May 20, 2008

NIEM QUALITY ASSURANCE STRATEGY AND PLAN

VERSION 1.0

URI: http://reference.niem.gov/niem/guidance/quality-assurance-strategy-and-plan/1.0/

NATIONAL INFORMATION EXCHANGE MODEL | NIEM.gov

⟨NIEM⟩™

Change History

No.	Date	Reference: All, Page, Table, Figure, Paragraph	A = Add. M = Mod. D = Del.	Revised By	Change Description
1.0	05/20/2008	All	A	NTAC Members	Initial version

Contents

Figures

Tables

1 Introduction

After an initial period of rapid development and deployment, NIEM has matured to a stage of maintenance and continued growth. A key focus of project governance going forward is greater attention to the quality of the data model. Building on the experience and lessons learned from the work on NIEM 1.0, 2.0, and multiple versions of the GJXDM, we can identify what has worked in the past and identify areas of known difficulty to focus on in the future.

1.1 Purpose and Scope

The purpose of this paper is to define the goals, metrics, reviews, and procedures that will improve the quality of the NIEM data model and its artifacts. More specifically, this applies to:

- NIEM data content, which is officially distributed as a set of XML schemas for releases and is also made available via alternative formats, such as a developer's spreadsheet and database. These measures primarily address NIEM Core and domains but also address code tables in a limited fashion.[1]
- Descriptive metadata, which is captured and used to facilitate the search for and the understanding of NIEM data content. This includes keywords and usage information on properties.
- Content submissions, which are provided by domain representatives and NIEM users. These submissions include new component requests, additions or modifications to existing content, and new domain submissions.
- NIEM release schemas and release artifacts.

While certain tools and services play a role in assessing and performing quality assurance measures, quality assurance on any tool itself falls under general software development lifecycle procedures and is outside the scope of this paper. Quality assurance practices for IEPDs are also outside the scope of this paper.

The primary participants in a specific implementation of this strategy and plan include:

- The NIEM Business Architecture Committee (NBAC) and the NIEM Technical Architecture Committee (NTAC), which are responsible for the semantic and technical quality of the NIEM data model as a whole and that of NIEM Core in particular.
- The NIEM Program Management Office (PMO), which will provide the necessary governance, determine scheduling, and evaluate quality assurance measures for NIEM releases.

[1] Modifications to code tables generally fall outside the authority of the NIEM Program, except in cases of error in NIEM proxies of externally defined code tables. Quality assurance on code tables addresses issues that are within the jurisdiction of NIEM.

- Domain representatives, who are responsible for the content and quality of their domains.
- Content submitters, who will need to conform to higher standards of quality required for data submissions.

1.2 Approach

Quality assurance will be described in a layered approach consisting of data quality, harmonization quality, and operational quality.

- Data quality will address individual components.
- Harmonization quality will address the data model as a whole.
- Operational quality will address the inputs and outputs of the model.

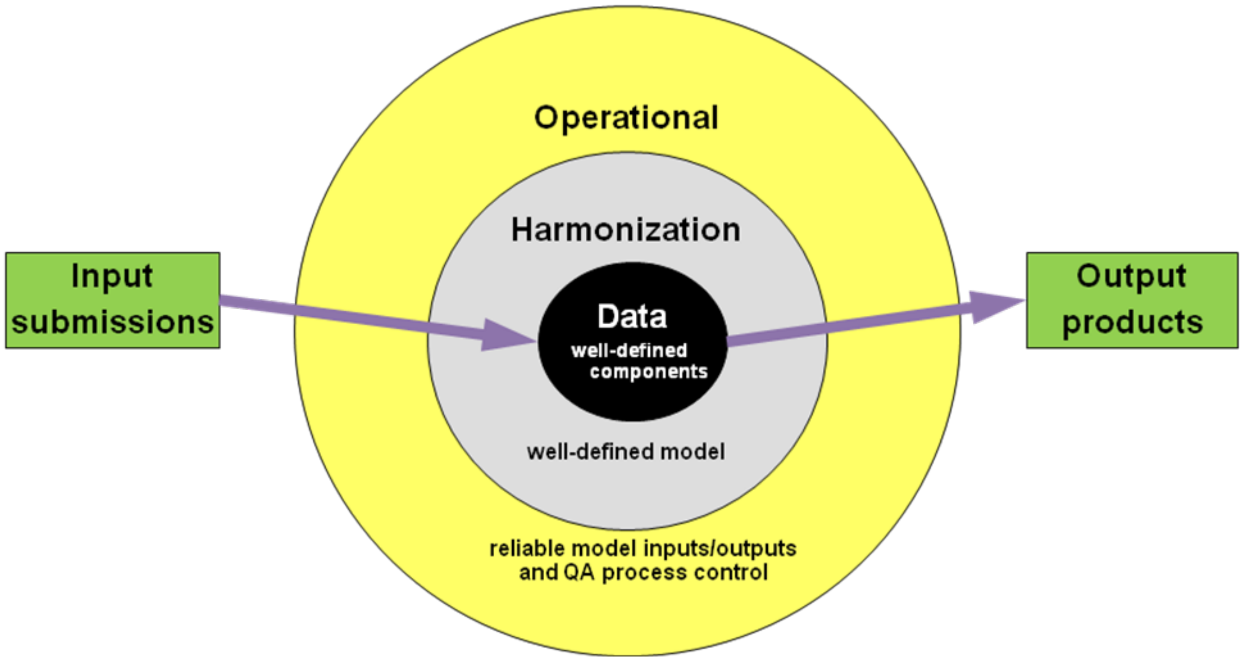

Figure 1: NIEM Quality Assurance Layers

This paper will describe each layer in a section that outlines its meaning, the ways by which it can be measured, the reviews that will be made to identify bugs and areas for improvement, and the procedures that will govern the process of improvement.

Quality can be very difficult to measure. In an attempt to quantify it, where possible statistics and metrics will be generated. These numbers will be snapshots of quality indicators of the model at specific release stages. As such, they may be used to identify areas where problems may exist and to identify incremental changes. When used with respect to previous snapshots, these numbers may be used to compare official releases, to compare a prerelease against a prior official release, and to observe general trends over time.

A lot of statistics and metrics will be presented in this paper. Some may be very easy to interpret, with clear and direct implications about the quality of the model. Others may be more difficult to interpret and may require collection and review over a period of time to determine what value, if any, they provide towards the evaluation and improvement of NIEM quality. Because of this, no effort will be made at present to assign weights or rankings to any of the statistics and metrics. If accumulated experience makes such an assignment reasonable in the future, it may then be possible to generalize an indication of the overall level of quality of the model.

Note that statistics and metrics may be available in multiple formats. Totals can be broken down by namespace, where relevant; percentages may be generated; and for a given count, its opposite may also be published (e.g., for the metric concerning the number of changed definitions, the number of unchanged definitions would also be made available). A caveat on statistics and metrics in general is that though some will be easy to collect and verify through tool support, others may be very difficult to gather and may not be complete. Difficult to gather and/or unreliable statistics will be denoted as such in this paper.

Finally, while there are many quality assurance metrics and checks for object-oriented design, few have been specifically adapted to XML Schema, outside of those that determine XML Schema validation. This paper will leverage some of the XML-specific metrics from SEARCH's[2] "Proposed NIEM Design Quality Metrics" and Joost Visser's "Structure Metrics for XML Schema."

