Specification for a Common Data Format for National Voter Registration Act (NVRA) Form Registration Interchanges

Sam Dana, Prometheus Computing
John Dziurlaj, Ohio SoS Office
Ericka Haas, ERIC for the States
John Sebes, OSET Foundation
John P. Wack, NIST
Sarah Whitt, Wisconsin GAB
and members of the NIST Voting Interoperability Public Working Group

Draft Version 0.1
July 01, 2016

Note: this draft specification is part of an eventual specification whose scope will include a common data format for the Federal PostCard Application (FPCA) form and other registration form as well as other typical interchanges with voter registration databases. This specification deals only with the data needed to support voter registration using the NVRA form.
Executive Summary

This publication is a specification for a common data format (CDF) for election-related data and for data interchange related to voter registration and related voter records usage. The format, known as the NIST VRI CDF, is comprehensive and detailed in its coverage of voter registration request data, yet at the same time very flexible, able to accommodate many voter registration record formats used throughout the U.S. This publication contains the XML schema specification, usage guidance, and background information about VRI.

The VRI CDF can be used as a common export format across different IT systems that exchange data related to voter registration or voter records. Such systems include: online voter registration (OVR) voter-facing systems; OVR back-end systems; motor vehicle department customer record management systems; voter record database systems; systems that produce or consume voter outcome data for EAVS [TBD ref], systems that perform cross-state voter record matching.

The VRI CDF is definitely not intended to provide a CDF for complete voter records in a voter records management system (VRMS). Rather, the assumption is that: existing states’ VRMSs have data schema that share a common core but also have many state-specific and system-specific elements; subsets of the common core can be used for data interchange. As a result, the VRI CDF is heavy use-case-based, focusing on specific types of data interchange already in common practice – rather than attempting a grand unification of disparate voter record data.

As a result, this specification was written primarily for the audience election IT organizations, already engaged and or preparing to be engaged in data interchange similar to the current cases of OVR dataflow, OVR/DMV matching, state/state matching.

The purpose of this specification is to provide this audience with ready-made CDFs to support in systems that match these use cases, as an alternative to the current practice where each system development project includes data modeling and data interchange design, implementation, and testing of one-off data interchanges. The CDF is intended to provide the modeling and interchange definition ready-made, enabling such projects to proceed without these one-off efforts.

The CDF is intended to accommodate 9 use cases for data interchange, though at present fully covers only the first use case (the NVRA form).

The XML schema associated with the VRI CDF is derived from a UML model designed to accommodate the different types of locality-specific voter registration processes across the U.S. and how these process share core data with local variations. The model was designed to accommodate different types of processes and their variations. The UML model can be re-used, modified, extended, and/or harmonized to meet the needs of other planned CDF specifications, for example voting devices such as electronic pollbooks, and systems that utilize offloaded pollbook records.
Table of Contents

TBD
1. Introduction

This publication is a specification for a common data format (CDF) for voter records data interchange related to registration. The specification includes a data model in UML (Unified Modeling Language) [ref TBD] that itemizes and defines the data involved in voter records data interchange related to registration. The XML format is derived from the UML model, as may be other non-XML formats such as JSON and CSV.

The primary features of this specification include:

- Major data elements and their attributes and associations are fully defined in a UML data model.
- The data model can be used to generate data formats for today’s voter records systems as well as for future systems to be developed.
- Election data and results can be reported at flexible levels from highly aggregated to very detailed.

Detailed instructions for implementation and use of the XML schema are also included.

1.1 Purpose

The purpose of this specification is to provide a data interchange format for voter records, with particular but not exclusive focus on interchange related to voter registration, especially online voter registration. A goal for specification is to include not only a concrete representation in XML (for use by interoperating systems to exchange voter record data with high fidelity to the semantics of the UML model), but also examples in lighter-weight formats such as JSON and CSV.

Advantages of using this specification include:

- A ready data interchange format for online voter registration (OVR) systems, removing the need for individual OVR system development projects to define data models and formats.
- Use in OVR systems currently being built, with early use and feedback on the CDF to inform continuing development of the CDF for additional use cases.
- Enabling software reference implementations of externalization and internalization of data in a CDF, promoting software re-use, and reduced efforts in development of systems that exchange voter record data.

