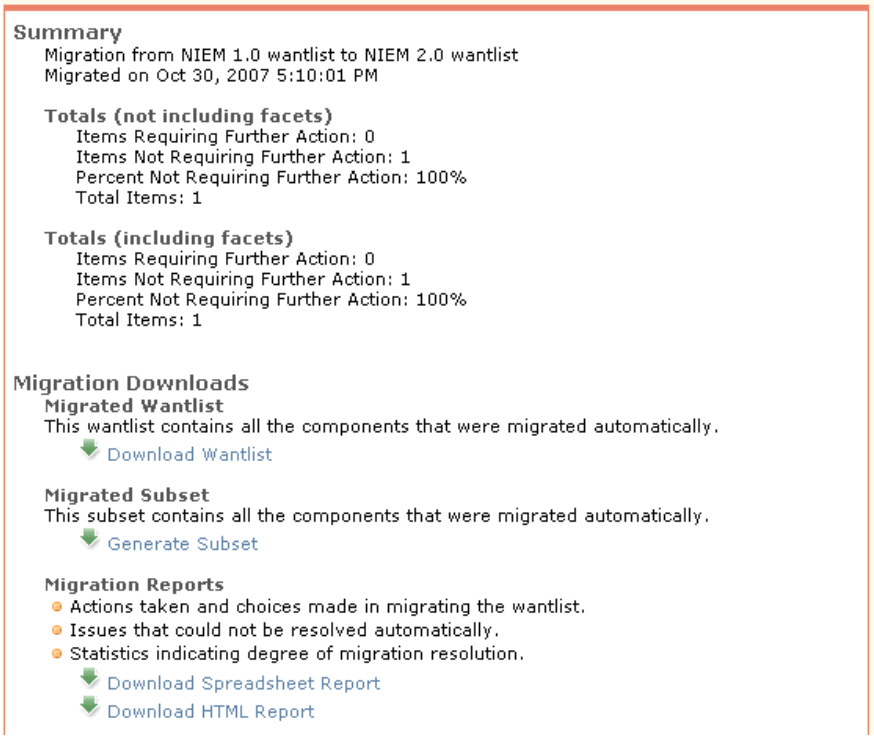


Figure 80: Migration Summary.

No Further Action Required. These are the items that were migrated automatically. Generally, this category includes simple transformations such as property and type name changes, namespace changes, and those items that have not changed from NIEM 1.0 to NIEM 2.0. If you load the migrated wantlist in the SSGT, no further action is required to include these items in the NIEM 2.0 subset schemas.

Items Requiring Further Action. These items that were not migrated automatically generally fall under one of two categories. Either the GJXDM element has not yet been mapped to the equivalent NIEM 2.0 element and the tool does not know how to handle the migration, or you are migrating a NIEM 1.0 wantlist and an element that existed in NIEM 1.0 is no longer available and has not been replaced in NIEM 2.0. The best course of action in these cases is to load the migrated wantlist into the SSGT and manually replace the missing items with the NIEM 2.0 equivalent. If you need the component in your exchange and no equivalent exists in NIEM 2.0, move the element to the extension schema.

NOTE: The tool will not assist in migrating your exchange, extension, or constraint schemas.

Best Practices

As new components have been added to NIEM 2.0, it is possible that the elements you included in your extension schema, because no equivalent existed in NIEM 1.0 or GJXDM 3.0, now have an equivalent in NIEM 2.0. You should use the NIEM elements where they provide a semantic match to the extension element you created in your original IEPD. This means your migrated subset schemas should include those elements that were originally in your extension schema and your document schema should reflect the new NIEM version of the element.

Appendix C: NIEM Resources

NISS Help Desk and Knowledge Base

The National Information Sharing Standards (NISS) Help Desk assists users in finding answers to technical questions regarding the content, principles, and best practices for using NIEM and other information sharing standards and tools. More than a conventional help desk, the NISS Help Desk contains a significant Knowledge Base that users can access online and then submit unanswered questions via the Web or telephone.

The NISS Help Desk and Knowledge Base are made possible through unique collaboration and funding support from the U.S. Department of Justice (DOJ), the U.S. Department of Homeland Security (DHS), and the U.S. Department of Transportation (DOT). Other partners include DOJ's Global Justice XML Structure Task Force (XSTF), the Georgia Tech Research Institute (GTRI), the IJIS Institute, National Center for State Courts (NCSC), and SEARCH—The National Consortium for Justice Information and Statistics.

The NISS Knowledge Base is a self-service interactive database that contains a variety of articles with the best available information from a variety of sources.

If a question cannot be answered by Knowledge Base, it may also be submitted via Internet or telephone to the NISS Help Desk.

The NISS Help Desk support is categorized into three levels. The goal of the Help Desk is to get answers back to developers within 24 hours, whenever possible.

- **Tier 1 Support:** Tier 1 support is available to users each Monday through Friday from 9:00 a.m. to 8:00 p.m. (EST), excluding federal holidays. Support is available via telephone, Web, and e-mail. This team provides support to user inquiries on issues related to GJXDM and NIEM.
- **Tier 2 Support:** Tier 2 support is available to users after an initial investigation is conducted and the request requires additional resources to provide advanced support services related to domain or technical expertise.
- **Referrals:** Referrals are provided to users after an initial investigation is conducted and a referral to an organization outside the Help Desk is advisable. These referral organizations specialize in training, technical assistance, new functionality, software corrections, and governance.

Hours of Operation	Access the NISS Help Desk via:	Access the NISS Knowledge Base via:
Phone Support: 9:00 A.M. – 8:00 P.M. (EST) E-mail Support: 9:00 A.M. – 8:00 PM (EST) Web: 24 hours a day, 7 days a week	Web: http://it.ojp.gov/NISS/helpdesk Phone: (877) 333-5111 (703) 726-1919 E-mail: nisshelp@ijis.org	Web: http://it.ojp.gov/NISS/helpdesk

NIEM Web Site[45]

The NIEM Web site, www.NIEM.gov, is a repository for the latest NIEM documentation and downloads, including model schemas, tools, and supplemental resources.

IEPD Clearinghouse

The IEPD Clearinghouse is an interactive repository Web site that provides government and industry IT professionals with information about planned, in-progress, and completed IEPD initiatives. Public and private developers can maximize resources and time by using the IEPD Clearinghouse to gain access to GJXDM and NIEM-compliant reusable artifacts. Funding agencies, policy makers, and managers can avoid duplicative efforts by researching in-progress IEPD development initiatives. Most important, the IEPD Clearinghouse enables directly relevant collaboration between organizations and people working to solve similar problems within the justice and public safety communities.