1.3 Objective Versus Subjective Quality

When the data model is reviewed, quality control can be thought of as falling into two separate categories—objective quality and subjective quality. Objective quality is a yes-or-no decision—"Is a definition misspelled?" or "Does the name of the type end with the representation term 'Type'?" Subjective quality, on the other hand, involves the semantics of the model and personal or group judgment. It includes such questions as, "Is the definition meaningful?"; or "Should a property be added to a type as content or as a reference, or should it go into a new association instead?" These kinds of decisions may involve trade-offs and can depend on the experience of the person or persons involved.

Objective quality is easier to measure and test because it does not involve the semantic side of the model—it concerns the structural aspects that are often readily identifiable by automated tool support against a predefined rule set. Despite the fact that these problems are identifiable by tools, resolving them still requires personal intervention.

While the large majority of objective-quality problems are caught and corrected before a release is made official, a few can be overlooked by an evolving, but not yet fully matured, testing system. The large size of the data model and sometimes a large number of changes can also make these problems difficult to identify from human review. As overlooked bugs are identified, the testing system is updated to include new checks to handle those problems in the future.

[2] SEARCH, The National Consortium for Justice Information and Statistics.

Subjective quality of the model is harder to assess and verify than objective quality. Unlike with objective quality, tools usually can do little to facilitate this effort other than filtering information and laying out data for people to review manually. Gauging this kind of quality is difficult because there is no solid standard by which measurements can be made. A definition that was written and approved of by group consensus may later be flagged as incomplete or confusing by a user for reasons the group had not considered. The modeling of a component that is satisfactory at one point in time may later become outdated because of such reasons as the introduction of new NIEM techniques, the evolution of source requirements, or the addition of similar content in outside domains. Similarly, resolving these issues may require the use of best judgment and compromise.[3]

2 Data Quality

In NIEM, data quality refers to how well individual components are defined, both structurally and semantically. There are many aspects of a component that can be independently reviewed for quality, without regard for the rest of the model. This kind of fine-grain analysis accomplishes several objectives of NIEM quality assurance:

- Ensure that each component is well-defined and conforms to NIEM.
- Improve the overall consistency of the model as a whole by individually evaluating and improving the quality of each component.
- Confirm that each component is meaningful standing alone. Since all NIEM components are globally defined, each may be used independently throughout the model itself and within user-defined IEPDs.

The NIEM Naming and Design Rules (NDR) is a standard that defines, among other things, rules for component conformance. As such, it will provide the primary basis against which data quality will be judged.[4] Relevant NDR criteria, as well as additional criteria, are described below. Note that some aspects of this quality assessment can be done by automated validation checks—others will require human review and judgment.

2.1 Statistics and Metrics

Statistics and metrics at the data layer will provide a basis on which to measure the quality of the structure and semantics of individual components. New administrative metadata will need to be captured in some areas to support the generation of these numbers.

2.1.1 Components Without Definitions

Missing definitions indicate a lower degree of quality of the model.

[3] One or more solutions for an issue may be developed and posted to the Component Staging Area (CSA) as described in the NIEM High-Level Tool Architecture. The CSA will facilitate asynchronous group review and discussion.

[4] Because NIEM quality assurance relies on the NDR for rules defining component conformance, changes to the NDR may affect this paper.

This count does not represent a determination of the quality of the available definitions, only their presence or absence. This metric should be easy to generate from available information.

2.1.2 Components With Descriptive Metadata

Descriptive metadata in the model includes keywords, example content, and usage examples on properties and types. Currently, these fields are very sparsely populated. Increases in these numbers may indicate an improvement in the quality of the model.

These counts do not represent the quality of the descriptive metadata, only their presence or absence. This metric should be easy to generate from available information.

2.1.3 Code Table Currency

Most code tables in NIEM are proxies of non-XML code tables externally managed by independent authoritative sources.[5] Updates to the external code tables are not always caught and reflected in the corresponding NIEM code tables.

The quality of code tables, in this case measured by their currency, cannot be judged simply by the time of their last update. Some code tables that have remained unchanged for years may still be current, while other code tables may fall out of date within months. To measure the currency of code tables, it is necessary to actively look for available updates.

This metric requires new metadata—namely, the last date of search for updates, which should be easy to collect.

2.1.4 Components/Definitions That Have Been Vetted

A high number of components and definitions independently reviewed by the NBAC or its designees according to the guidelines described later in this section indicates a higher degree of quality.

This metric does not represent the number of components or definitions that have changed, only those that have been reviewed. This metric requires new metadata and easily collected reviewer feedback.

2.1.5 Components/Definitions That Have Changed Since the Last Release

Looking at this metric alone, it is difficult to tell what it means. While a high number of component or definition changes represents improvements made to the model and, thus, higher quality, a low number of changes could indicate that the existing components and definitions are well-formed and meaningful, which also would indicate higher quality.

To get a clearer understanding of this metric, it should be analyzed with respect to the number of components and definitions that have been vetted. A low number of changes has less

[5] There are a number of outstanding issues to deal with concerning code tables, largely focused on update strategies. An updated approach to code tables will likely be specified in an independent document. There is a possibility that decisions made in that document may have a bearing on what is discussed here, but it is considered unlikely because this metric focuses simply on looking at code table currency and not on a particular update strategy itself.

significance for the quality of the model if those component and definitions have not also been vetted. A low number of changes for those components and definitions that have been vetted is an indication of higher quality.

This metric should be easy to generate from available information.

2.1.6 Vetted Components/Definitions That Have Changed Since the Last Release

A high number of changes to vetted components or definitions indicates that improvements have been made to the model and that quality has increased. A low number of changes as a result of vetting may indicate that the quality was already high. It also demonstrates stability.

This metric requires new metadata and reviewer feedback but will be straightforward to collect.

2.1.7 Text Properties With at Least One Corresponding Code Property

Code sets are beneficial to information exchanges because they provide a well-defined way of specifying data that can be programmed to be readily interpreted by a computer—something that can be difficult or impossible to do with free text fields. While not relevant in all cases, a higher correspondence of code properties to text properties may indicate a higher degree of quality in the model.

These numbers do not represent the quality of the available code sets, only their presence or absence with regard to text properties. Collecting this metric may present some difficulties in that some corresponding text and code properties may not be represented as substitutions for a single semantic abstract property. It may be more difficult to identify corresponding pairs when they are contained separately within a type or within different types (e.g., because of specialization or augmentation). This metric will be generated from available information but may only represent an approximation.

2.1.8 Potential Code Properties

With the understanding that code sets can improve information exchanges, this metric will identify the number of places in the model for which reviewers deem a code set would be beneficial. A low number indicates that most relevant code sets have been added to the model.

This metric requires the addition of new metadata and easily collected reviewer feedback.

2.1.9 Unresolved NCCT Issues Relevant to the Data Model

A high number of unresolved issues directly relating to the data model in the NIEM Configuration Control Tool (NCCT) may indicate a lot of known problems and, thus, lower quality. A similar metric—the number of resolved issues resulting in changes in the model—indicates improvements and higher quality.

Both metrics should be easy to generate from available information.

2.1.10 Components' Change Counts

Components with few changes over successive releases can be considered relatively stable. While it cannot be directly inferred that they are accurately defined or modeled, it can be assumed that they sufficiently meet requirements.

Alternatively, frequent changes to components over successive releases indicate improvements to the model but also indicate instability.

This metric should be easy to generate from available information.

2.2 Automated Validation Checks

The following are useful automated tests to identify bugs in the model or areas that may need additional review.