Early use in OVR systems is enabled by the surge in OVR activities following broad interest in OVR adoption following the PCEA Final Report recommendations. Early experience with draft CDFs for data exchange (between state registration systems and DMV systems) should better inform ongoing work on a complete set of CDFs for all the uses cases in the sub-group’s scope, described below.
1.2 Audience

The intended audience of this specification includes election officials, VR system designers and developers, as well as others in the election community including the general public. Some background in election administration or technology is useful in understanding the material in this specification.

1.3 Motivation and Methodology

This document was motivated primarily to reduce the inherent diversity for U.S. election officials in exchanging data related to voter registration. The current varying systems involved and data produced often do not interoperate, adding more complexity to the process. Additionally, there are sometimes significant variations among different jurisdictions within a state as well among the states themselves in the way they automate the voter registration and related parts of voter record management.

NIST and a community of U.S. election officials, analysts, and voting system technologists analyzed varying VR scenarios and their associated data interchanges, to analyze existing practices and to create a standard data interchange format for emerging OVR systems. From this preliminary analysis, 9 use cases were developed:

1. **Digital NVRA Submission**: Digital VR applications forms transmitted within state OVR systems, or to state OVR systems by third party OVR systems, following the format of the NVRA voter registration application form.

2. **Digital FPCA Submission**: Similar transmissions in OVR processes, following the format of the FPCA form.

3. **Digital State OVR Submission**: Similar transmissions in OVR processes, following the format of a state-specific form.

4. **Digital VR Update Submission**: Similar application forms including: voter registration update (change of name, change of address); change of voter status; absentee ballot request.

5. **OVR Transfer**: Subsets of such digital applications used for: Motor Vehicle or other NVRA agencies to forward user data to state OVR systems; 3rd-party OVR registrars to transfer users and user data to state OVR systems.

6. **DMV Match**: Subsets of such digital applications exchanged between state VR systems and DMV or similar systems, to perform driver's license data matching as part of OVR processing.

7. **NVRA Notification**: Data exchanged by DMV (or similar systems of other NVRA agencies) and VR systems, as part of NVRA compliance to digitally notify VR systems of DMV (or other) records of customers that requested new voter registration. May also include:
   - Similar data push from agency of existing customer records recently updated with change-of-address, as part of semi-automated steps toward permanent voter registration; or
• Other forms of data exchange to VR systems that might facilitate elements of automatic and/or permanent voter registration.

8. Cross-State Records Match: Data interchange between state VR systems for and systems for records matching, e.g. the ERIC system, or as part of inter-state cross-check activities.

9. EAVS Submission: Subsets of voter records externalized from voter records systems for purposes of data aggregation and reporting, including but not limited to EAVS reporting.

The initial focus of work the first use case, because it is the common basis of current efforts to develop new OVR systems, as an increasing number of states pass legislation enabling or requiring it. This initial focus was intended to quickly establish a baseline abstract data model as the basis for extension in later work on other use cases.

A UML data model was subsequently generated to represent the data associated with the first use case, and to show how the data elements are related and organized. Finally, an XML schema was generated from the UML data model. The XML schema defines the rules of the XML format.

The advantages of using a UML data model as an intermediate step to generating an XML schema include that the model is independent of the concrete XML format (or other potential formats that could be derived); relationships between data elements are easier to correctly define and visualize when they are independent of any specific data format. If changes are needed to the XML format, one can make changes to the UML model and then generate a new version of the format using commercial products.

Note that this specification addresses U.S. governmental elections and is not intended for use “as is” in other types of elections or in other countries. However, the specification was written with the intention that it be adaptable to other election environments.

1.4 Document Structure

Section 2 provides background information about the VRI effort and its basic goals. Section 3 provides a more details about the use cases currently in scope for the VRI effort, and provides overview of the data model for VRI.

Section 4 provides detailed documentation of the complete data model, in the context of an XML schema. While the basis in a concrete XML schema provides for specificity in the detailed documentation, usage of the VRI data model can also be accomplished in other formats such as YML or CSV, following the same schema.

Section 5 provides an initial draft for usage guidelines, limited to the one use case (Digital NVRA Submission) supported by the current draft. As the VRI working group continues to work on other use cases, the data model will evolve, and later drafts of this specification will extend the usage guidelines to address those data models.

Appendices A, B, and C provide typical definitions of Acronyms, Glossary, and Referenced documents and related content.