While the IEPD Clearinghouse site provides descriptive information about IEPDs, it does not contain the actual IEPDs and associated artifacts (such as documents, schema, etc.). To locate and download the actual IEPDs and artifacts, refer to the Web site listed under each IEPD information article. The IEPD Clearinghouse can be accessed at http://it.ojp.gov/iepd/.

IEPD Clearinghouse Benefits and Features

- Enables search for information about planned, developed, or implemented IEPDs.
- Allows organizations to share IEPD information.
- Provides links to real-world, reusable IEPD artifacts.
- Accelerates the design and development processes.
- Promotes utilization of information sharing standards such as GJXDM and NIEM.

NIEM Training

All courses are taught by practicing IT professionals with years of justice and public safety information technology design and implementation experience from both the public and private sectors. Two different course types are available.

- **The NIEM Executive Briefings** are two- to four-hour sessions targeting senior executives and decision makers.

[45] http://www.niem.gov/.

- **The NIEM Practical Implementer's Course** is a three-day, highly technical session for developers and implementers that begins with an introduction designed to provide a basic knowledge of XML. The Practical Implementer's Course includes exercises and a capstone case study, laying a solid foundation for NIEM knowledge.

For more information about NIEM training or to schedule a NIEM training, contact information@NIEM.gov or training@ijis.org. View the calendar of upcoming NIEM training and events at http://www.niem.gov/calendar/month.php.

NIEM Documents

- Documents aimed primarily at developers and implementers include:
  - Concept of Operations[46]
  - Naming and Design Rules[47]
  - NIEM Implementation Guidelines[48]
  - NIEM Terms and Definitions[49]
  - NIEM FAQs[50]
  - IEPD Requirements Specification[51]
  - Techniques for Building and Extending NIEM XML Components[52]
  - Summary of Changes: NIEM 1.0 to 2.0
- NIEM documents aimed primarily at executives include:
  - Executive Message[53]
  - Introduction to NIEM[54]
  - Value of NIEM[55]
  - Why NIEM Now[56]
  - 10 Key Points About NIEM[57]

[46] http://www.niem.gov/topicIndex.php?topic=file-conops.
[47] http://www.niem.gov/library.php#technical.
[48] http://www.niem.gov/implementationguide.php.
[49] http://www.niem.gov/topicIndex.php?topic=file-glossary.
[50] http://www.niem.gov/topicIndex.php?topic=FAQsPDF.
[51] http://www.niem.gov/topicIndex.php?topic=file-iepdRequirements.
[52] http://www.niem.gov/topicIndex.php?topic=techPDF.
[53] http://www.niem.gov/topicIndex.php?topic=file-briefing.
[54] http://www.niem.gov/topicIndex.php?topic=file-introduction.
[55] http://www.niem.gov/topicIndex.php?topic=ValueOfNIEMPDF.
[56] http://www.niem.gov/topicIndex.php?topic=whyNIEMnowPDF.
[57] http://www.niem.gov/topicIndex.php?topic=10KeyPointsPDF.

Appendix D: Changes in NIEM Constructs Versus GJXDM 3.0.3 Constructs

GJXDM was the precursor to NIEM. Both standards employ the constructs of associations, roles, and metadata, but each prescribes different mechanisms in the application of those constructs. Section 6 discussed the mechanisms prescribed by NIEM. This appendix briefly demonstrates how those constructs are put into practice in GJXDM in similar situations.

Associations in NIEM Versus Associations in GJXDM

NIEM recommends that a relationship between objects be modeled as a type (more specifically, an *association type*, as long as certain other criteria are satisfied). In contrast to NIEM, GJXDM models a relationship between objects as a *property*. The property can be represented in two forms—a *content element* or a *reference element*. For example, the relationship between a court order (an activity) and a judge (a person) issuing the court order can be represented in GJXDM by the content element j:ActivityIssuingJudge, or by the reference element j:ActivityIssuingJudgeReference contained in j:ActivityType.

The two options can be depicted as follows.

Figure 81: Example of GJXDM Property Represented as a Content Element.

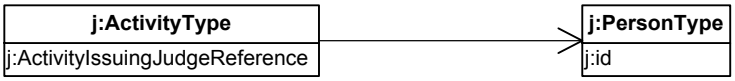


Figure 82: Example of GJXDM Property Represented as a Reference Element.

The following XML schema fragment from GJXDM shows the two options.

```
<!-- Subset schema (Justice namespace) →
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:i="http://www.it.ojp.gov/jxdm/appinfo/1"
xmlns:j="http://www.it.ojp.gov/jxdm/3.0.3"
targetNamespace="http://www.it.ojp.gov/jxdm/3.0.3"
…>
  …
  <xsd:attribute name="ref" type="xsd:IDREF"/>
  <xsd:complexType name="ReferenceType">
    <xsd:attribute ref="j:ref"/>
    <xsd:attributeGroup ref="j:SuperTypeMetadata"/>
  </xsd:complexType>
  …
  <xsd:element name="ActivityIssuingJudge" type="j:PersonType" nillable="true"/>
  <xsd:element name="ActivityIssuingJudgeReference" type="j:ReferenceType"/>

  <xsd:complexType name="ActivityType">
```

```
    <xsd:annotation>
      <xsd:appinfo>
        <i:info>
          <i:base i:namespace="http://www.it.ojp.gov/jxdm/3.0.3" i:name="SuperType"/>
        </i:info>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="j:SuperType">
        <xsd:sequence>
          <xsd:element ref="j:ActivityID" minOccurs="0" maxOccurs="unbounded"/>
          …
         <xsd:element ref="j:ActivityIssuingJudge" minOccurs="0" maxOccurs="unbounded"/>
         <xsd:element ref="j:ActivityIssuingJudgeReference" minOccurs="0"
maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  …
</xsd:schema>
```

Figure 83: GJXDM XML Schema Fragment Illustrating the Definition of j:ActivityType.

Roles in NIEM Versus Roles in GJXDM

NIEM models the role of an entity (a person or an organization) as a new type that points to that entity through a RoleOf reference. In contrast to NIEM, GJXDM roles are modeled as specializations of entities. For example, a missing person (j:MissingPersonType) is modeled as a specialization of a person (j:PersonType) in GJXDM.

This can be depicted as follows.