Table 1: General Quality Checks

Test	Requirement Source
No duplication of property names, type names, enumerations, or namespace prefixes within a namespace.	XML Schema
Definitions must be provided for properties, types, enumerations, and namespaces.	NDR Rules 6-4 through 6-8
Spell checks on parsed property name and type name terms.	NDR Rule 8-1
No invalid characters in property or type names: only upper- and lower-case letters, digits, and hyphens are allowed.	NDR Rule 8-2
No invalid characters in definitions.	NDR Rule 5-46
Definitions of the component classes matching those described by [Table 1: Standard Opening Phrases] of the NDR should begin with the matching opening phrase.	NDR Rule 6-13
Spell checks on definitions.	NIEM QA[6]
No leading or trailing spaces in component names, definitions, descriptive metadata, facet values, or namespace prefixes.	NIEM QA
NIEM Core properties and types must be tagged as either "universal" or "core."	NCCT #156

[6] This document is the source of "NIEM QA" checks. These checks are not explicitly specified by NDR rules. They are based on best practices, experience, and common sense. Failing any of these checks does not necessarily constitute an error but indicates that the model will require further review.

Table 2: Property Quality Checks

Category	Test	Requirement Source
Naming		
	Association properties must use the “Association” representation term.	NDR Rule 8-31
	Augmentation properties must use the “Augmentation” representation term.	NDR Rule 8-32
	Metadata properties must use the “Metadata” representation term.	NDR Rule 8-33
	Roles must use the “RoleOf” property term.	NDR Rule 8-34
	Elements must be UpperCamelCase.	NDR Rules 8-5, 8-6
	Attributes must be lowerCamelCase.	NDR Rules 8-4, 8-6
	Property names must not use the term “Type” unless by special exception.	NIEM QA
	All attributes and elements with simple content must use a representation term defined in Appendix A (Property Representation Terms).	NDR Rules 8-18, 8-21
	Property names must not use the representation term “Reference” in the data model unless by special exception.	NIEM QA
Data Types		
	Element properties must be of a complex data type.	NIEM QA
	Attribute properties must be of a simple data type.	XML Schema
	A property may not have multiple data types.	XML Schema
	All properties that are not abstract must have a data type.	NDR Rules 5-9, 5-10
Invalid References		
	An augmentation may not be a reference.	NIEM QA
	An association may not be a reference.	NIEM QA
	Attributes may not be references.	XML Schema
	Abstract properties may not be references.	XML Schema
Substitution Groups		
	A property may be in, at most, one substitution group.	XML Schema
	A property in a substitution group must have the same data type or a type derived from the data type of the substitution group head.	XML Schema
Miscellaneous		
	Abstract properties should have at least one substitutable property available, unless by special exception.	NIEM QA
	Elements and attributes references from external standards must have definitions.	NDR Rules 6-64, 6-65

Table 3: Type Quality Checks

Category	Test	Requirement Source
Naming		
	Type name must use the term “Type” as a representation term only, unless by special exception.	NDR Rule 8-22
	Association types must use the representation term qualifier “Association”; all other types may not.	NDR Rule 8-26
	Augmentation types must use the representation term qualifier “Augmentation”; all other types may not.	NDR Rule 8-27
	Metadata types must use the representation term qualifier “Metadata”; all others may not.	NDR Rule 8-28
	Simple types must use the representation term qualifier “Simple” immediately preceding the representation term “Type.”	NDR Rule 8-23
	Code types and any types derived from code types must use the representation term qualifier “Code.”	NDR Rules 8-24, 2-25
	A complex type may have the same local name as an XML Schema namespace simple type only if it does not contain anything other than the attribute group structures:SimpleObjectAttributeGroup.	NDR Rule 8-0.9
Type Inheritance		
	Complex types with complex content (CCC) can only inherit from other CCC types.	XML Schema
	Complex types with simple content (CSC) can only inherit from other CSC types.	XML Schema
	Simple types can only inherit from other simple types.	XML Schema
	Association types must extend either another association type or nc:AssociationType.	NDR Rule 6-39
	Adapter types must not be extended or restricted.	NDR Rule 6-66
	No multiple type inheritance.	XML Schema
Subproperties		
	A type may not contain duplicate properties, including those inherited by extension.	NDR Rule 6-36
	A type should not contain properties that are otherwise accessible via augmentation or roles.	NIEM QA
	A complex type with complex content (CCC) must contain at least one property.	NIEM QA
	NIEM-conformant types may not contain external elements or attributes.	NIEM QA
	A type should contain a substitution head rather than a substitution group member, unless by requirement.	NIEM QA
	Property participants in the relationship defined by an association must be references.	NDR Rule 6-40
Miscellaneous		
	Adapter types must contain only external properties.	NDR Rule 6-63
	Complex types with complex content should have content.	NIEM QA

Category	Test	Requirement Source
	Simple types must have a base type.	NIEM QA
	A complex type must be either an object type, role type, association type, metadata type, augmentation type, or adapter type.	NDR Rule 6-35
	No use may be made of XML type xsd:anySimpleType (acts as a wild card).	NDR Rule 5-8

Table 4: Namespace Quality Checks

Test	Requirement Source
Namespaces must have a URI.	NIEM QA
Domain release numbers must not be lower than the NIEM Core release number.[7]	NIEM QA
Conformant namespace release numbers must not be lower than the structure's namespace release number.[8]	NIEM QA
No spaces in the release number, URI, or namespace prefix.	NIEM QA

Representation terms of properties will be checked against the data type or parent of the data type to ensure a match. Additionally, each of the types listed below and their subtypes will be reviewed to make sure each property of that type has the correct representation term. See Table 7 (ISO 11179 and NDR-defined Representation Terms) in Appendix A for representation term definitions.

Table 5: Property Representation Terms and Corresponding Data Types

Property Representation Term	Data or Base Type for Element	Data Type for Attribute
Text, Name	nc:TextType	xsd:string
Amount	nc:AmountType	xsd:decimal
Quantity	niem-xsd:integer, niem-xsd:nonNegativeInteger, niem-xsd:decimal, nc:QuantityType	xsd:integer, xsd:nonNegativeInteger, xsd:decimal
Numeric, Rate, Value	nc:NumericType, niem-xsd:nonNegativeInteger	xsd:nonNegativeInteger, xsd:decimal
Percent	nc:PercentageType	xsd:decimal
Measure	nc:MeasureType	xsd:decimal
Identification	nc:IdentificationType	n/a

[7] Dependent on pending decisions from the NIEM Version Architecture specification.

[8] Same as above.

Property Representation Term	Data or Base Type for Element	Data Type for Attribute
ID	niem-xsd:string, nc:TextType	xsd:string, xsd:integer
Indicator	niem-xsd:boolean	xsd:boolean
Date	nc:DateType	xsd:date
Image	nc:ImageType	n/a
Binary, Graphic, Picture, Sound, Video	nc:BinaryType	n/a
Status	nc:StatusType	n/a
Code	*CodeType	*CodeSimpleType
Association	nc:AssociationType	n/a
Augmentation	*AugmentationType	n/a
Metadata	*MetadataType	n/a

Additionally, XML Schema validation of the schemas produced from the data model provides further testing of the structural conformance of the model. This is discussed further in the operational quality layer.

2.3 Manual Quality Inspections

Some of the biggest challenges in the data layer involve applying consistent NIEM modeling to components that are submitted by different users and sources at different times, providing meaningful definitions and identifying existing modeling problems. The following reviews and inspections will be done to evaluate and improve the syntactic, semantic, and structural quality of the model. Note that new administrative metadata will be captured in the model to record information about and results of the reviews.