Appendix D provides further information on work in progress that is not part of the main body of this specification, but will inform future versions. At present, the main work in progress topic is data model addresses.
2. Background

2.1 Genesis

The genesis of this CDF effort was the observation of duplication of efforts in the development of OVR systems, both state operated systems and 3rd party registrar (3PR) systems. In early work by the subgroup, analysis of OVR workflows and related workflows suggested several possibly similar use cases that could share a core data model. However, it was also recognized that the core data model could have a short term impact (and provide feedback for continuing work) if the first version of this specification focused on the needs of current or just-over-horizon OVR efforts.

As a result, the OVR workflow was first documented for requirements on CDFs, with particular focus on the “least common denominator” of digital instances of the NVRA form. Initial data modeling efforts focused on the NVRA form, but also included some tentative steps for meeting requirements of similar digital FPCAs and digital versions of state-specific VR forms. However, a complete handling of these was ruled out of the initial scope, because of a desire to defer the careful semantic modeling required to capture different policies related to absentee voting that are inherent in the FPCA and some state forms. For other state forms, data interchange can be accomplished by very minor extensions of the data model, e.g., additional enumeration values for VoterCharacteristics [fwd ref TBD].

2.2 VRI Data Exchange Workflow Overviews

This specification arose out of existing practices, and data interchange workflows. This section provides an overview of these workflows.

2.2.1 OVR Workflow Overview

Because the workflow of OVR influenced the requirements for the VRI CDF in its first version, a summary of a typical OVR workflow is in order. In many existing cases, OVR is essentially a simple process of data exchange by 2 interoperating systems, both operated by the same government authority, e.g., a state Secretary’s office or other state-level election authority.

The first of the 2 systems is a voter-facing user-interface system that implements online forms and user experience to collect largely the same data as existing paper forms. Such a “client” or “submitter” system’s job is largely done when user dialog activity is complete, and the registrant’s form information is marshaled into a data format for submission to the second system.

The second system’s role begins with receipt of a digital voter registration application form from the submitter system. The details of the receiver system (referred to by a variety of other terms, e.g., back-end, middle-end, middleware, wrapper, Web API service, and more) vary, but most often focus on storing the digitally submitted data for later review, where:

- The format and storage method of the stored data matches or mimics that of data stored in a legacy system used for human data entry from paper forms.
- The storage method has minimal or no requirement on the functioning of the legacy system, so that the latter may be considered the true back-end, and the receiver system a wrapper or intermediary system.
A closely related workflow is one in which the voter-facing system is operated by a 3PR, collects the same digital VR application form information, and transmits it to a state-operated system, often via an application programming interface (API) of the submitter system described above. The variety of representations of essentially the same data, in several different instance of this 3PR data interchange was one impetus to definition of this CDF.

Another 3PR workflow applies in cases where the 3PR system does not collect all the information required by the state system, and the purpose of the API (or other data exchange method) is to (a) transfer user-provided information to the state-operated submitter system, and (b) transition the user experience to the web UI of the submitter system, for the user to complete the required data entry.

2.2.2 VR Matching Overviews

Another common subset of the OVR process is a workflow that performs a DMV match step as part of OVR. A subset of the registrant’s personal data – include DMV ID number – is matched against DMV data to determine whether the ID number matches the rest of the personal data, and whether there is stored digital image of the person’s signature. This workflow is typical because of similar state laws that enable OVR as paperless because of an existing signature on file.

However, this type of VR data matching workflow is only one of several similar VR data match workflows, which notably include the use of the ERIC system [TBD ref] and the practice of interstate cross-check processes.
3. Use Cases and the VRI UML Model

This section describes the 9 use cases for VR data interchange covered by this specification, and then describes the UML model that implements the use cases and is used to generate the XML schema. This section is provided for background and to assist in understanding how the model and schema are structured and used.

3.1 Use Cases for OVR Data Interchange

The VRI CDF has the goal of addressing all the use cases described below. At present, however, this early version of the CDF is intended to focus on the first use case, in which the data corresponds to information on the Federal form (mandated by the NVRA (National Voter Registration Act) and distributed by the EAC (Election Assistance Commission). As a result, only the first use case below is described completely.

For each other use case, the current draft describes work to be done for the use case, partial support by the existing model, and high level requirements for extensions to the model.