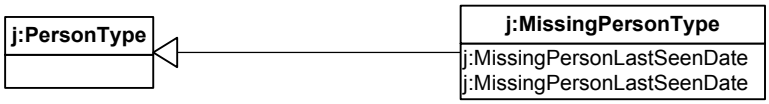


Figure 84: Definition of j:MissingPersonType.

The following XML schema fragment from GJXDM shows the definition of j:MissingPersonType.

```
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:j="http://www.it.ojp.gov/jxdm/3.0.3"
xmlns:i="http://www.it.ojp.gov/jxdm/appinfo/1"
targetNamespace="http://www.it.ojp.gov/jxdm/3.0.3"
…>
  …
  <xsd:complexType name="MissingPersonType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:info>
          <i:base i:namespace="http://www.it.ojp.gov/jxdm/3.0.3" i:name="PersonType"/>
```

```
        </i:info>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="j:PersonType">
        <xsd:sequence>
          <xsd:element ref="j:MissingPersonID" minOccurs="0" maxOccurs="unbounded"/>
          …
          <xsd:element ref="j:MissingPersonLastSeenDate" minOccurs="0"
maxOccurs="unbounded"/>
          …
          <xsd:element ref="j:MissingPersonLastSeenLocation" minOccurs="0"
maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
```

Figure 85: GJXDM XML Schema Fragment Illustrating the Definition of j:MissingPersonType.

Metadata in NIEM Versus Metadata in GJXDM

In GJXDM, j:SuperType contained 23 attributes representing metadata. Since all types in GJXDM ultimately derived from j:SuperType (see the following XML schema fragment), they inherited these 23 metadata attributes.

This method has many limitations. Because an xsd:attribute cannot be extended or restricted, it is not possible to capture additional metadata information. Because an xsd:attribute can only contain a value of the type xsd:string, it is not possible to capture metadata information that has a complex structure.

The following XML schema fragment shows the definition of j:SuperType and j:TargetType in GJXDM.

```
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:j="http://www.it.ojp.gov/jxdm/3.0.3"
xmlns:i="http://www.it.ojp.gov/jxdm/appinfo/1"
targetNamespace="http://www.it.ojp.gov/jxdm/3.0.3"
…>
  …
  <xsd:attributeGroup name="SuperTypeMetadata">
    <xsd:attribute ref="j:commentText" use="optional"/>
    <xsd:attribute ref="j:criminalInformationIndicator" use="optional"/>
    <xsd:attribute ref="j:distributionText" use="optional"/>
    <xsd:attribute ref="j:effectiveDate" use="optional"/>
    <xsd:attribute ref="j:effectiveTime" use="optional"/>
    <xsd:attribute ref="j:expirationDate" use="optional"/>
    <xsd:attribute ref="j:expirationTime" use="optional"/>
    <xsd:attribute ref="j:intelligenceInformationIndicator" use="optional"/>
    <xsd:attribute ref="j:languageText" use="optional"/>
    <xsd:attribute ref="j:lastUpdatedDate" use="optional"/>
    <xsd:attribute ref="j:lastUpdatedTime" use="optional"/>
    <xsd:attribute ref="j:lastVerifiedDate" use="optional"/>
    <xsd:attribute ref="j:lastVerifiedTime" use="optional"/>
    <xsd:attribute ref="j:probabilityNumeric" use="optional"/>
    <xsd:attribute ref="j:reliabilityNumeric" use="optional"/>
    <xsd:attribute ref="j:reportedDate" use="optional"/>
    <xsd:attribute ref="j:reportedTime" use="optional"/>
```

```
    <xsd:attribute ref="j:reportingOrganizationText" use="optional"/>
    <xsd:attribute ref="j:reportingPersonRoleText" use="optional"/>
    <xsd:attribute ref="j:reportingPersonText" use="optional"/>
    <xsd:attribute ref="j:sensitivityText" use="optional"/>
    <xsd:attribute ref="j:sourceIDText" use="optional"/>
    <xsd:attribute ref="j:sourceText" use="optional"/>
  </xsd:attributeGroup>

  <xsd:complexType name="SuperType">
    <xsd:attributeGroup ref="j:SuperTypeMetadata"/>
    <xsd:attribute ref="j:id"/>
  </xsd:complexType>

  <xsd:complexType name="TargetType">
    <xsd:annotation>
      <xsd:appinfo>
        <i:info>
          <i:base i:namespace="http://www.it.ojp.gov/jxdm/3.0.3" i:name="SuperType"/>
        </i:info>
      </xsd:appinfo>
    </xsd:annotation>
    <xsd:complexContent>
      <xsd:extension base="j:SuperType">
        <xsd:sequence>
          <xsd:element ref="j:TargetName" minOccurs="0" maxOccurs="unbounded"/>
          …
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>

  …
</xsd:schema>
```

Figure 86: XML Schema Fragment Illustrating the Definition of j:SuperType in GJXDM.

Appendix E: Glossary of Terms and Acronyms

Glossary of Terms

Term	Definition
Architecture	Architecture refers to the design of a system. It may refer to either hardware or software or a combination of both. The software architecture of a program or computing system is the structure or structures of the system. This structure includes software components, the externally visible properties of those components, the relationships among them, and the constraints on their use.
Artifact	Any tangible and potentially reusable documentation or output pertaining to an existing or potential information exchange.
Association	A NIEM construct that represents a relationship among two or more objects. A type, named for the kind of relationship it represents, links multiple objects under specific contexts and may contain properties that are characteristics of the relationship. This allows preservation of the object-oriented design principles of the data model, while allowing more granular specificity of meaning when two or more data objects are related.
Attribute	A characteristic of an object whose value may be used to help distinguish one instance of an object from others.
Augmentation	A method that has been developed to enable the reuse of type extensions that occur within particular domains for use elsewhere. This augmentation process seeks to avoid the duplicative defining of extensions that could not have been easily shared for mutual benefit before now.
Authoritative Source	The organization or entity (in NIEM, often a domain or community of interest) that has taken ownership of and update responsibility for a particular IEPD or other exchange artifact including schemas, code lists, etc.
Business Component Library (BCL)	The concept of facilitating the creation and storage of reusable components for NIEM IEPD creation. Business components typically consist of an aggregation of data components into a construct that serves a specific business need, such as assembling name and address elements to create a Home Address component. These components can then be reused, saving development time and costs and avoiding duplication of effort across NIEM implementations.
Business Context	A common frame of reference across business areas or domains allowing organizations to share information with specific goals or scenarios in mind.
Business Functions	The operations and procedures carried out to fulfill a business need or needs.