2.3.1 Component-Naming Review

The format of component names in NIEM is specified by the NDR, based on name subparts defined in Part 5 of the ISO 11179 standard. These subparts are:

- An object class term, which represents the concept or entity that the component describes.
- A property term, which represents the specific characteristic the component describes.
- A representation term, which describes the general representation form of the component.
- Optional qualifier terms, which may further clarify the object class term, property term, and/or the representation term.

An example of these component name subparts is represented in the figure below.

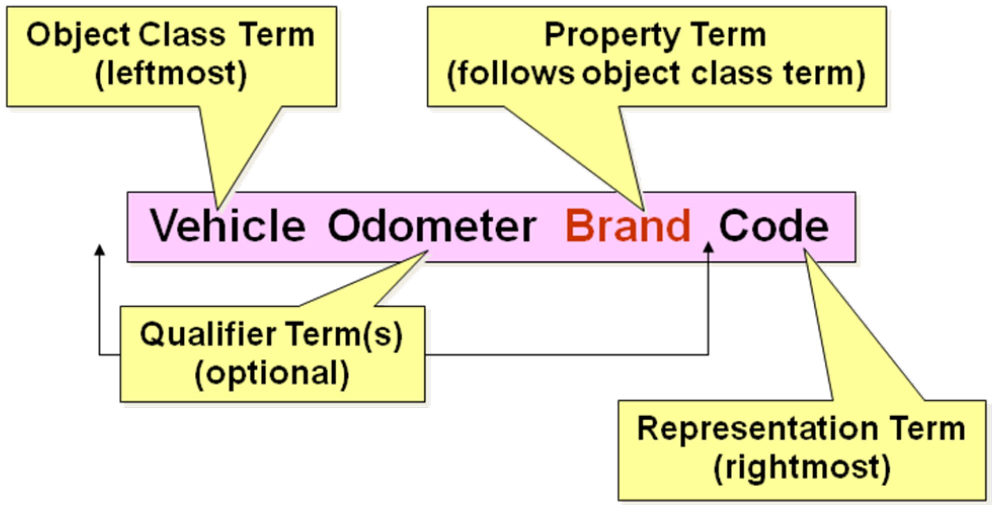
Object Class Term (leftmost)
Property Term (follows object class term)
Vehicle Odometer Brand Code
Qualifier Term(s) (optional)
Representation Term (rightmost)

Figure 2: Component Naming

Additional rules and special exceptions are defined by the NDR. Those rules relevant to quality assurance have been extracted and appear in Table 8 (NDR Component Naming Rules), Appendix B.

Component names will be reviewed to ensure that they follow ISO 11179 and NDR naming rules and that they are clear and meaningful. Furthermore, component names will be broken down into their ISO 11179, Part 5 subparts at this stage of the review to provide metrics and a foundation for name-based analysis and review described later in this paper.

2.3.2 Definition Review

Definitions will be reviewed against ISO 11179, Part 4 and NDR criteria. These criteria have been merged and are listed in the table below.

Table 6: Definition Review Guidelines

Category	Test	Requirement Source
Definitions must...		
	be stated in the singular	ISO 11179, Part 4
	state what the concept is, not only what it is not	ISO 11179, Part 4
	be stated as a descriptive phrase or sentence(s)	ISO 11179, Part 4
	contain only commonly understood abbreviations	ISO 11179, Part 4
	be expressed without embedding definitions of other data or underlying concepts	ISO 11179, Part 4
	have exactly one semantic meaning	NDR Rule 6-10
Definitions should...		
	state the essential meaning of the concept	ISO 11179, Part 4
	be precise and unambiguous	ISO 11179, Part 4

Category	Test	Requirement Source
	be concise	ISO 11179, Part 4
	be able to stand alone	ISO 11179, Part 4
	be expressed without embedding rationale, functional usage, or procedural information	ISO 11179, Part 4
	avoid circular reasoning	ISO 11179, Part 4
	use the same terminology and consistent logical structure for related definitions	ISO 11179, Part 4
	be appropriate for the type of metadata being defined	ISO 11179, Part 4
	reuse synonyms and terms from the component name only if it improves clarity	NDR Rule 6-9
Definitions may not...		
	redefine the object class for components that represent properties or subparts of that entity class	NDR Rule 6-11
	contain explicit representational or data typing information such as number of characters, type of characters, etc., unless the very nature of the component can be described only by such information.	NDR Rule 6-12

2.3.3 Descriptive Metadata Review

Existing keywords, example instances, and usage information will be reviewed for accuracy and applicability. New ones will be solicited from reviewers and domains.

2.3.4 Modeling Review

Components will be reviewed in the following areas to ensure that each is well-defined. The primary focus of these reviews is to look for obvious errors or oversights in modeling and to ensure the cohesion of each component—the semantic consistency of the name, definition, and directly related components (e.g., data type, subproperties, and parent type).

2.3.4.1 Property-Modeling Review

- The data type of a property should represent the kind of concept, entity, or object the property semantically defines.
- For each type that contains the reviewed property, the property should be a characteristic of that type.
- Stand-alone properties (those that are not contained within a NIEM type) should be intentional. Although not a definitive rule, likely errors will be properties that have both an ISO 11179 object class term and a property term.

  For example, if property "PersonSSNIdentification" did not appear under a type (it does), the reviewer should look at the object class term of the name ("Person") and determine whether this property is a characteristic of a person object in NIEM.

Alternatively, actual stand-alone justice property "Misdemeanor" is not a likely oversight. Note that of these two examples, only the first has an ISO 11179 property term in the name.

Note that associations and substitution group members (not substitution group heads) are not candidates for this particular review—both are designed to be stand-alone properties.

- Substitution group members should each provide a more specialized meaning or representation of their substitution group heads; substitution group heads should be a semantic generalization of their members.
- The existence of attributes within the model is discouraged and should occur only when the usage of elements in its stead results in coherence problems.

2.3.4.2 Type-Modeling Review

- A type uniquely represents an object type, a role type, an association type, an augmentation type, an adapter type, or a metadata type.
- All subproperties of a type are characteristics of the concept, entity, or object the type represents.
- Subproperties that directly participate in the relationship defined by an association type should be references.
- Each property of the reviewed type maintains semantic consistency.
- Inheritance, augmentations, and roles are applied in accordance with the rules and guidelines specified by the NDR.
- Facets on a type are unique and well-defined.

2.4 Quality Assurance Process Overview

For quality assurance of the data layer, statistics and metrics will be generated for two distinct purposes—to identify areas for improvement during prerelease development and maintenance on the model, and to create benchmarks for independent and comparative quality assessment at the time of release.

Objective quality validation checks will be run at each stage of prerelease and before the official release. It is not expected that alpha releases will pass all of these validation checks, but special effort should be made to correct known bugs before a beta is released.

While most identified bugs will be fixed before a final release, it is possible that some will remain. For example, missing definitions in code tables may not be able to be resolved, and complex remodeling efforts may require more time or resources than available. It will be up to the NBAC to make special exceptions on a case-by-case basis with the goal of providing the best possible value to the NIEM model and user community. When components that have been identified with temporary or long-term shortcomings are nonetheless deemed to be of merit, the focus on quality assurance and the visibility of statistics and metrics should not prevent their inclusion in the model. Note that these exceptions will be documented in the model to ensure that reviewers are aware of the special circumstances.

Subjective quality reviews should be run on an ongoing basis. As NIEM continues to evolve, rules, guidelines, and techniques may change. User requirements will also continue to evolve. Subsequent reviews and vetting of components may not be high priority and may require less time and effort than the initial reviews, but they will help to ensure a high standard of consistency and will serve to keep the model up to date.