3.1.1 Use Case: Digital NVRA Form Submission

This use case – the “NVRA Use Case” for short – is for the exchange of a digital form of the paper NVRA form [TBD ref], as part of an OVR workflow. The current model and schema cover all of the voter data fields specified by the NVRA form, in the core data element Voter described below. The current model and schema also includes meta-data about an NVRA form and the submission of it from a requesting party to a receiving party; see the VoterRecordsRequest and VoterRecordsResponse core data elements described below.

3.1.2 Use Case: Digital FPCA Form Submission

This use case – the “FPCA Use Case” for short – is for the exchange of a digital form of the paper FPCA form [TBD ref], as part of an OVR workflow. The current model and schema cover many of the FPCA voter data field simply because of fields in common with the NVRA. Also covered are FPCA-related extensions to schema enumeration types, for example, voter types for the several different types of UOCAVA voters. In addition, some FPCA-specific data fields are supported, e.g., for ballot delivery preference.

Also notably included are data model elements intended to capture state-specific data fields that are not part of the FPCA form directly, but included by reference to state-specific preparation instructions that, as a set, can change over time.

There is one primary reason why these parts of the current model and schema do not support the FPCA use case as completely as the NVRA use case. The sub-group specifically did not want to tackle, in the first version, the varying and potentially subtle semantics of the absentee request that, unlike the NVRA form, is a fundamental part of the FPCA semantics.

Continuing work on the use case will primarily address absentee semantics, as well as reaching closure on representation of non-NVRA FPCA data fields, including international address representation.

3.1.3 Use Case: Digital State OVR Submission
This use case, like the FPCA use case, is essentially an extension of the data representation development of the NVRA use case. However, this use case spans potentially all forms that are used by individual states as an alternative to the NVRA form, capturing state specific fields not present in the NVRA form or the FPCA form.

Continuing work on the use case will clearly involve compilation and analysis of many state forms, including leveraging such analysis already performed. One important “low hanging fruit” part of this work will be extended enumeration types for state practices, for example the `VoterClassificationType` enumeration would need an extension for Rhode Island’s form’s “over 16 years of age” characteristic. More challenging parts of this work will include data modeling of wholly new data elements specific to states.

### 3.1.4 Use Case: Digital VR Update Submission

This use case addresses the different semantics of VR requests that are updates to existing records. The core data model may not require much extension, but the schema will likely require extended meta-data to capture the different semantics of an update request – for example, extensions to the enumerations for `SuccessAction` and `RegistrationRequestType` that could serve to clarify the distinctions between the enumerations.

### 3.1.5 Use Case: OVR Transfer

This use case is intended to extend the model and schema as needed to cover the data exchanged between a 3rd party system (including NGO operated OVR system, or a government operated system, e.g. DMV) and a state OVR system. In many instances of this use case, the data exchanged may be a subset of the NVRA data, but additional meta-data may be required.

### 3.1.6 Use Case: DMV Match

This use case is likely to be a subset of the NVRA use case, but will need to capture two-way information flow, for example: a request with a subset of VR personal information include DMV ID number; a response that could indicate “no match” or other error conditions, but could also include a match that returns VR personal information on file, particularly current address.

### 3.1.7 Use Case: DMV Notification

This use case is likely to be very similar to the NVRA use case, including the same NVRA form data, but perhaps including additional meta-data, such as the type of NVRA agency that is submitting a digital NVRA form on behalf of a customer of the agency.

### 3.1.8 Use Case: Cross-State Records Match

This use case may be a subset of the NVRA use case, in terms of voter data exchanged, but may require additional transactional semantics, and alternative data representations to omit or mask PII.

### 3.1.9 Use Case: EAVS Submission

This use case will likely use a subset of the voter descriptive data of the current core schema, using a subset to describe a single voter in terms of the voter data required to create EAVS survey
submissions. For example, it would include those fields that describe the voter characteristics pertinent to EAVS (e.g. UOCAVA characteristics, non-UOCAVA absentee voter characteristics), and other information about a voter that does not include PII, such as year of birth, zip code of address, party affiliation, race/ethnicity, and other demographic data.

In addition, however, work on this use case will likely extend the data model and schema to record a “voter outcome” in addition to information about the voter herself. For EAVS, the core voter outcomes are events such as “absentee ballot counted” and “absentee ballot not counted” and “reason for absentee ballot not counted”.