Term	Definition
Business Model	A view of the business at any given point in time. The view can be from a process, data, event, or resource perspective and can be the past, present, or future state of the business. Creating a business model is often one of the initial steps when exploring information sharing needs and potentials.
Business Need	Often used as a justification for decisions or actions in a business setting, the business need addresses those outcomes that would most assuredly achieve business success.
Business Requirements	The requirements implicit in a transaction or information exchange in order to satisfy the business need of the parties involved. Business requirements and rules are often documented within an IEPD.
Business Rules	Policies and other restrictions, guidelines, and procedures that constrain the use of information exchanges. Often, these rules are incapable of being documented directly within the XML schema artifacts within an IEPD and thus must be documented separately and agreed upon by parties engaging in the exchange.
Business Scenarios	Real-world scenarios that are used to describe or justify a use case for a certain business model.
Cardinality	The number of instances of an entity in relation to another entity, e.g., one-to-one, one-to-many, many-to-many.
Change Management	The process of developing a planned approach to change in an organization. In NIEM, often refers to managing change impacts of new releases and modifications to the data model or domain structure.
Class	Description of a set of objects that share the same attributes, operations, methods, relationships, and semantics.
Code	A symbolism or abbreviation of a term or concept meant to shorten the communication time and eliminate potential ambiguity of meaning.
Code Table	A set of related codes and their definitions. In NIEM, code tables have their own namespaces and can be internal or external.
Common	A NIEM concept referring to the common semantic understanding of data components across more than one business domain.
Common Component	A data component in the NIEM Core namespace, marked with metadata to indicate that it is shared by one or more NIEM domains. See also ***Universal Component***.
Common Data Component	Data components used in exchanges between two or more domains but not universally shared.

Term	Definition
Common Vocabulary	A term for consistency of definition of terms across domains or communities of interest. This is the primary goal of the NIEM data dictionary.
Community Of Interest (COI)	A group of organizations or government agencies with a common interest, often to share information that typically can act authoritatively when developing, harmonizing, and managing the data components used in interdomain exchanges.
Community Of Practice (COP)	Lines of business within the government and external organizations that are dedicated to the support of common business functions. Communities of practice are often less formalized in function and decision making than communities of interest and serve as a forum to share best practices and work products.
Component	An object, meant to interact with other objects, that encapsulates certain inherent functionality. These may be organizational components, data components, etc. In NIEM, component is often used to describe data elements that are either Universal, Common, or Domain-specific in the NIEM framework.
Component Mapping Template (CMT)	The tool of choice for mapping components that are used by organizations or domains that are being compared with those that currently exist in NIEM to identify overlap or gaps between the two.
Conceptual Data Model (CDM)	A data model that defines the real-world entities and the relationships between these entities in a business context. A CDM is typically constructed as an Entity Relationship Diagram (ERD), e.g., a UML class diagram.
Configuration Management	The control and adaptation of the evolution of complex systems and the evaluation and approval of changes that affect interrelationships between components of those systems. Configuration management is closely related to change management.
Conformance	The requirement that those who participate in NIEM by contributing data components or creating and sharing IEPD artifacts are following the agreed-upon procedures for doing so and that all documentation meets minimum criteria and the NIEM Naming and Design Rules where applicable.
Conformant Schema	A schema that maintains the XML schema syntax requirements of NIEM as specified by the NIEM Naming and Design Rules.
Constraint Schema	A schema with the purpose of restricting or constraining content that appears in instances of the subset schema.
Controlled Vocabulary	A list of terms that have been enumerated explicitly with unambiguous, nonredundant definitions and are governed by a COI.

Term	Definition
Core	The Core refers to the NIEM data model, which is composed of the Universal and Common namespaces, containing all components that are determined to be relevant and semantically agreed upon by some or all participating domains. NIEM Core could be said to contain all reusable components that are not domain-specific and are governed by NIEM processes and policies regarding promotion and maintenance of those components.
Core Component	A data component that meets the criteria to be promoted to the Common or Universal namespaces.
Data	Facts represented in a readable language (such as numbers, characters, images, or other methods of recording) on a durable medium. Data on their own carry no meaning. Empirical data are facts originating in or based on observations or experiences. A database is a store of data concerning a particular domain. Data in a database may be less structured or have weaker semantics (built-in meaning) than knowledge in a knowledge base. Compare data with information.
Data Architecture	A component of the design architecture, the data architecture consists of, among others, data entities, which have attributes and relationships with other data entities. These entities are related to the business functions.
Data Artifact	A collective term for electronic artifacts related to the presentation, description, representation, or storage of data. Examples are documents and XML schemas.
Data Component	Basic business data items that represent real-world objects and concepts. Information that is exchanged between agencies can be broken down into individual data components—for example, information about people, places, material things, and events.
Data Dictionary	A set of data elements and their definitions, including any metadata and representations associated with them.
Data Element	A basic unit of data having definition, identification, representation, and values; the lowest level of physical representation of data.
Data Exchange	Fixed, recurring transactions between parties, such as the regular exchange of environment testing data among federal, state, local, and tribal entities.
Data Harmonization	The process of comparing two or more data component definitions and identifying commonalities among them that warrant being combined or harmonized into a single data component.
Data Model	A graphical and/or lexical representation of data, specifying its properties, structure, and interrelationships.

Term	Definition
Data Object	An aggregation of information from data component(s) that represent discrete information about a subject area. Data objects with a clear business context become business components.
Data Promotion	The identification of data components that are semantically agreed upon between NIEM domains, or among all NIEM domains, and are reclassified in a higher-level namespace.
Data Reference Model (DRM)	One of the five models in the Federal Enterprise Architecture reference model framework to aid in describing the types of interactions and exchanges that occur between the federal government and its various customers, constituencies, and business partners.
Data Registry	A registry that is centered on the discovery of data elements and components. See also *Registry*.
Data Repository	A repository that is centered on the storage and cataloging of data elements and components. See also *Repository*.
Data Standard	Agreed-upon structure for representing data in machine-readable format, often used to facilitate information exchange through common understanding and recognition of the data elements used.
Data Steward	A data steward has the role of surrogate owner of a data element or entity for an enterprise. A data steward provides the definition and parameters of a data element or entity for the enterprise.
Data Type	A constraint on the type of data that an element or attribute may hold (e.g., “date,” “string,” “float,” or “integer”).
Discovery	The act of locating a machine-processable description of a Web service-related resource that may have been previously unknown and that meets certain functional criteria. It involves matching a set of functional and other criteria with a set of resource descriptions. For NIEM, discovery normally refers to the search for IEPDs and data components within a repository that can be reused in IEPD development.
Document	A file containing unstructured and/or semistructured data resources. A discrete and unique electronic aggregation of data produced with the intent of conveying information.
Domain	A set of people, organizations, and processes having comparable business functions designed to achieve similar goals irrespective of organizational boundaries. Domains often have the business requirement and the capability to harmonize data for exchange. In NIEM, domains are Lines of Business (LoBs).
Domain Model	A domain model is a conceptual view of a system or an information exchange that identifies the entities involved and their relationships.