The highest priority for subjective reviews should be placed on all new components, modified components, and those components from NIEM Core that have not previously been reviewed or vetted.

Despite all of the rules and guidelines that can be made available, the reliability of subjective reviews will depend greatly on the review participants themselves, their experience with NIEM, and their knowledge of relevant subject matter. In recognition of this, the NBAC must give special consideration in choosing the designated reviewers and review panels.

Finally, reviews and bug reports may identify errors in code tables outside the jurisdiction of NIEM. Contacts should be established, where possible, so that relevant feedback may be provided to the appropriate code table authoritative sources.

3 Harmonization Quality

Users need to ensure that individual components are well-defined in the data quality layer. Harmonization quality in NIEM refers to the cohesiveness, integrity, and consistency of the model as a whole. Because of the large size of the model and the relatively independent nature of the domains, semantic overlap and isolated modeling of related components presents a significant challenge. A comprehensive approach to the semantic review of the model has the following objectives:

- Each component in the model should define a unique meaning, concept, or representation.
- Components should be defined in the most fitting namespace, to be discussed below.
- Related components need not all be defined in the same namespace or at the same time, but they should be defined or refactored in such a way that eliminates duplication and promotes consistency.

Note that this layer of quality primarily deals with semantic consistency and thus falls under subjective review. It will have little support from tool automation. The addition of an ontology layer to NIEM could have a considerable impact on harmonization quality, but there are no definitive plans for such an addition at this time.

3.1 Statistics and Metrics

Harmonization layer statistics and metrics will provide a basis on which to measure the cohesive quality of the model.

3.1.1 Class Terms From Component Names

One step in the manual review from the data layer was to parse each component name into its individual ISO 11179, Part 5 subparts. This work will be used in part to generate statistics showing the usage of class terms within the model and by namespace.

These statistics may not have a direct implication for the quality of the model but are informative nonetheless. They will loosely serve to identify the general objects defined in the model, as well as their size. In addition, they may help to identify those objects that are defined across multiple namespaces and should later provide a basis for review of semantic overlap and duplication.

As mentioned, this will require the breakdown of component names into ISO 11179, Part 5 subparts. This is not likely to be an easy or quickly completed task and will affect the release of these numbers. No additional metadata will need to be captured, and once the name subparts have been identified and collected, tools can generate the statistics.

3.1.2 Component Dependencies

This metric will show how closely related components are to other components.[9] While component dependency and reuse is a necessary and normal part of NIEM data modeling, high levels of dependencies may indicate areas with greater effects on harmonization efforts. They may also indicate below- or above-average outliers with a need for further review.

This metric is easily generated from data currently captured in the model.

3.1.3 Type Hierarchy

This metric[10] will show the depth and breadth of the type inheritance hierarchy in NIEM. While having no direct implications towards the quality of the model, areas of deep inheritance may indicate outliers with a need for further review.

This metric is easily generated from data currently captured in the model.

3.1.4 Direct Interdomain Dependencies

This will show the amount of component reuse across domains and which domains are affected by dependencies.[11] This metric[12] serves more for informational purposes and benchmarks rather

[9] Specific means of calculating this metric will be reviewed by the NTAC. Proposals from SEARCH's "Proposed NIEM Design Quality Metrics" (type coupling) and Joost Visser's "Structure Metrics for XML Schema" (coupling, instability) will be part of this consideration.

[10] Specific means of calculating this metric will be reviewed by the NTAC. Proposals from SEARCH's "Proposed NIEM Design Quality Metrics" (type inheritance).

[11] The current leaning of the NTAC with respect to the NIEM Version Architecture specification is that interdomain dependencies should be allowed. This has not yet been decided conclusively.

[12] Specific means of calculating this metric will be reviewed by the NTAC. Proposals from SEARCH's "Proposed NIEM Design Quality Metrics" (namespace coupling) and Joost Visser's "Structure Metrics for XML Schema" (coupling, instability) will be part of this consideration.

than giving a strong implication on the quality of the model, although high numbers may suggest the need for refactoring and harmonization.

This metric is easily generated from data currently captured in the model.

3.1.5 Indirect Interdomain Dependencies

This will show the estimated cost of refactoring domain components with dependencies up into NIEM Core. For example, if another domain is dependent upon the justice property "Arrest," there is an indirect dependency on the subproperties of j:ArrestType, their subproperties, and so forth. In addition, there may be indirect dependencies or effects on things, such as parent types, facets, substitution groups, and related associations.

This metric has little bearing on the quality of the model. Instead, it provides useful information for harmonization review and evaluating the merit of eliminating dependencies versus retaining them on a case-by-case basis. This is discussed further later in this section. This metric is easily generated from data currently captured in the model.

3.1.6 Interdomain Dependencies by Component

This will show the components with one or more interdomain dependencies. Higher numbers of dependencies on a component may indicate a need for refactoring.

This metric is easily generated from data currently captured in the model.

3.1.7 Namespace Cohesion

This will show how closely related components in a namespace are.[13] A higher degree of cohesion may indicate that namespaces are broken down based on logical groupings of components. This, in turn, may imply higher quality of the model.

Note that this metric is independent of the effects of authoritative sources over logical breakdowns of namespaces. This metric is easily generated from data currently captured in the model.

3.1.8 NIEM Components That Are Extended in Registered IEPDs[14]

This will show where additional real-world requirements exist for NIEM components. Components that are extended by multiple IEPDs may have greater need for review and harmonization, as discussed later in this section.

[13] Specific means of calculating this metric will be reviewed by the NTAC. Proposals from SEARCH's "Proposed NIEM Design Quality Metrics" (namespace cohesion) and Joost Visser's "Structure Metrics for XML Schema" (coherence) will be part of this consideration.

[14] It would be impossible to locate every IEPD that is created. This metric will look at extensions in IEPDs that have been posted for sharing and reuse in the federated IEPD registry/repository, as described by the NIEM High-Level Tool Architecture.

The generation of this metric involves accessing properly formatted NIEM document and extension schemas from certain known IEPD registries. It will require no additional metadata. [15]

Note that this metric is used to judge the quality of NIEM components. It is not designed to look at or provide feedback on the quality of IEPD extensions.

3.1.9 Components That Have Been Vetted for Harmonization

A high number of components that have been reviewed and vetted for harmonization quality in the current release by the NBAC or its designees indicates a higher degree of quality for the model in terms of both modeling and currency. Those components that have been reviewed only in previous versions still represent higher quality, but they do run the risk of being outdated to some degree with respect to the latest techniques and data requirements.

This metric does not indicate the number of components that have been changed, only the number of components that have been reviewed. This metric requires new metadata and reviewer feedback but will be simple to collect.

3.1.10 Components That Are in the Component Staging Area (CSA)

A high number of components in the Component Staging Area (CSA)[16] at the time of release indicates a larger number of unresolved modeling and vetting efforts. A lower count could indicate either that outstanding issues have been resolved or simply a lack of utilization of the staging facility.

This count involves a staging area that is planned but does not yet exist. The development of the CSA and the population of content by content submitters and model reviewers will directly affect the introduction of this metric and its worth.

3.2 Automated Validation Checks

Since harmonization quality deals with the semantics of the model, few automated checks are available. Those that are used will only be able to apply straightforward rules to structural aspects of the model in order to approximate interpretation of semantics.

3.2.1 Component Name Duplication

The simplest of harmonization checks, duplication of property and type names, will be identified by tool support. This will be reviewed to determine whether the duplication (1) results in semantic overlap; (2) is the result of one or more poorly named components; or (3) is valid within the defining namespaces.