It is hoped that a narrow scope of data modeling of voter outcomes can avoid scope creep into the broader area of “voter participation history” in a voter record, but nevertheless express an election-specific voter characteristic that captures the voter’s participation (or lack thereof). Though not included in the current specification, early data modeling indicated that the core of these extensions might be limited to two new enumeration types, for outcome and outcome-reason. As with many election data CDFs, these enumerations will be difficult to compile in a complete and definitive way, and may require a method for the enumerations to be extended without requiring a re-issuance of the CDF specification document.

### 3.2 The OVR UML Model

This section presents the UML model that was structured to implement the use cases. It includes a class diagram that shows a picture of the model and an overview of how to read the relationships between the classes so as to understand how the model and XML schema are structured.

The UML model represents a format-independent description of the data involved in voter record data interchange as required by the use cases of the specification. Its primary benefit is that it unambiguously defines and describes the data elements and how they are related without requiring readers to know the technical details of any particular data format implementation such as XML. By using a model-based approach, the resultant data format is more likely to be well-structured and more tolerant to modifications. The data format can be generated from the model using commercial tools, thus if changes need to be made to a format, the model can be changed and the format can be re-generated.
Figure 1 shows a high-level view of the class diagram for a **VoterRecordsRequest**, minus its attributes and certain ancillary classes.

Figure 2 shows a high-level view of the class diagram for a **VoterRecordsResponse**, minus its attributes and certain ancillary classes.
3.2.1 Major Classes

The abstract schema for the CDF centers on a **VoterRegistration** object, its attributes, elements, and subsidiary objects. The following lists the major data elements of the model, along with a note about its status in the current draft.

The data model is comprised of two main parts, a request for voter record data interchange, and a response. The request model includes support for OVR requests, as well as at least partial other forms voter record data interchange, in use cases that have not yet been addressed. The response model currently supports response to OVR requests, to indicate to a requestor whether the request as valid, if not why not, and if so what its disposition is.

- **VoterRegistration** is the core element of the model, with attributes that record voter information such as date of birth, race or ethnicity, gender, email, phone, fax. These, along with subsidiary elements noted below, define all the elements and attributes that model an OVR request based on the NVRA form, along with a few additional components that will need to be refined other use cases are considered and new elements added for them.
  - **Name** is comprised of the elements of the name of a person. It is required element of a **Voter** and may also be an element of a **RegistrationAssistant**.
  - **PreviousName** has the same structure, used to record a voter’s previous name.
- **VoterClassification** records one or more attributes of a voter, such as eligibility requirements that are indicated in a voter registration request (citizen, non-felon, etc.) or various kinds of UOCAVA status. As more use cases are considered, the enumeration type for this element will likely be revised and extended.

- **VoterId** includes the elements that describe a government identification document of the voter’s, include the document’s type, ID number, provider, date of issuance, and digital image.

- **ContactMethod** represents a method of contacting the voter, e.g. via phone, email. A **VoterRegistration** can contain multiple **ContactMethod** instances. In the case of **ContactMethod** of type Phone, the phone can have one or more capabilities such as voice, fax, SMS.

- **Party** indicates the voter’s party affiliation, both by a string name, and an external identifier.

- **Signature** is used to record an image and metadata of a person’s manual signature, for cases where a signature image is captured as part of a voter registration request, including the registrant’s signature.

- **RegistrationAssistant** represents the information required when the registrant was assisted by another person. It can include a **Signature**.

- **RegistrationWitness** represents the information required in situations where a voter registration request needs to be witnessed by another person. It can include a **Signature**.

Note: Several of these subsidiary elements are expected to be extended as more use cases are considered, particularly the enumeration types for **VoterClassification**, and for the kinds and issuers of **VoterId**. In addition, the model also include a subsidiary object of a **Voter**, called **AdditionalInfo** is not needed for the initial use case, but will be needed for future work, where changing state-specific extensions to the FPCA need to be modeled.

- **VoterRecordsRequest** is the top-level element that embodies the semantics of a data interchange of a voter registration request and other similar interchanges. It can indicate which kind of action is being requested – create a voter registration record, update one, read one, etc. – as well as some specific actions such as change of address, update deceased status, etc. It notes which if any of existing standard forms is being represented digitally, e.g. an FPCA or the Federal standard form (a.k.a. NVRA form). It also other records elements pertinent not to the semantics of the **VoterRegistration** itself, but to the dataset being exchanged by interoperating parties, for example, the issuer of the request, the date and time of issuance.