Term	Definition
Domain-Specific Components	A component that meets technical standards, complies with NIEM requirements, and is of interest to a specific domain managed and harmonized by the appropriate COI.
Element	The fundamental building block of an XML document. XML elements can contain other elements and/or text data. XML elements are composed of a start tag, content, and end tag.
Enterprise	A business association consisting of a recognized set of interacting business functions, able to operate as an independent, stand-alone entity. Enterprisewide information sharing is meant to refer to the breadth and diversity of informational needs of such an association.
Entity	An information sharing unit. All agencies are entities; so are courts and legislative bodies. Private organizations that share governmental information are also entities, as are private persons.
Exchange Mapping	The process of comparing desired exchange content to the exchange specifications to ensure semantic compatibility prior to information exchange.
Exchange Model	A reference to the National Information Exchange Model as a provider of exchange modeling standards and best practices.
Exchange Package	A description of specific data exchanged between a sender and a receiver. The exchange package is usually coupled with additional documentation, sample XML instances, business rules, etc. to compose an IEPD.
Exchange Schema	A schema with the purpose of defining the actual content model of the information exchange within an IEPD. The document schema works in conjunction with the subset, extension, and constraint schemas to form a complete package that represents the exchange.
Exchange Specification	Any details describing the exchange, including schemas, business rules, and more. This term often describes the contents of an Information Exchange Package.
eXtensible Markup Language (XML)	A structured, extensible language for describing information being sent electronically from one entity to another. XML schema is the preferred standard to define the rules and constraints for the characteristics of the data, such as structure, relationships, allowable values, and data types.
Extension Schema	An XML schema that defines data elements that are to be used in an exchange but do not exist in the NIEM model, which, therefore, must be extended.
External Standard	A standard with a governing body outside the scope of NIEM whose products must be used in conjunction with NIEM in exchanges.

Term	Definition
Framework	In software development, a framework is a defined support structure in which another software project can be organized and developed. A framework may include support programs, code libraries, a scripting language, or other software to help develop and glue together the different components of a software project.
Functional Standard	A standard describing the functionality and business processes that are required when performing business tasks or functions. Functional standards do not specify the actual data involved in the process.
Gap Analysis	An analysis performed to identify overlaps and gaps between one or more information sets, systems, or exchange methods. This is often one of the first steps taken by two organizations looking to engage in information exchange.
Global Justice XML Data Model (GJXDM)	A data model and dictionary sponsored by the U.S. Department of Justice and governed by the Global Justice Information Sharing Initiative. The GJXDM and its related processes are the basis on which NIEM was built, in partnership with the U.S. Department of Homeland Security.
Information	Contextual meaning associated with or derived from data.
Information Exchange	The transfer of information from one organization to another, specifically in concert with NIEM IEPD exchange processes and recommended procedures.
Information Exchange Package (IEP)	A description of specific information exchanged between a sender and a receiver. The information exchange package is usually coupled with additional documentation, sample XML instances, business rules, etc. to compose an IEPD. IEP may sometimes be referred to simply as Exchange Package.
Information Exchange Package Documentation (IEPD)	The aggregation of IEP information to form a complete set of documentation to completely describe an information exchange. This may include additional documentation, business rules, sample instance data, etc.
Information Exchange Package Documentation (IEPD) Lifecycle	The IEPD development lifecycle contains a set of steps that should be followed circularly until the final conditions are met. This lifecycle in detail can be found in the NIEM Introduction document.
Information Exchange Package Documentation (IEPD) Template	The template created by NIEM to define the required and optional components that may be included in a NIEM-conformant IEPD.
Information Sharing	The broad concept of sharing information between agencies or organizations that do not inherently have access to such information. The need for robust nationwide information sharing is the guiding principle of the NIEM program.

Term	Definition
Instance	A specific occurrence of an entity. See also *XML Instance*.
Interoperability	The ultimate goal of any information sharing exercise refers to the seamless interconnection between disparate systems for the purpose of sharing information relevant to either party. Interoperability is both a prerequisite and a result of efficient information sharing.
Line of Business (LoB)	A business purpose or function that crosses organizational boundaries. This concept was made popular in particular at the federal government level as a part of the Federal Enterprise Architecture effort to reorganize government resource allocation in a more efficient manner.
Machine-Readable Format	Refers to information or data that is in a format that can be easily processed by a computer without human intervention while ensuring that no semantic meaning is lost.
Message	The basic unit of communication between a requester and a provider of information. A message typically encompasses an IEPD and includes additional transport-specific metadata relating to routing, security, and more.
Metadata	Structured data about data. Metadata includes data associated with either an information system or an information object for purposes of description, administration, legal requirements, technical functionality, use and usage, and preservation.
Namespace	A namespace is a collection of objects in which the names of the objects are unique. The solution to naming conflicts in XML, using XML namespaces, can help alleviate issues that arise where XML elements and attributes use identical names. A namespace typically aligns with a domain that has responsibility over maintaining the components within that namespace.
Naming and Design Rules (NDR)	The NDR specifies rules and requirements of schemas developed under the NIEM program and guarantees conformance to those that follow them. The NDR is intended to ensure interoperability even when different developers are building IEPDs independently.
NIEM Configuration and Control Tool (NCCT)	The primary tool used for inserting and tracking technical and business issues with the NIEM data model and to help the Program Management Office in prioritizing input from the stakeholder community.
NIEM Domain	A business domain that is assigned a NIEM namespace, has responsibility to act as an authoritative source and steward for domain-specific data, and is able to propose promotions of data to Universal or Common namespaces.
NIEM Participating Parties	Organizations that have signed the memorandum of understanding (MOU) for the National Information Exchange Model (NIEM). Participating parties include ODNI, DHS, DOJ, and Global. Other organizations will become participating parties as described in the MOU.