[15] More specifically, the generation of this metric will involve accessing NIEM document and extension schemas from the federated IEPD registry/repository, as described by the NIEM High-Level Tool Architecture.

[16] The CSA is described in greater detail in the NIEM High-Level Tool Architecture.

3.2.2 Component Name Similarities

String comparison algorithms will be run to identify closely related component names. As with component name duplication, results will be reviewed to determine whether the similarity is due to semantic overlap or poor naming or is actually valid.

Note that there is a very strong likelihood of excessive false hits if results from corresponding text/code properties (which are very similarly named) are not suppressed.

3.3 Manual Quality Inspections

Manual reviews will result in the bulk of harmonization improvements.

3.3.1 Review of Components With Shared Class Terms in the Name

The ISO 11179, Part 5 class term in a component name indicates the real-world concept or entity that the component represents. There is a greater potential for duplication or overlap in the model when different domains define or contribute to the same real-world concept or entity.

3.3.2 Review of Specializations, Augmentations, and Roles by Type

Recall that specializations define additional content for a mutually exclusive subclass of a type. Augmentations define additional content within a domain that can be applied to the base type and roles define additional content for a type within a specific context or usage.

In any case, when a type has multiple sources of additional content, there exists a potential for semantic overlap and harmonization.

3.3.3 Review of Multiple Corresponding Code Properties

Multiple code sets for a given property may provide a degree of flexibility for users. Whether this is a pro or a con for the model depends on point of view and the specific code sets. In some cases, multiple code sets define semantically unique sets of data values and provide value to the model. In other cases, code sets may overlap or provide the same codes in different formats (e.g., the same information represented as both alphabetic and numeric codes), indicating a lack of harmonization. Multiple corresponding code sets will be reviewed to see whether the potential for harmonization exists.

3.3.4 Component Harvesting From Registered IEPDs

IEPDs are definitions for real-world usage of NIEM components. Extensions in IEPDs of NIEM components indicate that those components do not meet all requirements. While it is not expected that all extensions will or even should be added to NIEM, these are requirements that should be reviewed for merit in inclusion within the model.

3.3.5 Namespace Cohesion Review

As part of this review, components within a namespace will be grouped together based on their normal interdependencies—a property is of a data type, a type contains properties and may extend another type, associations relate multiple properties together, and so forth. Programmatically dividing namespaces into these closely bound blocks of components will show

how many disparate groups of data exist. Reviewers should look at this breakdown of components with regard to their authoritative sources to determine whether they should be factored into separate namespaces.

3.3.6 Domain Dependency Review

Dependencies between domains will be identified. These will be reviewed in conjunction with the indirect dependency metric to determine whether the components in question should be refactored and possibly moved up to NIEM Core. Some general points to consider in making this assessment include:

- If a domain is simply reusing another domain's component with little or no customization, there is little refactoring to be done and there may be no merit in moving the component up to NIEM Core.
- If the component is heavily reused or is the basis for a lot of customization, there may exist a need for harmonization and refactoring. There may also be reason to move the component to NIEM Core.
- The authoritative source may play a role in determining to which namespace the component belongs.
- The full effects of refactoring the dependencies should be evaluated. The indirect dependency metric provides a good start in determining the cost of moving a component to a different namespace. Depending on the specific context, large-scale move operations may not be worth the elimination of a dependency.

There are no hard-and-fast rules for when domain dependencies should be eliminated or maintained.[17] Judgment should be used to determine a solution that balances the simplicity and usability of the model with both precise application of modeling rules and the fulfillment of domain requirements.

3.3.7 Dependency Modification Harmonization

Because changes to NIEM, even in domains, are not isolated events, it is appropriate for those affected to have the chance to review changes that will affect them and to adjust as necessary. Changes to NIEM Core and domains both have the potential of affecting other domains and IEPDs. Dependent domains and registered IEPDs will be identified so that their representatives may have the chance to evaluate the modifications and react if needed.[18]

[17] Guidelines for domain dependencies are detailed in the NIEM Version Architecture specification, which is still under discussion.

[18] The NIEM Version Architecture specification will provide additional details.

3.3.8 Content Reviews and Vetting

After looking in likely areas for semantic overlap, users should conduct a general review and vetting of NIEM Core and domain content. This will help to identify nonobvious problems and will help to ensure consistency of modeling throughout the model.

3.3.9 Namespace Content Overviews

The size of the data model and relatively independent domains can make harmonization difficult. It follows that content submitters may have trouble grasping what content is available to them in domains and even in NIEM Core itself. It will be even more challenging for users and submitters new to NIEM. As the model grows, so will this problem.

To reduce the learning curve and to promote general awareness of namespace content, lists of high-level objects (entities, activities, concepts, things) and their definitions should be created and maintained for NIEM Core, domains, and external standards. These high-level object lists will serve as a table of contents of sorts for NIEM namespaces.

This will facilitate harmonization in two respects—identifying existing overlap in the model and helping to avoid overlap in the future by improving content awareness. Additionally, this provides a very basic top-down approach to support the introduction of the model to new users.

3.4 Quality Assurance Process Overview

As with the data layer, statistics and metrics for the harmonization layer will be generated for problem identification and general maintenance in prerelease stages and for quality benchmarks and comparative analysis at release.

Reviews should be rerun on a periodic basis as necessary maintenance on the model, but the introduction of a new domain or modified domain content should have higher priority for review, as should existing content that has never been vetted.

Review panels may benefit from the inclusion of members from different domains with a range of relevant subject-matter knowledge.

4 Operational Quality

The purpose of this layer in NIEM is to ensure high-quality inputs and outputs to the data model. Despite efforts towards stability, NIEM will always remain a somewhat fluid model, incorporating new domains and changing requirements, producing new releases, and adapting as necessary to support the user community. As such, it is important to the quality of the model that the content submissions and other inputs be well-defined and especially important to the users that the releases and other outputs be reliable.

Inputs to the model include new content submissions, change requests for existing content, bug reports, and independent domain updates.[19] Outputs of the model include prerelease and release versions of schema distributions, documentation, and associated artifacts, such as the model database.

4.1 Statistics and Metrics

Statistics and metrics at this layer of quality control will be used: (1) to evaluate the quality of inputs before they are added to the model; (2) to assess the quality of prereleases before a final release is generated; (3) to generate benchmarks for evaluation of the quality of the release and to compare it against other releases; and (4) for general informational purposes.

4.1.1 Number of New Domains in a Release

This is an informative metric that shows one way to measure the growth of the data model. This count is easily generated from data currently captured.

4.1.2 Number of Changes to the Model

This metric will be broken down in several ways to show a clearer idea of the kinds of changes that have been applied:

- Number of remodeling changes
- Number of definition changes
- Number of changed components with interdomain dependencies
- Number of deletions
- Number of deprecations
- Number of additions
- Number of modified domains

While most of these numbers are generally informative, the number of remodeling changes, in particular, has the greatest bearing on the stability of the model. Though remodeling can generally be assumed to be an improvement to quality, it also has greater impact on the user community.

4.1.3 Number of Registered IEPDs Affected by Release Changes

This is an informative metric that will show the impact of model changes on real-world usage of NIEM. Note that this count represents only a subset of existing IEPDs.

[19] Although currently in discussion, the current approach to independent domain updates appears to be independent staging of additive changes for a domain. Before a NIEM release, these staged changes would be incorporated and harmonized into the data model. See the NIEM Version Architecture specification for additional detail.