  - **VoterRecordsDataRequestSource** is an optional element that can describe the organization that is making the request. In an OVR context, it might be omitted if the requester is a voter making a request directly, but might be included in the case where the requester is third party registrar or NVRA agency that is making the request on the voter’s behalf.

- **VoterRecordsResponse** is the complement to **VoterRecordsRequest**. Modeling on this element is in progress, mainly limited to the OVR related use cases, and the types of response data an OVR back-end system might return to the sender of a **VoterRegistration**. The main purpose in OVR is represent either an error response or an acknowledgement response from an entity that received a the request.
In addition to these elements, the model also includes a variety of enumerations, most of which are referred to above, and all of which are illustrated in Figure TBD.

### 3.2.2 Addresses: Linkage to External Standards

Perhaps the most complex part of a voter registration request, and other related data exchange of voter record data, consists of addresses. There are multiple kinds of addresses for VR purposes (current registration address, previous registration address, mailing address, overseas address), and multiple sub-types of addresses for location purposes and for mailing purposes. The schema supports the following kinds of VR addresses and subtypes:

- **Registration Address and Previous Registration Address**
  - A structured street address with distinct elements of street name, number, type, and other structured components, plus city, state, and ZIP.
  - An unstructured street address with a single string for street address, plus city, state, and ZIP.
  - A location description address with city, state, zip, and a description of a location that doesn’t have a street address.

- **Mailing Address and Mail Forwarding Address**
  - A structured street address as above.
  - An unstructured street address as above.
  - P.O Box address.
  - Military or diplomatic address, e.g. APO, DPO, FPO.
  - Rural Route address.
  - Non-US address with country and postal code, and other elements sufficient for a local election official to create a functional postal address for international mailing.

Each of these subtypes of address needs to be represented in this CDF. However, rather than revisit the complexities of address structure, the sub-group decided to include existing external standards. At present, the CDF references one such standard, the *United States Thoroughfare, Landmark, and Postal Address* data standard [TBD ref], issued by TBD, and covering all the complexity of addresses managed by or encountered by USPS.

While leveraging the work of external standards, it is not intended that users of this CDF should master all the detail of external standards. Rather, this CDF maps each of the subtypes listed above, to a single canonical address structure in the external standard. The intention is that producers of datasets would represent each address as one of these subtypes, and use the canonical structure defined in this specification.

These subtypes fall into 2 categories at present: those that can be completely captured by the use of the FGDC standard in its current form; those that can be partially represented by current FGDC, but will be the focus of ongoing work to revise that standard for more complete representation. The next two sections describe each of these two categories, with references to worked examples for the former, and references to appendix material for work-in-progress on the latter.

#### 3.2.2.1 Voter Registration Request Address SubTypes Represented by FGDC Structures
The following list provides a mapping from each of these sub-types to an FGDC structure. A detailed canonical example of an FGDC structure is provided for each of these sub-types, in Section 5.2.

- Structured street address: an FGDC `NumberedThoroughfareAddress`.
- P.O. Box Address: an FGDC `USPSPostalDeliveryBox`.
- Rural Route Address: an FGDC `USPSPostalDeliveryRoute` structure using the typical route and box number structure, combined with typical place, state, and ZIP.
- Military or Diplomatic Address: an FGDC `USPSPostalDeliveryRoute` structure where “APO” (or similar) serves as a place name, and the state is one of a defined set of statelike destinations, “AE”.
  - Note: `USPSPostalDeliveryRoute` is used for both rural route and military address subtypes, a later version of this specification may need to provide specific usage guidelines to enable consumers to differentiate these two address types.

### 3.2.2.2 Voter Registration Request Address SubTypes Represented by FGDC General Address Class

The FGDC `GeneralAddressClass` object is at present the only FGDC option for the following remaining VR address subtypes. `GeneralAddressClass` is the basic fallback of an address represented as a single undifferentiated string. However, the following subtypes require some additional structure to be worked out in upcoming revisions of that standard, and this specification document.