Term	Definition
NIEM.gov	The public Web site meant to serve as the primary entry point for all information and resources related to the NIEM program.
Normalization	A process that eliminates redundancy, organizes data efficiently, and reduces the potential for anomalies during data operations and improves data consistency.
Object-Oriented Programming	Object-oriented programming combines data structures and functions (computer directions) to create "objects," making it easier to maintain and modify software.
Ontology	An explicit formal specification of how to represent the objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that hold among them. In computer science, an ontology is the attempt to formulate an exhaustive and rigorous conceptual schema within a given domain, a typically hierarchical data structure containing all the relevant entities and their relationships and rules (theorems, regulations) within that domain.
Open Architecture	Open architecture systems are designed to allow system components to be easily connected to devices and programs made by other manufacturers.
Pilot Project	A project established to evaluate new technology, or to develop and implement exchange standards for information that is common among NIEM participating agencies and exchanged as part of their current or intended business practices.
Practitioner	Practitioners act as the "users" of information exchange standards, which are implemented in real systems. They can act as data providers and consumers in an information exchange and often act as participants with subject-matter expertise to help define the business needs of information exchanges as they are developed.
Protocol	A set of formal rules describing how to transmit data, especially across a network. Low-level protocols define the electrical and physical standards to be observed, bit- and byte-ordering and transmission, and error detection and correction of the bit stream. High-level protocols deal with the data formatting, including the syntax of messages, the terminal-to-computer dialogue, character sets, sequencing of messages, etc.
Quality Assurance	A process by which the quality of design and performance of a system or data is tested and verified prior to implementation.
Reconciliation	The process of bringing two differing data sets or processes together to be synchronized to promote interoperability between them.

Term	Definition
Reference Architecture	The generalized architecture of several end systems that share one or more common domains. The reference architecture defines the infrastructure common to the end systems and the interfaces of components that will be included in the end systems. The reference architecture is then instantiated to create a software architecture of a specific system. The definition of the reference architecture facilitates deriving and extending new software architectures for classes of systems. A reference architecture, therefore, plays a dual role with regard to specific target software architectures. First, it generalizes and extracts common functions and configurations. Second, it provides a base for instantiating target systems that use that common base more reliably and cost effectively.
Reference IEPD	An IEPD that has been designated as a reference IEPD has been endorsed by an Authoritative Source as a shining example or base exchange template that should be reused as is or modified to perform a similar business function.
Registry	Authoritative, centrally controlled store of information that facilitates discovery and reuse. A NIEM registry of IEPDs would act as a store or pointer to all known IEPDs in existence or currently under development to allow implementers to take advantage of parallel efforts.
Repository	An information system used to store and access information, schemas, stylesheets, controlled vocabularies, dictionaries, and other work products.
Role	A technique specifically adopted to enhance the desired contextual meaning of components in a data exchange. By allowing a data component to take a context-specific "role," the data model becomes infinitely flexible to model a variety of exchange needs. For example, a person could take on the role of a law enforcement official, a witness, or a plaintiff. By utilizing a role methodology, the object-oriented nature of the model can be preserved while allowing explicit customization that does not depend on object inheritance.
Scalability	A term that describes how well a system can be adapted and expanded to meet increased demands and is a key motivating factor to a program such as NIEM with national implications.
Scenario-Based Planning	A process of planning and identifying data exchanges by analyzing a business process and describing information exchanges using use-case scenarios to justify the need for those exchanges.
Schema	See *XML Schema*.
Schema Subset Generation Tool (SSGT)	The preferred tool used to generate schema subsets from the NIEM data model without needing to edit the model schema itself. Subsets are saved and shared via the wantlist mechanism.

Term	Definition
Scope Creep	The slow and continuous expansion of the scope or a project, such as data type or routine, resulting in a broad, unfocused, and unmanageable scope and usually leads to cost overruns, missed deadlines, and loss of original goals.
Semantic Consistency	A driving force behind the need for data standards, consistency of terminology, and data definitions is essential for information exchanges to be effective, understood by all parties involved, and machine-readable.
Service	An abstract resource that represents a capability of performing tasks that form a coherent functionality from the point of view of data providers and requesters.
Service Description	A set of documents that describe the interface to and semantics of a service.
Service Interface	The abstract boundary that a service exposes. It defines the types of messages and message exchange patterns that are involved in interacting with the service, together with any conditions implied by those messages.
Service-Oriented Architecture (SOA)	An architectural style whose goal is to achieve loose coupling among interacting software agents. A service is a unit of work done by a service provider to achieve desired end results for a service consumer. Both service provider and service consumer are roles played by software agents/brokers on behalf of their owners. The communication can involve either simple data exchange or two or more services coordinating some activity. Some means of connecting services to each other is needed.
Service Semantics	The behavior expected when interacting with the service. The semantics expresses a contract (not necessarily a legal contract) between the provider entity and the requester entity. It expresses the effect of invoking the service. Service semantics may be formally described in a machine-readable form, identified but not formally defined, or informally defined via an agreement between the provider and the requester.
Sponsor	An organizational entity that supports a specific IEPD or set of data components for inclusion in NIEM.
Stakeholder	A person or organization that has a vested interest in a project or entity and the direction that entity takes.
Subject-Matter Expert (SME)	Those people or organizations with experience in a particular business process or those practitioners who have demonstrated knowledge of a certain line of business or information exchange area.
Subset Schema	A subset of the primary NIEM Schema, a schema whose components are taken entirely from the parent schema while excluding those components that are unnecessary for a given exchange.

Term	Definition
Type	A description of a class of objects that share the same operations, abstract attributes and relationships, and semantics.
Type Extension	The extension of a type to include additional concepts or components that are necessary for an exchange or to specialize a concept in the data model.
Type Hierarchy	The high-level (abstract) to low-level (specific) arrangement of derived types within a data model.
Uniform Resource Identifier/Uniform Resource Name (URI/URN)	Identifiers meant to explicitly and uniquely identify a namespace or schema location, usually based on a naming convention according to the sponsoring organization.
Universal	A NIEM concept referring to the common semantic understanding of data components across all or nearly all business domains.
Universal Component	A data component in the NIEM Core namespace, marked with metadata to indicate that it is shared by all or nearly all NIEM domains. See also ***Common Component***.
Use Case	A business process example of an information flow, most commonly used as a basis for exchange modeling around the business needs of an organization. See also *Scenario-Based Planning*.
Validation	The documented process of showing that a system is stable and capable of producing predetermined outcomes; answers the question of whether it does what the user really requires.
Wantlist	A portable construct used in the SSGT to save and reuse schema subsets of the overall NIEM data model. A wantlist can be saved or loaded directly from the SSGT tool. A wantlist is an XML instance that specifies the NIEM data components required (and therefore selected) by the user for the subset schema he/she is building. It does not include NIEM data components the user-selected set depends on.
Web Service	A software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