The generation of this metric involves accessing IEPD metadata from designated registries.[20] It will require no additional metadata.

4.1.4 Additional IEPD Metrics

Additional informative metrics outlining usage of NIEM include:

- Number of registered IEPDs
- Number of ongoing IEPD efforts
- Number of IEPDs being used in real exchanges

The numbers show general interest and actual usage of NIEM. Note that these counts will represent only a subset of IEPDs in existence. The generation of these metrics involves accessing registered IEPD metadata.[21]

4.1.5 Number of Independent Schema Distribution and Model Artifact Reviews

A higher number of reviews of the schema distribution and model artifacts indicates greater quality in the products. These reviews are discussed in more detail below. The reliability of this metric depends on the quality and depth of the reviews themselves. Reviewer feedback will be required to generate this metric.

4.2 Input Reviews and Support

The current process for input collection and review is loosely defined. Following a more comprehensive and structured process for input review would increase the quality of the data being added to the model and, in turn, increase the quality of the model itself.

4.2.1 Input Collection

Currently, new domains or sizable content additions are collected via the Component Mapping Template (CMT), a specially formatted spreadsheet designed several years ago to capture source requirements and their mappings to existing or proposed NIEM components. Although the CMT is able to meet basic needs, shortcomings have been identified with the format and new NIEM techniques have been introduced that are not well-represented (e.g., substitution groups).

Using CMT implementer feedback, a new specification will be developed to capture content additions or modifications.[22] Additionally, new content and modification changes are captured by textual description and file attachments in the NCCT.

[20] More specifically, the generation of this metric will involve accessing NIEM subset wantlists from the federated IEPD registry/repository, as described by the NIEM High-Level Tool Architecture. Grant funding requirements may include a directive to publish IEPDs to this federated registry/repository for the purposes of quality control and impact assessment.

[21] Specifically, these metrics involve accessing metadata defined by the 3.1.2F (IEPD Metadata) interface and registered in federated IEPD registry/repository, both described by the NIEM High-Level Tool Architecture.

Future avenues of input to the data model include domain-independent updates [23]and the harvesting of NIEM extensions from registered IEPDs.[24]

4.2.2 Input Validation and Review

Currently, there are no rigorous validation reviews of input to the data model. When new content and change requests are submitted via the CMT, the information is manually reviewed to determine whether most of the NIEM modeling rules have been applied correctly and whether the CMT formatting guidelines have been followed or are approximated enough to manipulate the data into a loadable format. Requests submitted via the NCCT are reviewed by the NBAC body or by domain representatives.

Inputs submitted via the CMT that deviate from the NDR and prescribed formatting rules result in additional processing, increased time required to integrate content into the model, and a greater likelihood for misinterpretation of intent. The same consequences are to be expected for any format that is to be used. A service[25] that would check inputs for NIEM conformance[26] and format adherence would raise the quality of the input and support scalability of the model as new domains and content are added.

For purposes of data quality and harmonization, new input should be reviewed and vetted before being added to the model.[27] The validation checks and reviews identified in the earlier sections should be run to ensure the quality of this data. Additionally, the creation of namespace content overviews by new domains and sizable content submitters would make the identification of semantic overlap easier, especially when time is limited for review.

4.3 Output Review and Support

The current process for reviewing data model outputs includes running some of the validation checks outlined in the above sections, looking for bugs by running distribution schemas against XML Schema validators, and distributing a series of prereleases for review by NIEM governance and user communities. This process can be expanded and better defined to promote reliability of releases and model artifacts.

[22] This new specification is defined by the 5.3.1F (CSA Change Submission) interface in the NIEM High-Level Tool Architecture.

[23] The approach for domain-independent updates is currently in discussion. See the NIEM Version Architecture specification for more detail.

[24] This will require accessing extension schemas from IEPDs registered in the federated IEPD registry/repository, as described by the NIEM High-Level Tool Architecture.

[25] For the 5.3.1F (CSA Change Submission) interface (the CMT replacement), this service would be the 5.3.2S (CSA Change Submission WS) as defined by the NIEM High-Level Tool Architecture.

[26] This would be performed by the 3.9S (NIEM Validation WS)—a Web service that will validate schemas for NIEM conformance, as specified by the NIEM High-Level Tool Architecture.

[27] The Component Staging Area (CSA) would be useful in facilitating this process by allowing content to be staged and vetted before being added to the model. See the NIEM High-Level Tool Architecture for more detail.

4.3.1 Release Strategy

The NIEM release strategy was designed to correspond with standard practices of software lifecycles. The stages of NIEM prerelease are:

- Alpha—New content is added, all pending major changes are applied
- Beta—Bug fixes, cleanup, general improvements, testing period
- Release candidate—Minor bug fixes, tweaks

While the NIEM PMO may approve certain exceptions, close adherence to this strategy will ensure adequate time for testing and issue resolution.

In the past, the release process usually allowed for only a one- to two-week period of review after each prerelease was published. The frequency and limited time period of reviews did not support rigorous evaluation of either the applied changes or the integrity of the model as a whole. Future major releases of NIEM will have the benefit of an extended beta testing period. A time frame of up to six months may be allotted for this beta release to allow more in-depth community review and pilot project testing, which will provide technical as well as business requirement feedback.

4.3.2 Increased Visibility of Changes

When publishing prerelease distributions for review, it will be important for users to clearly identify (1) what has changed since the last prerelease and (2) a culmination of all changes since the last official release. Identifying these changes[28] will make focused reviews easier and more effective.

4.3.3 Reviews

4.3.3.1 Tool-Based Reviews

Before every release, a series of tool-based reviews will be run to assess the quality of the model. Reports will be generated from the following and will be distributed to the PMO, NBAC, and NTAC:

- Automated validation checks to assess objective data and semantic quality.
- Schematron rules to test for NDR conformance.
- XMLSpy validation of distribution schemas to test for XML Schema conformance.
- Xerces validation of a specially generated instance that represents every component from the model to test for XML Schema conformance.
- Statistics and metrics to provide a basis for the evaluation of the quality of the model.

[28] Described in greater detail in the NIEM Version Architecture specification.

Issues or problems identified by these tests should be resolved before a distribution is released. If metrics indicate that the quality of the model has decreased since the last release, NIEM governance may decide that additional improvements must be made.

4.3.3.2 General Reviews

General reviews of the data model and schemas occur for every distribution at each phase of the release cycle. These reviews involve members of NIEM governance and user communities evaluating the integrity of the content and whether the modeling fulfills associated business requirements. This kind of review goes beyond the objective evaluation provided by tool testing and actually looks at the semantics of the data.

Using the evaluation guidelines outlined above in the data and harmonization-layer sections, the NBAC or its designees should conduct a thorough review of the entire model prior to the next major release following NIEM 2.0. Subsequently, such detailed review of the entire model should not be needed as long as close attention is paid to changes and new content.

4.3.3.3 Usage of NIEM

Casual and even detailed visual review has not been able to identify all problems in NIEM in the past. Oversights can occur if, for instance, problems are obscured by the size of the model or reviewers are unaware of how components will actually be used. For this reason, it is important to test NIEM prereleases in actual projects that can provide technical as well as business requirement feedback.

Previously, feedback from real-world NIEM usage occurred only after releases were officially distributed. With the elongation of the beta review period in the release strategy, pilot projects with a wider array of development tools will be able to implement beta distributions, utilize model artifacts, and provide feedback that may be incorporated into the official release.

4.3.4 Available Support

The NIEM Configuration Control Tool (NCCT) provides a place to aggregate feedback and issues with the model, as well as log comments and document resolutions.