For each of the 3 subtypes below, a simple example will show the scope for additional structure. Appendix D provides additional information on work-in-progress for adding additional structure, while also specifying how to use FGDC in the current version of this specification.

- Location description address: an unstructured “street address” that isn’t actually an address, coupled with structure to provide city, state, and ZIP. Current example:
  ```xml
  <GeneralAddressClass>
  "The camp under I-10 near Earheart, behind the Home Depot. New Orleans, LA, 70125"
  </GeneralAddressClass>
  ```

- Un-structured street address: an unstructured true street address that is not separated into separate elements, coupled with structure to provide city, state, and ZIP. Current example:
  ```xml
  <GeneralAddressClass>
  "4242 Joseph St, Second Floor
  New Orleans, LA, 70125"
  </GeneralAddressClass>
  ```

- Non-US street address: an unstructured true street address that is not separated into separate elements, coupled with structure to provide very basic common international address structures like country, postal code, municipality. The very modest goal here is to enable US election officials to have data required to effectively address mailing to a
registered voter living abroad, but without being definitively correct in the exact mailing address structure of any or all other countries. In other words “Good enough for a good faith effort to get it to the voter.” Current example:

```xml
<GeneralAddressClass>
  "2 Fogaras ut.
  Pecs
  Hungary
  7624"
</GeneralAddressClass>
```
4. XML Schema

**NOTE:** This section is being developed separately by John Wack for inclusion when complete.

This section contains documentation and discussion of the features included in the XML schema that follows the UML model described above.
5. Schema Aspects and Usage

This section presents an overview of various aspects and usage of the schema, including
1. General layout of instance files
2. How UML relationships from Section 3 are implemented in the XML schema
3. How addressing is implemented for registration addresses and mailing addresses.

No attempt is made to provide a complete overview of usage, but more to describe in general how most elements are to be used. Guidance is limited to the one use case supported by this version of the specification.

5.1 General Layout of XML Instance Files

For the use case of digital NVRA form submission, the layout of the instance file is fairly straightforward. After the top level VoterRecordsRequest and its optional subsidiary, the core of the instance file is the VoterRegistration, followed by some or all of its subsidiaries (e.g., Name, Signature, VoterID) as described in Section 3.2.1.

5.2 Examples of Addresses

Within this typical structure of an instance file, within the Voter structure, there are several options for representation of addresses, as described in Section 3 above. The below are some specific examples of each type of address, with yellow highlight for the familiar elements of the content, broken out from the XML representation of the FGDC standard.

5.2.1 FGDC USPSPostalDeliveryBox

<USPSPostalDeliveryBox>
  <USPSBox>
    <USPSBoxType>PO BOX</USPSBoxType>
    <USPSBoxId>00145</USPSBoxId>
  </USPSBox>
  <CompletePlaceName Separator="">
    <PlaceName
      PlaceNameType="MunicipalJurisdiction"
      ElementSequenceNumber="0"
      GNISFeatureID="0">
      CLEVELAND
    </PlaceName>
  </CompletePlaceName>
  <StateName>OHIO</StateName>
  <ZipCode>43101</ZipCode>
  <ZipPlus4>1111</ZipPlus4>
</USPSPostalDeliveryBox>

5.2.2 FGDC USPSPostalDeliveryRoute Used for Military Address

<USPSPostalDeliveryRoute>
  <USPSAddress>
    <USPSRoute>
      <USPSBoxGroupType>PSC</USPSBoxGroupType>
    </USPSRoute>
  </USPSAddress>
</USPSPostalDeliveryRoute>
<USPSBoxGroupId>4</USPSBoxGroupId>
</USPSRoute>
<USPSBox>
  <USPSBoxType>BOX</USPSBoxType>
  <USPSBoxId>3</USPSBoxId>
</USPSAddress>
<CompletePlaceName Separator=""/>
  <PlaceName
    PlaceNameType="MunicipalJurisdiction"
    ElementSequenceNumber="0"
    GNISFeatureID="0">
    APO
  </PlaceName>
</CompletePlaceName>
  <StateName>AE</StateName>
  <ZipCode>96278</ZipCode>
</USPSPostalDeliveryRoute>