Term	Definition
Web Services Description Language (WSDL)	An XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages, regardless of what message formats or network protocols are used to communicate.
XML	A structured, extensible language for describing information being sent electronically by one entity to another. XML schema is the preferred standard to define the rules and constraints for the characteristics of the data, such as structure, relationships, allowable values, and data types.
XML Instance	An instance of XML that contains actual data whose format and inclusion are controlled by the associated XML schema.
XML Schema	Defines the vocabulary (elements and attributes), the content model (structure, element nesting, and text content), and data types (value constraints) of a class of XML documents. NOTE: When written with a capital "S," the term refers specifically to the XML Schema Definition (XSD or WXS) language developed by the W3C. However, when written with a lowercase "s," the meaning is more generic, referring to any of several schema languages for use with XML, such as DTDs, RELAX NG, Schematron, etc. In both cases, an XML schema is used to validate XML instances to verify that the instances conform to the model that the schema describes.

3946

3947

Glossary of Acronyms

Acronym	Definition
AIC	Architecture and Infrastructure Committee
BCL	Business Component Library
BJA	Bureau of Justice Assistance
BRM	Business Reference Model
CCB	Configuration Control Board
CIO	Chief Information Officer
CIS	Central Index System
CM	Configuration Management
CMT	Component Mapping Template
COI	Community of Interest
ConOps	Concept of Operations
COP	Community of Practice
CTISS	Common Terrorism Information Sharing Standards
DAS	Data Architecture Subcommittee
DHS	U.S. Department of Homeland Security
DMM	Data Model Maturity
DOJ	U.S. Department of Justice
DON	Department of the Navy
DRM	Data Reference Model
ebXML	Electronic Business XML
EIC	Emergency Interoperability Consortium
EMS	Emergency Medical Services

Acronym	Definition
EOC	Emergency Operations Center
ESC	Executive Steering Committee
FACA	Federal Advisory Committee Act
FAQs	Frequently Asked Questions
FEA	Federal Enterprise Architecture
GIS	Geographical Information System
GJXDM	Global Justice XML Data Model
Global JXDM	Global Justice XML Data Model
Global	Global Justice Information Sharing Initiative
GTRI	Georgia Tech Research Institute
GUI	Graphical User Interface
GXSTF	Global XML Structure Task Force
HSIN	Homeland Security Information Network
HSPD	Homeland Security Presidential Directive
IAFIS	Integrated Automated Fingerprint Identification System
ICE	Immigration and Customs Enforcement
ICMWG	Intelligence Community Metadata Working Group
IEM	Information Exchange Modeling
IEP	Information Exchange Package
IEPD	Information Exchange Package Documentation
IRS	Internal Revenue Service
IRTPA	Intelligence Reform and Terrorism Prevention Act
ISE	Information Sharing Environment
ISO	International Standards Organization

Acronym	Definition
IT	Information Technology
JIEM	Justice Information Exchange Model
JMIE	Joint Maritime Information Element
JTTF	Joint Terrorism Task Force
LEISP	Law Enforcement Information Sharing Program
LEO	Law Enforcement Online
LInX	Law Enforcement Information Exchange
LoB	Line of Business
MOU	Memorandum of Understanding
NASCIO	National Association of State Chief Information Officers
NBAC	NIEM Business Architecture Committee
NC&OC	NIEM Outreach and Communications Committee
NCCT	NIEM Configuration Control Tool
NCIC	National Crime Information Center
N-DEx	Law Enforcement National Data Exchange (FBI)
NDR	Naming and Design Rules
NGA	National Geospatial-Intelligence Agency
NIBRS	National Incident Based Reporting System
NIEM ESC	NIEM Executive Steering Committee
NIEM PMO	NIEM Program Management Office
NIEM	National Information Exchange Model
NISS Help Desk	National Information Sharing Standards Help Desk
NIST	National Institute of Science and Technology
Nlets	Nlets, The International Justice and Public Safety Information Sharing Network

Acronym	Definition
NPEP	National Priority Exchange Panel
NTAC	NIEM Technical Architecture Committee
NTIA	National Telecommunication and Information Administration
OASIS	Organization for the Advancement of Structured Information Standards
ODNI	Office of the Director of National Intelligence
OJP	Office of Justice Programs
OOP	Object-Oriented Programming
OWL	Web Ontology Language
PM-ISE	The Program Manager, Information Sharing Environment
PMO	Program Management Office
POST	National Association of Peace Officers Standards and Training
QA	Quality Assurance
QOD	Quality of Design
RC	Release Candidate
R-DEx	Regional Data Exchange (FBI)
RDF	Resource Definition Framework
ROI	Return on Investment
SAR	Suspicious Activity Reporting
SitReps	Situation Reports
SME	Subject-Matter Expert
SOA	Service-Oriented Architecture
SSAN	Social Security Account Number
SSGT	Schema Subset Generation Tool
TWPDES	Terrorist Watchlist Person Data Exchange Standard

Acronym	Definition
URI	Uniform Resource Identifier
URN	Uniform Resource Name
VIN	Vehicle Identification Number
W3C	World Wide Web Consortium
WIP	Work in Progress
WSDL	Web Services Description Language
XML	Extensible Markup Language
XSIWG	XML Schema Interoperability Working Group
XSL	XML Stylesheet Language
XSTF	XML Structure Taskforce

3950

Appendix F: NIEM 2.0 Reference Schemas

This appendix lists the names and descriptions for the code lists and external adapter schemas as supplied with NIEM 2.0.