The NIEM help desk serves as valuable support for users with questions or problems—additionally, help desk staff feed unresolved issues into the NCCT for future action.

The projected Component Staging Area (CSA)[29] proposes to provide a place to share and vet changes to the NIEM data model prior to making actual changes, to identify dependencies, and to facilitate the review process. With the increased number of changes and reviews that are expected and with review participants located across the country, a tool that asynchronously facilitates a structured review process should prove to be of significant assistance in improving the quality of the model.

[29] Described in the NIEM High-Level Tool Architecture.

4.3.5 Quality Assurance Disclosure

A quality assurance report will be distributed with each release. This report will document checks that have been made on the current release, checks that have run on prior releases but not on the current release, and checks that have been identified in the NIEM quality assurance strategy but have not yet been implemented. This report will also contain a list of special exceptions that would otherwise appear to be overlooked bugs.

5 Summary

With early efforts focused primarily on the rapid development of NIEM, attention is now shifting to the quality and reliability of the model. This can be addressed at three levels—the data layer to ensure well-defined components, the harmonization layer to ensure a well-defined model, and the operational layer to ensure well-defined inputs and quality outputs.

Data model quality can be difficult to measure. Statistics and metrics can be utilized to identify areas of low quality, as well as to provide quality benchmarks that can be analyzed independently and also compared against those from other releases to observe trends.

A variety of automated checks and manual reviews are necessary to identify bugs and to improve general quality. Automated checks are best suited to test for objective quality problems—those that are identified by straightforward and structural-based rules. Evaluations requiring semantic judgment are left for manual review.

Because NIEM and its user community continue to grow, quality assurance will be an ongoing effort. Old requirements evolve and new ones will be added. A thorough and comprehensive approach to quality assurance will result in better improvements of the model and increased reliability and consistency.

5.1 Recommendations and Next Steps

It is the recommendation of the NTAC to implement the measures presented in this paper to improve the quality of NIEM by following a clear and well-defined approach. The emphasis on quality assurance and the time that it will require will have a direct bearing on release planning and scheduling.

The next step for NIEM quality assurance is to draft an implementation plan. The implementation plan will define a specific course of action for implementing the measures that have been outlined only at a high level in this paper. This plan will:

1. Prioritize manual checks and reviews to ensure the most effective use of available resources.
2. Describe calculations that will be used to generate statistics and metrics.
3. Define the processes and steps that will be necessary for implementation.
4. Address specific tools as defined by the NIEM High-Level Tool Architecture.

Appendix A: Property Representation Terms

Table 7: ISO 11179 and NDR-Defined Representation Terms

Primary Representation Term	Secondary Representation Term	Definition
Amount		A number of monetary units specified in a currency where the unit of currency is explicit or implied.
Binary Object		A set of finite-length sequences of binary octets.
	Graphic	A diagram, a graph, mathematical curves, or similar representation.
	Picture	A visual representation of a person, object, or scene.
	Sound	A representation for audio.
	Video	A motion picture representation; may include audio encoded within.
Code		A character string (letters, figures, or symbols) that for brevity, language independence, or precision represents a definitive value of an attribute.
DateTime		A particular point in the progression of time, together with relevant supplementary information.
	Date	A particular day, month, and year in the Gregorian calendar.
	Time	A particular point in the progression of time within an unspecified 24-hour day.
ID		A character string to identify and distinguish uniquely one instance of an object in an identification schema from all other objects in the same schema, together with relevant supplementary information.
	URI (Source: NDR)	A string of characters used to identify (or name) a resource. The main purpose of this identifier is to enable interaction with representations of the resource over a network, typically the World Wide Web, using specific protocols. A URI is either a Uniform Resource Locator (URL) or a Uniform Resource Name (URN). The specific syntax for each is defined by [RFC3986].
Indicator		A list of two mutually exclusive Boolean values that express the only possible states of a property.
Measure		A numeric value determined by measuring an object along with the specified unit of measure.
Numeric		Numeric information that is assigned or determined by calculation, counting, or sequencing. It does not require a unit of quantity or a unit of measure.
	Value	A result of a calculation.
	Rate	A representation of a ratio where the two units are not included.
	Percent	A representation of a ratio in which the two units are the same.

Primary Representation Term	Secondary Representation Term	Definition
Quantity		A counted number of nonmonetary units, possibly including fractions.
Text		A character string (i.e., a finite sequence of characters) generally in the form of words of a language.
	Name	A word or phrase that constitutes the distinctive designation of a person, place, thing, or concept.

Appendix B: Component-Naming Rules

Table 8: NDR Component-Naming Rules

Applies To	Component Name Requirement	Requirement Source
Characters in Name		
	The hyphen character may appear in component names only when used as a separator between parts of a single word, phrase, or value that would otherwise be incomprehensible without the use of a separator.	NDR Rule 8-3
Word Forms		
	A noun used as a term in a component must be used in singular form, unless the concept itself is plural.	NDR Rule 8-8
	A verb used as a term in a NIEM component must be used in the present tense, unless the concept itself is past tense.	NDR Rule 8-9
	Articles, conjunctions, and prepositions shall not be used in component names, except where they are required for clarity or by standard convention.	NDR Rule 8-10
Name Generation		
	With exceptions, property names shall be in the form: object class qualifier terms (0+) object class term (1) property qualifier terms (0+) property term (1) representation qualifier terms (0+) representation term (1)	NDR Rule 8-11
Object Class Term		
	The object class term of a component shall consist of a term identifying a category of concrete concepts or entities.	NDR Rule 8-12
Property Term		
	A property term shall describe or represent a characteristic or subpart of an entity or concept.	NDR Rule 8-13
Qualifier Terms		
	Multiple qualifier terms may be used within a component name as necessary to ensure clarity and uniqueness within its namespace and usage context.	NDR Rule 8-14
	The number of qualifier terms should be limited to the absolute minimum required to make the component name unique and understandable.	NDR Rule 8-15
	Qualifier order should not be used to differentiate names.	NDR Rule 8-16
Representation Terms		
	If any word in the representation term is redundant with any word in the property term, one occurrence should be deleted.	NDR Rule 8-17

Appendix C: Acronyms and Abbreviations

CSA	Component Staging Area
IEPD	Information Exchange Package Documentation
NBAC	NIEM Business Architecture Committee
NCCT	NIEM Configuration and Control Tool
NDR	Naming and Design Rules
NIEM	National Information Exchange Model
NTAC	NIEM Technical Architecture Committee
PMO	Program Management Office

Appendix D: References

ISO 11179, Part 4: ISO/IEC 11179-4:2004, Information technology—Metadata registries (MDR)—Part 4: Formulation of data definitions. Second edition, 2004-07-15. Available from http://standards.iso.org/ittf/PubliclyAvailableStandards/.

ISO 11179, Part 5: ISO/IEC 11179-5:2005, Information technology—Metadata registries (MDR)—Part 5: Naming and identification principles. Second edition, 2005-09-01.

NDR: The NIEM Naming and Design Rules specification. Available from http://niem.gtri.gatech.edu/niemtools/documentation.iepd.

NIEM High-Level Tool Architecture (under development—NTAC Sharepoint).

NIEM Version Architecture specification (under development—NTAC Sharepoint).

“Proposed NIEM Design Quality Metrics,” SEARCH, 2007. (NTAC Sharepoint.)

Visser, Joost, “Structure Metrics for XML Schema,” 2006. http://www.di.uminho.pt/~joost.visser.