5.2.3 FGDC USPSPostalDeliveryRoute Used for Rural Route

<USPSPostalDeliveryRoute>
  <USPSAddress>
    <USPSRoute>
      <USPSBoxGroupType>Rural Route</USPSBoxGroupType>
      <USPSBoxGroupId>4</USPSBoxGroupId>
    </USPSRoute>
    <USPSBox>
      <USPSBoxType>Box</USPSBoxType>
      <USPSBoxId>9</USPSBoxId>
    </USPSBox>
  </USPSAddress>
  <CompletePlaceName Separator=""/>
  <PlaceName
    PlaceNameType="MunicipalJurisdiction"
    ElementSequenceNumber="0"
    GNISFeatureID="0">
    Wynne
  </PlaceName>
</CompletePlaceName>
  <StateName>AR</StateName>
  <ZipCode>72396</ZipCode>
</USPSPostalDeliveryRoute>

5.2.4 FGDC “NumberedThoroughfareAddress”

<NumberedThoroughfareAddress>
  <CompleteAddressNumber
    AddressNumberParity="Even"
    AttachedElement="Not Attached">
    A
  </AddressNumberPrefix>
</AddressNumberPrefix>
Appendix A  Acronyms

Appendix B  Glossary

Appendix C  References

Appendix D – Work in Progress on Address Structures

At present, the FGDC standard does not provide adequate structure for 3 of the address subtypes required by this specification. Sections below provide some of the work-in-progress that may include a minor update to FGDC that fills these gaps.

- Un-structured street address: in progress; one possibility is to encode the single string of street address as a LocationDescription, and dis-ambiguate via an enumeration in an enclosing object specified in this schema.
  - Note: at present, both the enclosing structure, and the single address element are not yet defined.

- Non-US address: not covered in FGDC.
  - Note: work is in progress to determine what address elements (besides country and postal code) are minimally sufficient for a U.S. local election official to create a functional address for international mailing. Candidates include non-semantic elements (AddressLine1, AddressLine2, etc.) and semantic elements (street address, locations, municipality, state/province/division, etc.).

FGDC Location Address

A Location Address consists of an FGDC LocationDescription element combined with Place, State, and ZIP enclosed in an FGDC structure that is to be determined. The following provides an example of how a Location Address might be structured.

<TBD>
  <LocationDescription>
    "The camp under I-10 near Earheart, behind the Home Depot"
  </LocationDescription>
  <CompletePlaceName Separator=""/>
    <PlaceName
      PlaceNameType="MunicipalJurisdiction"
      ElementSequenceNumber="0"
      GNISFeatureID="0">
      NEW ORLEANS
    </PlaceName>
  </CompletePlaceName>
  <StateName>LA</StateName>
  <ZipCode>70125</ZipCode>
</TBD>
FGDC Location Used for Unstructured Address

An unstructured voter registration address is actually unstructured in the street address, but requires the usual structures for place, state, and ZIP. One possible approach to representing this subtype of address is to use the same LocationDescription as in the above case, but with a different enclosing object that is to be determined.

<
TBD
(LocationDescription>
  “123 MAIN ST. Second Float”
</LocationDescription>
<CompletePlaceName Separator="">  
  <PlaceName  
    PlaceNameType="MunicipalJurisdiction"  
    ElementSequenceNumber="0"  
    GNISFeatureID="0">  
    AKRON  
  </PlaceName>
</CompletePlaceName>
<StateName>OH</StateName>
<ZipCode>44320</ZipCode>
</TBD>

Non-US Address With Semantics

Non-US addresses can be part of Voter Records, as a mailing address of a voter living abroad. Voter records need to have a recorded non-US address that is sufficient for addressing postal mail to the voter with a reasonable chance of delivery based on the information provided by the voter. However, it is far beyond scope to have an address CDF that can encompass all counties’ variations in postal addressing, into which the voter-provided data can be cast for maximum likelihood of delivery.

Also, it may or may not be the case that FGDC can encompass even a basic structure for international addresses, either a simple sequence of AddressLine elements (to provide conventional line breaks in voter-provided addresses), or AddressLine elements coupled with a few named fields with conventional semantics. An example of the latter is below.

<
InternationalAddress>
  <AddressLine> Rudas ut. 5, Emelek 2 </AddressLine>
  <Locality>Pecs</Locality>
  <StateProvinceDivision> </StateProvinceDivision>
  <Country> Hungary </Country>
  <PostalCode> 7624 </PostalCode>
</InternationalAddress>