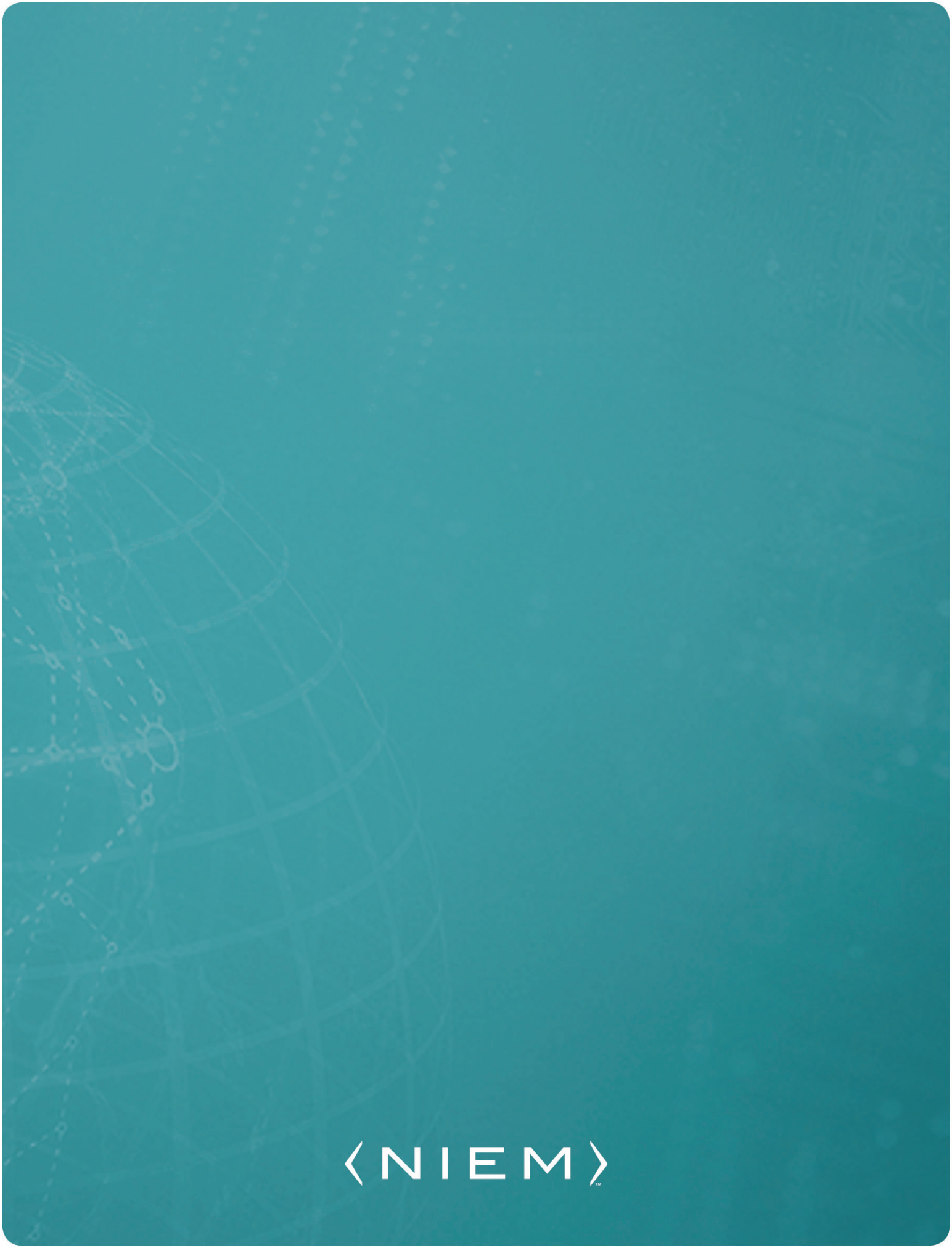
Code Name	Description
ansi_d20	Motor vehicle administration codes from ANSI D20, the Data Dictionary for Traffic Record Systems, maintained by the American Association of Motor Vehicle Administrators (AAMVA).
ansi-nist	ANSI/NIST Fingerprint and Biometric standard.
apco	Association of Public-Safety Communications Officials—International (APCO).
atf	Bureau of Alcohol, Tobacco, and Firearms
post-canada	Province codes for Canada.
census	Employment codes from the U.S. Census Bureau.
dea	Drug Enforcement Administration
dod_jcs-pub2.0-misc	Intelligence discipline codes from the U.S. Department of Defense (DoD) Joint Publication 2.01.
edxl	Emergency Data Exchange Language (EDXL)
edxl-cap	EDXL Common Alerting Protocol
edxl-de	EDXL Distribution Element
fbi	FBI code lists for National Crime and Information Center (NCIC-2000), National Incident-Based Reporting System (NIBRS), and Law Enforcement National Data Exchange (N-DEx).
fips_10-4	Countries, dependencies, areas of special sovereignty, and their principal administrative divisions from the Federal Information Processing Standards (FIPS) 10-4.
fips_5-2	Codes for the identification of the states, the District of Columbia, the outlying areas of the United States, and associated areas from the Federal Information Processing Standards (FIPS) 5-2.
fips_6-4	Counties and equivalent entities of the United States, its possessions, and associated areas from the Federal Information Processing Standards (FIPS) 6-4.

Code Name	Description
geospatial	Defines NIEM adapter types for external geospatial components defined by OGC, LIF, LandXML, IAI, and ANSI. Note for schema readers: The XML/Schema specification does not require processing implementations to transitively import definitions from imported schemas. To ensure that all required definitions are available, a schema must reimport the schemas that are imported by the schemas it imports. Such reimports are noted in the documentation.
have	EDXL Hospital AVailability Exchange (HAVE)
hazmat	Pipeline and Hazardous Materials Safety Administration, Office of Hazardous Materials Safety.
iso_3166	Codes for the representation of names of countries and their subdivisions from the International Organization for Standardization (ISO) 3166-1:1997.
iso_4217	Codes for the representation of currencies and funds from the International Organization for Standardization (ISO) 4217:2001.
iso_639-3	Codes for the representation of names of languages—Part 3: Alpha-3 code for comprehensive coverage of languages.
itis	Integrated Transportation Information System
lasd	Los Angeles County Sheriff's Department
mmucc_2	Model Minimum Uniform Crash Criteria
mn_offense	Statute and offense codes from the state of Minnesota.
nga	National Geospatial Agency
nlets	Nlets, The International Justice and Public Safety Information Sharing Network
nonauthoritative-code	Nonauthoritative codes for the direction of a person's pose in an image.
sar	Suspicious Activity Reporting
twpdes	Terrorist Watchlist Person Data Exchange Standard
ucr	Crime reporting codes from Uniform Crime Reporting.

Code Name	Description
unece_rec20-misc	Miscellaneous unit of measure codes from the United Nations Economic Commission for Europe Recommendation No. 20, Codes for Units of Measure used in International Trade.
usps_states	U.S. state and possession abbreviations from the U.S. Postal Service (USPS).
ut_offender-tracking-misc	Plea and military discharge codes from the Utah Offender Tracking Database, version 2.03.

3955

3956


NIEM